sensors

MDPI

Review

A Comprehensive Survey on the Non-Invasive Passive Side-Channel Analysis

Petr Socha , Vojtěch Miškovský and Martin Novotný *

Department of Digital Design, Faculty of Information Technology, Czech Technical University in Prague, 160 00 Prague, Czech Republic

* Correspondence: martin.novotny@fit.cvut.cz

Abstract: Side-channel analysis has become a widely recognized threat to the security of cryptographic implementations. Different side-channel attacks, as well as countermeasures, have been proposed in the literature. Such attacks pose a severe threat to both hardware and software cryptographic implementations, especially in the IoT environment where the attacker may easily gain physical access to a device, leaving it vulnerable to tampering. In this paper, we provide a comprehensive survey regarding the non-invasive passive side-channel analysis. We describe both non-profiled and profiled attacks, related security metrics, countermeasures against such attacks, and leakage-assessment methodologies, as available in the literature of more than twenty years of research.

Keywords: side-channel analysis; side-channel attacks; side-channel countermeasures; embedded systems; security

Citation: Socha, P.; Miškovský, V.; Novotný, M. A Comprehensive Survey on the Non-Invasive Passive Side-Channel Analysis. *Sensors* **2022**, *22*, 8096. https://doi.org/10.3390/s22218096

Academic Editors: Lei Xu and Xinxin Fan

Received: 28 September 2022
Accepted: 19 October 2022
Published: 22 October 2022

1. Introduction

In the past few decades, computer systems and communication networks have become an essential part of our everyday lives. Various computing devices are used not only as tools for many professionals, but also for entertainment. These devices include embedded devices, such as payment cards, biometric passports, smart cars, trains, or whole cities, and even medical devices like pacemakers. Being surrounded by devices connected to the Internet, our private lives are endangered more than ever [1].

Special attention must therefore be given to ensure security of computer systems and their users. Various measures are employed to achieve confidentiality, integrity, availability, and non-repudiation of data with efficiency, ease of use, and cost in mind. Nowadays, widely used algorithms, such as Rijndael/AES [2,3] or RSA [4], are considered secure from the cryptoanalytic point of view. However, their implementations may leak sensitive information through the cryptographic device's side channels, potentially compromising the entire system.

Side-channel attacks exploit the data-dependent side channels, such as power consumption of the cryptographic device [5,6] or its electromagnetic radiation [7], in order to extract secret information such as cipher keys. Such attacks pose a severe threat to both hardware and software cryptographic implementations, especially in the IoT environment where the attacker may easily gain physical access to a device, leaving it vulnerable to tampering. Various countermeasures have been proposed to prevent such attacks. Masking is a widely used technique based on randomization of the processed data [8–11], making it difficult to exploit the leakage. Hiding is another common approach, which aims to conceal the exploitable leakage in either side-channel signal amplitude or time [12–15]. Recent real-world attack examples show that uncompromising protection and testing of embedded cryptographic implementations is necessary [16].

This paper presents theoretical background and the state of the art in the area of non-invasive passive side-channel attacks. We map the history of this field and provide both a theoretical and practical overview. We present a systematic classification of both side-channel attacks and side-channel countermeasures and describe these. Therefore, our publication can serve as a good starting point for new side-channel researchers, as well as a universal reference. It is structured into seven sections as follows:

- Section 2, Side-Channel Security: Introduces side-channel leakage origin, measurement setup, formal model of the leakage, and leakage functions.
- Section 3, Non-Profiled Attacks: Describes non-invasive passive non-profiled attacks.
- Section 4, Profiled Attacks: Describes non-invasive passive profiled attacks.
- Section 5, Side-Channel Related Metrics: Describes both experimental and theoretical attack-related metrics.
- Section 6, Countermeasures Against Attacks: Describes hiding and masking countermeasures.
- Section 7, Attacks on Protected Implementations: Describes extensions of the presented attacks for attacking implementations with countermeasures.
- Section 8, Leakage Assessment: Describes methods for evaluations of side-channel leakage.

2. Side-Channel Security

Side channels of digital systems that may be used to compromise the system include power consumption [6,17–19], electromagnetic radiation [7], combinational logic delay [20,21], timing [5], and more. Some of these side channels are mutually dependent. For example, the relationship between current intensity and the magnetic field can be shown, e.g., by Biot–Savart law [22], and the combinational logic delay can be convincingly modeled as inversely proportional to the voltage drop [23]. This paper focuses on the dynamic power consumption side channel, but our presented concepts may be relevant for other side channels as well.

Side-channel attacks may be classified in many different ways [22], such as invasive/non-invasive or active/passive. Invasive attacks require depackaging the chip in order to access internal components, such as data buses, whereas non-invasive attacks only exploit the external access. Active attacks tamper with proper functionality of the device (e.g., by introducing faults), whereas passive attacks only make use of observation of the device during its undisturbed operation. This paper focuses on non-invasive passive attacks only.

Side-channel attacks can also be classified as either horizontal or vertical. Horizontal attacks exploit leakage during a single algorithm execution, whereas vertical attacks exploit leakage from multiple executions. For example, considering a hardware implementation of the RSA algorithm that uses naïve square and multiply exponentiation, either only square operations are performed, or both square and multiply operations are performed during computation, for every exponent bit, depending on the bit being zero or one. This not only influences execution time, but, in some cases, it also allows the attacker to directly read the secret key from a single measured power/EM trace by graphing the trace, as the two operations form distinctive patterns [5]. This kind of a horizontal side-channel attack is called simple power analysis. Unlike this simple example, this paper focuses on vertical side-channel attacks, where the information is typically contained in the instantaneous signal amplitude as further described below.

A CMOS inverter model is depicted in Figure 1. Three different dissipation sources can be observed in such a CMOS structure [24]:

- static leakage current,
- short-circuit current, and
- capacitance charge and discharge.

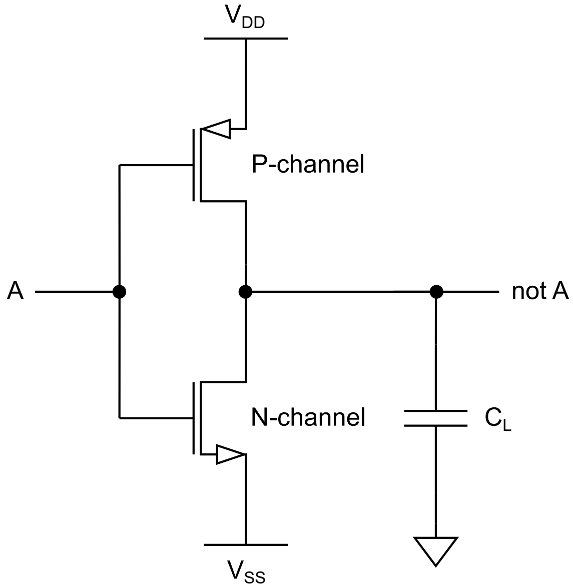


Figure 1. A CMOS inverter model.

When the inverter input presents a stable voltage corresponding to 0 or 1, one of the transistors is open and the other one is closed. In this case, only static leakage current is present. When the input changes, short-circuit current can be observed for a brief period of time when both transistors are open. Furthermore, the modeled load capacitance C_L has to be charged to the proper voltage when the input changes its value. Therefore, based on the instantaneous current consumption, it can be easily distinguished whether a transition happened or not. This fact is exploited by the most common leakage models as described later in this section. The consumption during a transition is demonstrated in Figure 2.

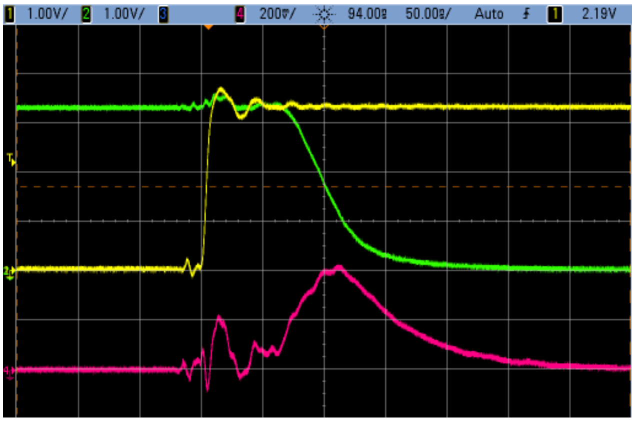


Figure 2. A CMOS inverter current consumption. The yellow line is the inverter input, the green line is the inverter output. The pink line is the current consumption, where peaks during the transition are clearly observable.

Because the P-channel MOSFET majority carriers have lower mobility and the minority carriers have lower lifetime, in contrast to the N-channel MOSFET [25,26], the P-channel MOSFET is typically built larger than the N-channel MOSFET [25], resulting in different characteristics, most importantly on-resistance and propagation delay (for non-inverter gates) [27]. Due to differences between N-channel and P-channel MOSFETs, the output value after transition can also be distinguished by the instantaneous current [28].

This simple example illustrates data dependency of the instantaneous power consumption, which is the main cause of the power-related side-channel information leakage in CMOS-based integrated circuits. Vertical attacks exploiting this kind of leakage typically require multiple side-channel measurements, unlike the previously described simple power analysis attack.

2.1. Measurements

The device's side channel is typically observed during a cryptographic operation, resulting in a measurement record, so-called trace, i.e., a vector of samples. For example, a single trace of dynamic power consumption during Rijndael/AES encryption in an FPGA is visualized in Figure 3. As mentioned earlier, multiple aligned traces, such as this one, are typically required for a successful attack, although single-trace attacks are sometimes also possible. This subsection briefly describes different measurement methods.

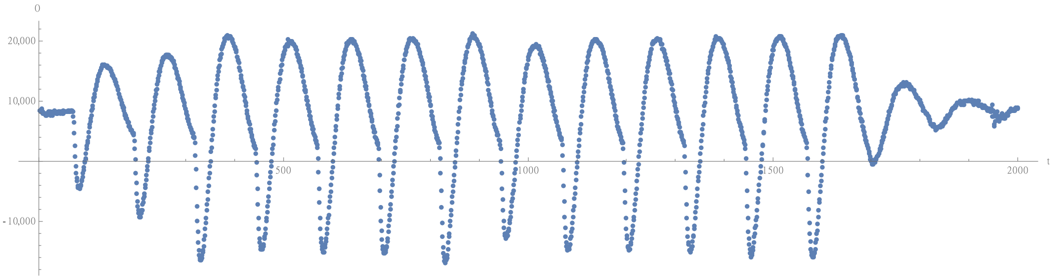


Figure 3. Rijndael/AES encryption FPGA power trace.

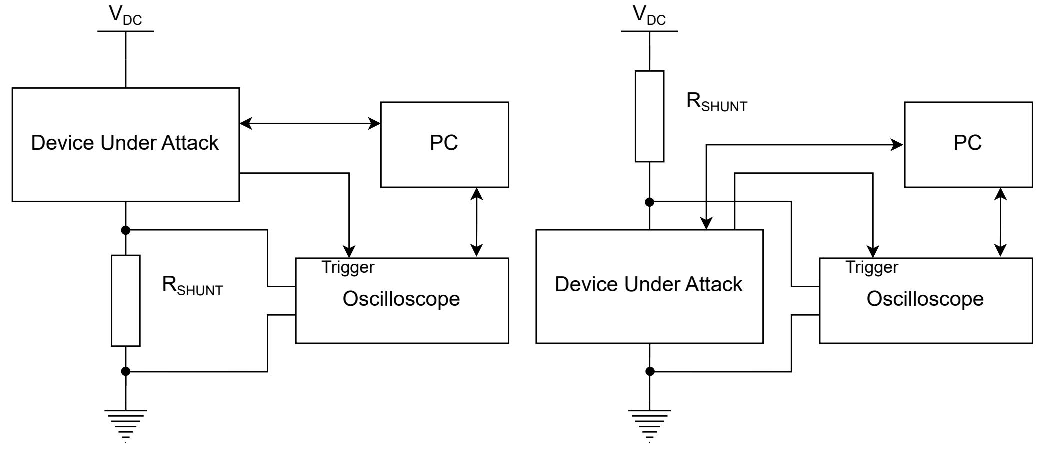
2.1.1. Power Consumption

Power consumption of the cryptographic device is typically measured by using an oscilloscope which samples voltage across a shunt resistor. The current can then be obtained, knowing both resistance and voltage, by using Ohm's law $I = \frac{U}{R}$. However, raw ADC values corresponding to the voltage can be directly used in a typical attack scenario, because the current and the voltage are assumed to be linearly dependent, as long as the oscilloscope setup parameters are consistent during all measurements.

Various measurement setups are described in [29], the differences being primarily in the shunt resistor placement.

- Shunt resistor in GND path, with the voltage across the resistor being sampled by the oscilloscope, as shown in Figure 4a.
- Shunt resistor in V_{DC} path, with the voltage across the cryptographic device being sampled by the oscilloscope, as shown in Figure 4b, also observing voltage drops of the power regulator.

The latter setup offers an advantage when the device under attack has multiple power networks, because it allows the attacker to measure the just cryptographic core consumption. When the voltage is measured by using a shunt resistor in the ground path (Figure 4a), the measured voltage typically contains more noise such as noise caused by the device's peripheral drivers. When measuring in the V_{DC} path (Figure 4b), the DC shift must be removed (unless measuring in a differential mode), which can be ensured either by using the oscilloscope's AC mode, or by using external DC blocker. Other choices for power measurement include differential or current probes. However, these are not recommended unless necessary, as they present an additional source of environmental noise [29].



(**a**) GND path measurement (**b**) V_{DC} path measurement

Figure 4. Example of a power measurement setup.

In a real-world attack, any decoupling capacitors near the cryptographic core must be removed, as they might filter out the relevant voltage changes. Correct power measurement setup is crucial for successful side-channel analysis. Parameters such as the environmental noise, sampling rate, or synchronization jitter have a direct impact on the attack success [30].

2.1.2. Electromagnetic Radiation

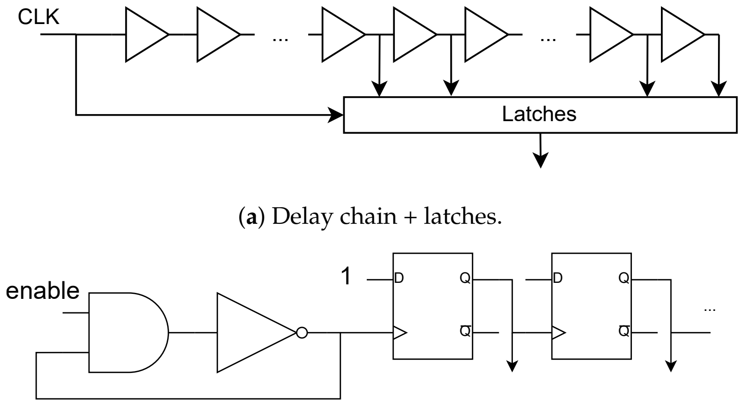
Similarly to power measurement, an oscilloscope connected to a near-field probe may be used for measurement of electromagnetic radiation [7]. As mentioned earlier, there is a close relationship between the power consumption and the radiation [22].

Attacking electromagnetic radiation offers more degrees of freedom compared to the power analysis. The attacker can examine a particular part of the chip only, and she can choose from a wide variety of probes. Consequently, EM analysis may provide a very powerful tool at the cost of more intricate and more costly employment. Further discussion of EM side channels is outside of the scope of this paper.

In addition to directly measuring electromagnetic radiation of the device under attack, data-dependent leakage may also be unintentionally broadcast by a radio transmitter present on the same chip (such as SoC bluetooth/WiFi transmitters with built-in encryption) [31]. In mixed-signal systems on chip, the leakage from the digital part of the chip couples through substrate to the high-frequency analog radio transmitter [32]. This class of attacks is called screaming channels and it allows the attacker to successfully reveal cipher keys from traces obtained by a radio receiver from even 15 m distance [33].

2.1.3. Combinational Logic Delay

Combinational logic delay inside a chip can be satisfactorily modeled as inversely proportional to the voltage drop of the internal power network [23], which is data-dependent due to the switching activity. In FPGA chips, the delay can be measured internally by using a delay-chain monitor [20,34], shown in Figure 5a, or by using a ring oscillator monitor [21], shown in Figure 5b. Acquired delay traces can be used in side-channel analysis in a similar fashion as the power traces.



(**a**) Delay chain + latches.

(**b**) Ring oscillator + asynchronous counter.

Figure 5. Combinatorial logic delay monitors.

Furthermore, crosstalk between two long wires inside the chip can be detected [35] by using a ring oscillator monitor as a receiver. In a multitenant FPGA chip setting, where independent customers share the same FPGA accelerator, e.g., in a cloud environment, these monitors open the possibility for remote and even automated large-scale side-channel attacks.

2.2. Formal Model

This subsection presents a formal model of side-channel leakage and corresponding terminology as described in [36,37]. The presented model is used for attack descriptions in the following sections.

Consider a physical device performing a cryptographic operation $\mathrm{E}_k(x)$, depending on a secret (sub)key $k \in \mathcal{K}$, where $\mathcal{K} = \mathbb{B}^m = \{0,1\}^m$, $x \in \mathcal{X}$. The unknown (sub)key is then modeled as a random variable $\mathbf{K} : \Omega \to \mathcal{K}$, the processed data as a random variable $\mathbf{X} : \Omega \to \mathcal{X}$.

Key-dependent state transitions (bit flips) occur inside the device during the execution of E_k. These state transitions are described as word pairs $(v_1, v_2) \in \mathcal{W}$, where $\mathcal{W} = \mathbb{B}^n \times \mathbb{B}^n$, v_1 is the previous state, and v_2 is the new state. Unknown transitions (word pairs) are modeled as a random variable $\mathbf{W} : \Omega \to \mathcal{W}$.

State transitions $\mathbf{W}$ induce side-channel leakage $\mathbf{L}$ on space $\mathcal{L}$, modeled by a side-channel leakage function $\mathsf{L}(\mathbf{W})$. Leakage $\mathbf{L}$ is measured through the noisy physical observable $\mathbf{O}$ on space $\mathcal{O}$.

The model describes a cascade of two channels $\mathbf{W} \to \mathbf{L} \to \mathbf{O}$. This cascade is comprised of a leakage channel $\mathbf{W} \to \mathbf{L}$ through which information on processed words $\mathbf{W}$ leaks in $\mathbf{L}$, and observation channel $\mathbf{L} \to \mathbf{O}$ through which the attacker obtains noisy information on $\mathbf{L}$. The described channels are illustrated in Figure 6.

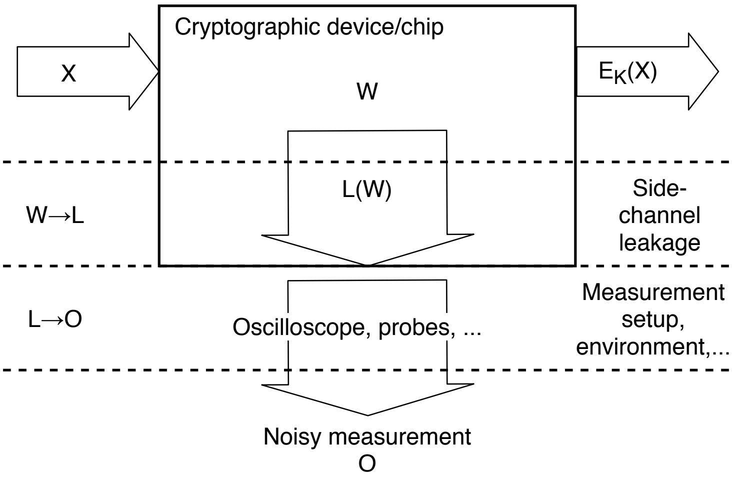

Cryptographic device/chip
X
W
$E_K(X)$
L(W)
W→L
Side-channel leakage
L→O
Oscilloscope, probes, ...
Measurement setup, environment,...
Noisy measurement O

Figure 6. Illustration of channels involved in side-channel analysis.

Observing **O** then means measuring $q \in \mathbb{N}^+$ traces $o_{x_i}(t)$, $i = 1, 2, \ldots, q$, of device's side channel (e.g., power consumption) $\mathbf{O}(t)$, while processing known data x_i. In the case of Rijndael/AES, o_{x_i} might be a trace similar to the one in Figure 3, and x_i might be a corresponding plaintext or ciphertext block.

A side-channel attack is then defined as determining the (sub)key k by reconstructing the words **W** and using information on **L** contained in **O**. For example, the attack may be performed in these steps:

1. The real leakage function L is unknown, so the attacker assumes a hypothetical leakage function $\hat{\mathrm{L}}$ (described in Section 2.3).
2. The attacker makes a guess $\hat{k} \in \mathcal{K}$ on the real (sub)key k.
3. Based on the known data **X**, she computes an intermediate value $f_{\hat{k}}(\mathbf{X})$ within the $\mathrm{E_k}$ computation.
4. The intermediate value implies a guess $\mathbf{W}_{\hat{k}}$, which in turn implies a guess $\hat{\mathbf{L}}_{\hat{k}} = \hat{\mathrm{L}}(\mathbf{W}_{\hat{k}})$.
5. Finally, the attacker checks if the guess $\hat{\mathbf{L}}_{\hat{k}}$ is compatible with the observed **O**.

This attack scenario assumes that the real (sub)key k is fully enumerable in a reasonable time and space. As shown in Sections 3 and 4, side-channel attacks typically target a single subkey, e.g., an octet in the case of Rijndael/AES.

2.3. Leakage Function

Side-channel attacks can be classified into two groups according to the approach of the hypothetical leakage function $\hat{\mathrm{L}}$:

- Non-profiled attacks, in which the attacker only makes use of an explicit leakage function, which is effective for a range of devices (e.g., based on CMOS technology) instead of being tailored for a specific one.
- Profiled attacks, which consist of a profiling step, in which the attacker examines a duplicate of the device under attack and she creates her own leakage approximation. Furthermore, her approximation inherently takes noise contained in **O** into account, making her empirical model more effective. This model is used for the attack, and an explicit leakage function may or may not be used during the process.

In addition to the differences regarding the approach to the hypothetical leakage function, these two types of attacks also assume a differently powerful attacker: for a profiled attack, an exact duplicate of the device under attack is required, whereas it is not for a non-profiled attack.

This subsection briefly introduces widely used explicit leakage functions necessary for non-profiled attacks, which are discussed thereafter in Section 3. Profiled attacks are discussed later in Section 4.

2.3.1. Hamming Distance and Hamming Weight

A Hamming distance leakage function for $v_1, v_2 \in \mathbb{B}^n$ is defined as a number of bit positions at which the words v_1, v_2 differ [18]:

$$\hat{\mathrm{L}}^{\mathrm{HD}}(v_1, v_2) = \mathrm{HD}(v_1, v_2) \in \mathcal{L} = \{0, 1, \ldots, n\}. \tag{1}$$

The function corresponds to the number of bit flips in an n-bit wide register during (v_1, v_2) transition. It is a generally applicable model suitable for attacking CMOS logic. The Hamming distance is equal to the Hamming weight of XOR of the operands: $\mathrm{HD}(v_1, v_2) = \mathrm{HW}(v_1 \oplus v_2)$, where Hamming weight HW is defined as a number of bits in the words that are set to one.

When v_1 is a zero vector, the Hamming distance leakage function reduces to Hamming weight leakage function [38],

$$\hat{\mathrm{L}}^{\mathrm{HW}}(v_2) = \mathrm{HW}(v_2) \in \mathcal{L} = \{0, 1, \ldots, n\}, \tag{2}$$

for $v_2 \in \mathbb{B}^n$. This is often the case when attacking software implementations in microcontrollers [6,39].

Assuming Hamming weight $\mathrm{HW}(x), x \in \mathbb{B}^n$ and uniformly distributed values of n bits in a word $x \in \mathbb{B}^n$, the following properties hold for Hamming weight (and consequently Hamming distance),

$$\mathrm{HW}(x) \sim \mathrm{Binom}(n, \frac{1}{2}), \tag{3}$$

$$\mathbb{E}(\mathrm{HW}(x)) = \frac{n}{2}, \ \mathrm{Var}(\mathrm{HW}(x)) = \frac{n}{4}. \tag{4}$$

Furthermore, the binomial distribution can be satisfactorily approximated by the normal distribution for $p = \frac{1}{2}$ [40], and therefore

$$\mathrm{HW}(x) \approx \mathcal{N}(\frac{n}{2}, \sqrt{\frac{n}{4}}). \tag{5}$$

Generalized Distance Model

Less commonly, different weights may be assigned for 0 → 1 and 1 → 0 transitions, resulting in a generalized distance leakage function. For example, weight 1.5 may be used instead of 1 for the 1 → 0 transition to provide a more effective attack on some platforms [41].

2.3.2. Identity

The identity leakage function [36]

$$\hat{\mathrm{L}}^{\mathrm{id}}(f_{\hat{k}}(x_i)) = f_{\hat{k}}(x_i) \in \mathcal{L}, \tag{6}$$

for $x_i \in \mathcal{X}$, is equal to the targeted intermediate value within the $\mathrm{E_k}(x_i)$ computation. It is the most general leakage function in the sense that it puts no assumptions on the cryptographic device or technology.

3. Non-Profiled Attacks

Non-profiled attacks can be divided into:

- parametric/moment-based attacks, which exploit statistical moments (such as mean or variance). Typical examples include differential power analysis [6] or correlation power analysis [17,18];
- non-parametric/information-theoretic attacks, which exploit the entire underlying statistical distribution. A typical example is mutual information analysis [36]; and
- machine learning-based attacks, namely the deep learning power analysis [42].

These attacks are presented in more depth in this section.

Unless stated otherwise, all of the attack descriptions in this section assume the attacker has already acquired $q \in \mathbb{N}^+$ traces $o_{x_i}(t)$, $i = 1, 2, \ldots, q$, of the device's side channel $\mathbf{O}(t)$ (e.g., power consumption), while processing known data x_i (e.g., plaintext), where bits in x_i are uniformly distributed. Usage of uniform plain text gives a good confidence about uniformity of intermediate values during the computation, because a cipher where properties such as diffusion are expected is typically targeted.

The q measured traces can be modeled as q samples from a multivariate random variable $\mathbf{O}(t)$, where the dimension of the variable corresponds to a number of sampling points within single trace. All the attacks presented in this section, except for the last one, are univariate, i.e., only a single point in time is examined, which is desirable when the sensitive intermediate value manifests itself at a single time instant. In this section, unless stated otherwise, it is assumed that the interesting time instant $t = \tau$ is known and only the single relevant sampling point is considered. The q measured traces are therefore considered a univariate random variable $\mathbf{O}(\tau)$.

In practice, when the time instant is unknown, the attack is performed at every time instant t independently. The final attack evaluation thus typically requires more attention and skill due to a larger false-result chance. Alignment of the traces is required when synchronization of the measurements (e.g., by using a trigger signal) is not possible.

3.1. Differential Power Analysis (DPA)

The differential power analysis [6] attack is performed in these steps:

1. Assume a single bit ($n = 1$) Hamming weight (or distance) leakage function $\hat{\mathsf{L}}$.
2. Enumerate (sub)key guesses $\hat{k} \in \mathcal{K}$.
3. Compute an intermediate value $v_2 = f_{\hat{k}}(x_i), \forall \hat{k}, x_i$ (and the previous state v_1 if Hamming distance is used) and consider only a single bit (e.g., the LSB).
4. For every guess $\hat{k}$, partition measurements o_{x_i} into two groups $O_0^{\hat{k}}, O_1^{\hat{k}}$ according to the leakage function $\hat{\mathsf{L}}$:
$$O_0^{\hat{k}} = \{o_{x_i} \mid \hat{\mathsf{L}}(v_1, f_{\hat{k}}(x_i)) = 0\}, \tag{7}$$
$$O_1^{\hat{k}} = \{o_{x_i} \mid \hat{\mathsf{L}}(v_1, f_{\hat{k}}(x_i)) = 1\}. \tag{8}$$
5. Select the guess $\hat{k}$ for which the groups' $O_0^{\hat{k}}, O_1^{\hat{k}}$ means differ the most.
   - For wrong guesses $\hat{k}$, the traces for which $\mathsf{L} = 0$ and the traces for which $\mathsf{L} = 1$ are theoretically uniformly distributed in both groups.
   - For the right guess $\hat{k}$, the groups $O_0^{\hat{k}}, O_1^{\hat{k}}$ should be distinguishable by their mean value, due to the bias caused by the fixed bit.

In the last step, the original Kocher's DPA [6] selects the guess $\hat{k}$ for which the absolute difference of means between the two groups is greatest. More formally, the hypothesis about equal means may be examined by using Welch's t-test or a similar statistic.

Example: Attacking First Round of Rijndael/AES

1. Assume Hamming weight or identity single-bit leakage (Hamming weight is equivalent to identity for $n = 1$).
2. Select an enumerable key-dependent intermediate value during Rijndael/AES encryption: $f_{\hat{k}}(x_i) := \mathrm{Sbox}(x_i \oplus \hat{k})$. Sbox is an 8-bit bijection, $f_{\hat{k}}(x_i)$ is therefore computable for each byte independently of the other bytes.
3. Enumerate byte subkey guesses $\hat{k} \in \mathcal{K} = \{0, 1, \ldots, 255\}$, compute the intermediate value $v_2 = f_{\hat{k}}(x_i), \forall \hat{k}, x_i$ and choose, e.g., the LSB.
4. Use the Hamming weight of the LSB as leakage function and partition traces o_{x_i} to groups $O_0^{\hat{k}}, O_1^{\hat{k}}$.
5. Select the subkey guess $\hat{k}$ for which the two groups differ the most.

Because DPA builds its hypothesis on a single bit value, its assumptions regarding the leakage are very general. This may be one of the reasons for false results, so-called "ghost peaks". The choice of the bit in $f_{\hat{k}}(x_i)$ has direct impact on the attack success. These facts are a motivation for multi-bit DPA as described further [43,44].

3.2. Multi-Bit DPA and Partitioning Power Analysis (PPA)

Bevan's approach to multi-bit extension of the DPA is performing the original DPA independently for different bits of intermediate value $f_{\hat{k}}(x_i)$ and summing all the independent differences of means [43]. Then the subkey guess with the greatest summed difference is selected. This approach reduces the number of traces necessary as well as the chance of a false result [43].

Messerges's multi-bit extension of the original DPA suggests using n-bit Hamming weight or Hamming distance leakage model [38], therefore utilizing the whole $f_{\hat{k}}(x_i)$ value. This time, two sets $O^{\hat{k}}_{<}, O^{\hat{k}}_{\geq}$ are defined so that

$$O^{\hat{k}}_{<} = \{o_{x_i} \,|\, \hat{\mathsf{L}}(f_{\hat{k}}(x_i)) < \frac{n}{2}\}, \tag{9}$$

$$O^{\hat{k}}_{\geq} = \{o_{x_i} \,|\, \hat{\mathsf{L}}(f_{\hat{k}}(x_i)) \geq \frac{n}{2}\}, \tag{10}$$

and their difference is examined similarly to the original DPA.

Partitioning power analysis [45,46] is a generalization of the multi-bit DPA. Assuming an n-bit $f_{\hat{k}}(x_i)$ intermediate value and a Hamming weight or Hamming distance leakage function, the traces o_{x_i} are partitioned into $(n+1)$ sets $O^{\hat{k}}_0, \ldots O^{\hat{k}}_n$ so that

$$O^{\hat{k}}_j = \{o_{x_i} \,|\, \hat{\mathsf{L}}(f_{\hat{k}}(x_i)) = j\}. \tag{11}$$

The distinguishing statistic (which is a difference of means in the original DPA) is then defined by using weights $a_j \in \mathbb{R}$ as

$$\sum_{j=0}^{n} a_j \cdot \mu_{O^{\hat{k}}_j}, \tag{12}$$

where $\mu_{O^{\hat{k}}_j}$ are means of the aforementioned groups.

The original DPA is a special case of 1-bit PPA where $a_0 = -1, a_1 = 1$. Bevan's 4-bit DPA is a special case of 4-bit PPA where $a_0 = -\frac{1}{8}, a_1 = -\frac{1}{4}, a_2 = 0, a_3 = \frac{1}{4}, a_4 = \frac{1}{8}$. Messerges's n-bit DPA is a special case of n-bit PPA where $a_j = -1$ for $0 \leq j < \frac{n}{2}$, and $a_j = 1$ for $\frac{n}{2} \leq j \leq n$ [46].

3.3. Correlation Power Analysis (CPA)

The correlation power analysis [17,18] attack is performed in these steps:

1. Assume a Hamming weight or Hamming distance leakage function $\hat{\mathsf{L}}$.
2. Enumerate (sub)key guesses $\hat{k} \in \mathcal{K}$.
3. Compute an intermediate value $v_2 = f_{\hat{k}}(x_i), \forall \hat{k}, x_i$ (and the previous state v_1 if using Hamming distance).
4. For every key guess $\hat{k}$, pairs $(o_{x_i}, \hat{\mathsf{L}}(v_1, f_{\hat{k}}(x_i)))$ represent samples from joint distribution $(\mathbf{O}, \hat{\mathbf{L}}_{\hat{k}})$. (In other words, every trace is paired with the predicted Hamming weight/distance).
5. Compute Pearson correlation coefficient $\rho_{\hat{k}} = \frac{\mathrm{Cov}(\mathbf{O}, \hat{\mathbf{L}}_{\hat{k}})}{\sigma_{\mathbf{O}} \sigma_{\hat{\mathbf{L}}_{\hat{\mathbf{k}}}}}$ for every $\hat{k}$.
6. Select the guess $\hat{k}$ for which the value of $|\rho_{\hat{k}}|$ is the highest.

Assuming there is a linear dependence between the predicted leakage and the physical observation, a significant correlation $\rho_{\hat{k}}$ should appear for the right guess $\hat{k}$, while for a wrong guess, the $\rho_{\hat{k}}$ should converge to zero.

Example: Attacking Last Round of Rijndael/AES

1. Assume the architecture illustrated in Figure 7 and a Hamming distance leakage (both v_1, v_2 must be derived). Let $\mathbf{Y} = \text{E}_\text{k}(\mathbf{X})$, i.e., ciphertext.
2. Let $v_2 = y_i$. The previous register state is then $v_1 = f_{\hat{k}}(y_i) = \text{Sbox}^{-1}(\text{Perm}^{-1}(y_i \oplus \hat{k}))$. Both values are once again enumerable for each byte independently of the other bytes due to the fact that the MixColumns operation is not performed in the last round.
3. Enumerate byte subkey guesses $\hat{k} \in \mathcal{K} = \{0, 1, \ldots, 255\}$, and compute the intermediate value $f_{\hat{k}}(y_i), \forall \hat{k}, y_i$.
4. Compute the leakage function $\hat{\text{L}}^{\text{HD}}(f_{\hat{k}}(y_i), y_i), \forall \hat{k}, y_i$.
5. Compute Pearson correlation coefficient $\rho_{\hat{k}} = \frac{\text{Cov}(\mathbf{O}, \hat{\mathbf{L}}_{\hat{k}})}{\sigma_{\mathbf{O}} \sigma_{\hat{\mathbf{L}}_{\hat{\mathbf{k}}}}}$ for every $\hat{k}$.
6. Select the guess $\hat{k}$ for which the value of $|\rho_{\hat{k}}|$ is the highest.

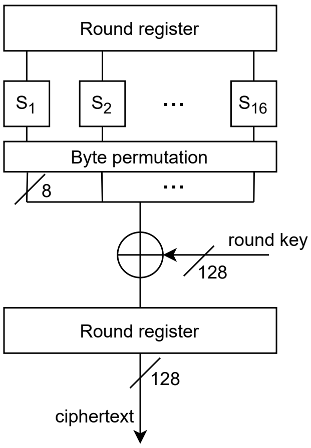


Figure 7. Architecture of Rijndael/AES last round. The scheme is unrolled for illustration purposes only; both "round register" blocks depict the same hardware register.

Unlike previously described DPA attacks, which use a partitioning approach, the CPA attack uses a comparative approach. However, CPA with a single-bit leakage function is equivalent to the original DPA. Interestingly, CPA is equivalent to normalized PPA with weights implicitly given by the distribution of bits in $f_{\hat{k}}(x_i)$. For a uniform distribution of bits in $f_{\hat{k}}(x_i)$, CPA is equivalent to the Bevan's multi-bit DPA [46].

The CPA attack assumes a linear relationship between the predicted leakage and the physical observation. However, this requirement can be relaxed to monotonicity by using the Spearman coefficient instead of the Pearson coefficient [47]. Similar to DPA, CPA exploits statistical moments such as mean or covariance, and therefore requires a "normally" distributed observation channel. The success of the attack largely depends on the quality of the leakage approximation and present noise.

3.4. Mutual Information Analysis (MIA)

The mutual information analysis [36] attack is performed in these steps:

1. Assume an arbitrary leakage function $\hat{\text{L}} \in \mathcal{L}$ (with some restrictions, as explained in Section 3.4.1).
2. Enumerate (sub)key guesses $\hat{k} \in \mathcal{K}$.
3. Compute an intermediate value $v_2 = f_{\hat{k}}(x_i), \forall \hat{k}, x_i$ (and the previous state v_1 if using Hamming distance).

4. Let $L_0, \dots, L_l$ be subsets of $\mathcal{L}$ so that the set $\{L_0, \dots, L_l\}$ is a partitioning of $\mathcal{L}$. The elements $L_j, j = 0, \dots, l$, are called atoms.
5. Associate inputs x_i that leak L_j under key guess $\hat{k}$ to $L_j^{\hat{k}}$:
$$L_j^{\hat{k}} = \{x_i \,|\, \hat{\mathrm{L}}(v_1, f_{\hat{k}}(x_i)) \in L_j\}. \tag{13}$$
Each partition $\{L_0^{\hat{k}}, \dots, L_l^{\hat{k}}\}$ induces a subdivision of measurements o_{x_i}.
6. Define conditional distributions $\{\mathbb{P}_{\mathbf{O}\,|\,L_j^{\hat{k}}}\}_{j=0}^{l}$ by using the subdivision of $\mathbf{O}$, and let $\mathbb{P}_{\mathbf{O}}, \mathbb{P}_{\mathbf{L}_{\hat{k}}}$ be probability distributions of $\mathbf{O}$, $\hat{\mathbf{L}}_{\hat{k}}$.
7. Select $\hat{k}$ with the highest mutual information $\mathbf{I}(\hat{\mathbf{L}}_{\hat{k}}; \mathbf{O})$.

Unlike DPA or CPA, the mutual information analysis exploits mutual information, which is defined as
$$\mathbf{I}(\mathbf{X}; \mathbf{Y}) = D_{KL}(\mathbb{P}_{\mathbf{X},\mathbf{Y}} \,||\, \mathbb{P}_{\mathbf{X}} \otimes \mathbb{P}_{\mathbf{Y}}), \tag{14}$$
where D_{KL} is Kullback–Leibler divergence, i.e., a statistical distance describing probability distribution difference. Mutual information is directly related to entropy H:
$$\mathbf{I}(\mathbf{X}; \mathbf{Y}) = \mathrm{H}(\mathbf{X}) - \mathrm{H}(\mathbf{X}|\mathbf{Y}) = \mathrm{H}(\mathbf{X}) + \mathrm{H}(\mathbf{Y}) - \mathrm{H}(\mathbf{X}, \mathbf{Y}) = \mathbf{I}(\mathbf{Y}; \mathbf{X}), \tag{15}$$
where $\mathrm{H}(\mathbf{X}|\mathbf{Y})$ is conditional entropy, $\mathrm{H}(\mathbf{X}, \mathbf{Y})$ is joint entropy. Mutual information can be intuitively interpreted as the amount of information obtained about **X** by observing **Y** or, in other words, the reduction of uncertainty in **X** obtained by observing **Y**.

The mutual information computation may go as follows:

1. Using measurements o_{x_i} belonging to $L_j^{\hat{k}}$, estimate the conditional distribution $\mathbb{P}_{\mathbf{O}\,|\,L_j^{\hat{k}}}$ and the conditional entropy $\tilde{\mathrm{H}}(\mathbf{O}|\hat{\mathbf{L}}_{\hat{k}} = j)$.
2. Compute the conditional entropy $\tilde{\mathrm{H}}(\mathbf{O}|\hat{\mathbf{L}}_{\hat{k}})$ by using $\{\tilde{\mathrm{H}}(\mathbf{O}|\hat{\mathbf{L}}_{\hat{k}} = j)\}_{j=0}^{l}$.
3. By using all of the measurements o_{x_i}, estimate the distribution $\mathbb{P}_{\mathbf{O}}$ and the entropy $\tilde{\mathrm{H}}(\mathbf{O})$.
4. Compute the mutual information $\tilde{\mathbf{I}}(\hat{\mathbf{L}}_{\hat{k}}; \mathbf{O}) = \tilde{\mathrm{H}}(\mathbf{O}) - \tilde{\mathrm{H}}(\mathbf{O}|\hat{\mathbf{L}}_{\hat{k}})$.

Example: Attacking AES/Rijndael with Minimum Assumptions

1. Assume an identity leakage function $\hat{\mathrm{L}}(f_{\hat{k}})$, e.g., three MSBs of $f_{\hat{k}}(x_i) = \mathrm{Sbox}(x_i \oplus \hat{k})$.
2. Enumerate byte subkey guesses $\hat{k} \in \mathcal{K} = \{0, 1, \dots, 255\}$, and compute the intermediate value $f_{\hat{k}}(x_i), \forall \hat{k}, y_i$.
3. For every subkey guess, associate each o_{x_i} with an atom of $\{L_i^{\hat{k}}\}_{i=0}^{7}$ based on its input's predicted leakage.
4. For every subkey guess, estimate the densities $\mathbb{P}_{\mathbf{O}}$ and $\mathbb{P}_{\mathbf{O}|\hat{\mathbf{L}}_{\hat{k}}}$ and compute the mutual information $\tilde{\mathbf{I}}(\hat{\mathbf{L}}_{\hat{k}}; \mathbf{O})$.
5. Select $\hat{k}$ with the highest mutual information $\tilde{\mathbf{I}}(\hat{\mathbf{L}}_{\hat{k}}; \mathbf{O})$.

Crucial aspect of mutual information analysis is the estimation of probability densities $\mathbb{P}_{\mathbf{O}}$ and $\mathbb{P}_{\mathbf{O}|\hat{\mathbf{L}}_{\hat{k}}}$. Some of the choices include:

- histogram, i.e., a non-parametric discrete estimate;
- kernel density estimate, i.e., a non-parametric continuous estimate; and
- finite mixture model, i.e., a (semi-)parametric continuous estimate.

The quality of the estimate has a direct influence on both attack success and computational complexity [48,49].

A histogram provides a simple and efficient estimate with the most critical parameter being the number of bins [48]. The best estimate would require as many bins as there are values in the domain; however, it would be problematic to get enough values in every bin for it to be statistically significant. Less bins result in less information, but also lower susceptibility to noise. The original MIA [36] proposes using $l+1$ bins, i.e., as many bins as there are atoms in the $\mathcal{L}$ partitioning.

A kernel density estimate provides better attack results than the histogram [48] at the cost of its higher computational complexity. In this case, the kernel and bandwidth are the most critical parameters. Popular kernel choices include Epanechnikov (which is mean-square-error optimal) and Gaussian (for its convenience). Bandwidth has a similar role as bins in histograms, and it can be intuitively seen as a "smoothing parameter." Generally speaking, the attacker aims to select the bandwidth as small as allowed by the data. A "rule-of-thumb" bandwidth estimator can be used alongside the Gaussian kernel [50].

A finite mixtures model assumes the underlying distribution to be a mixture of distributions, whose parameters are estimated, e.g., by using the expectation-maximization algorithm. Typically, a mixture of Gaussians is assumed [51].

Mutual information analysis puts no hard assumptions on the leakage function or the underlying distributions and provides sound results even with a simple identity leakage function. It provides a generic and powerful side-channel distinguisher (although it is less efficient in scenarios well-suited for DPA/CPA) [48,49,52].

3.4.1. Leakage Function in Partitioning Attacks

Mutual information analysis allows for an arbitrary leakage function, giving the attacker a great degree of freedom. Although Hamming weight or Hamming distance leakage functions may be used when it is possible to predict both v_1, v_2, their usage inherently leads to a loss of information. Identity leakage function, which is much more generic, may also be used. If the Hamming weight/distance estimate is possible, identity is shown to be less efficient, but still effective [48].

The leakage function must be selected so that a different $\hat{k}$ must not yield a permutation of $\hat{\mathbf{L}}_{\hat{k}}$. For example, assume that using the identity of Rijndael/AES bijective S-box output. Different $\hat{k}$ then leads to a permutation of $\{L_0^{\hat{k}}, \dots, L_l^{\hat{k}}\}$, which means that the mutual information is constant and independent of $\hat{k}$ [36]. This limitation can be easily overcome, e.g., by choosing only seven least significant bits of the Sbox output, or by using a Hamming weight/distance [36].

This problem does not only pertain to MIA, but to every partitioning attack (all the presented non-profiled attacks except CPA). When $f_{\hat{k}}(x_i)$ is an injective function, an attack using trivial partitioning where each value belongs in its distinct class will always fail [53,54].

3.5. Kolmogorov–Smirnov Analysis (KSA)

The Kolmogorov–Smirnov analysis [48,55] attack is performed in these steps:

1–6. The first six steps are same as for mutual information analysis in Section 3.4. Define the conditional distributions $\{\mathbb{P}_{\mathbf{O}\,|\,L_j^{\hat{k}}}\}_{j=0}^{l}$ and the distribution $\mathbb{P}_{\mathbf{O}}$.

7. For every $\hat{k}$, compute the average Kolmogorov–Smirnov distance between $\mathbb{P}_{\mathbf{O}}$ and $\mathbb{P}_{\mathbf{O}\,|\,L_j^{\hat{k}}}$, optionally further normalized by $\frac{1}{|O_{L_j^{\hat{k}}}|}$, where $|O_{L_j^{\hat{k}}}|$ is a size of the measurements set belonging to atom $L_j^{\hat{k}}$:

$$\mathop{\mathbb{E}}_{j}\left(\frac{1}{|O_{L_j^{\hat{k}}}|} D_{KS}(\mathbb{P}_{\mathbf{O}} \,||\, \mathbb{P}_{\mathbf{O}\,|\,L_j^{\hat{k}}})\right). \tag{16}$$

8. Select the key guess $\hat{k}$ with the largest average KS-distance.

The Kolmogorov–Smirnov distance between $\mathbb{P}_{\mathbf{X}}$ and $\mathbb{P}_{\mathbf{Y}}$ is defined as

$$D_{KS}(\mathbb{P}_{\mathbf{X}} \,||\, \mathbb{P}_{\mathbf{Y}}) = \sup_{x} |F_{\mathbf{X}}(x) - F_{\mathbf{Y}}(x)|, \tag{17}$$

where $F_{\mathbf{X}}$ is a cumulative density function of $\mathbf{X}$. The Kolmogorov–Smirnov analysis is heavily inspired by the mutual information analysis. However, instead of estimating probability density function, the easier-to-obtain cumulative density function is used.

Alternatively, interclass Kolmogorov–Smirnov Analysis (iKSA) [56] distinguishes the key guess using the distance between the conditional distributions:

$$\frac{1}{2}\underset{j,j'}{\mathbb{E}}(D_{KS}(\mathbb{P}_{\mathbf{O}\,|\,L_j^{\hat{k}}}\,||\,\mathbb{P}_{\mathbf{O}\,|\,L_{j'}^{\hat{k}}})). \tag{18}$$

Other choices for comparison of the distributions include Cramér–von Mises criterion or different F-divergences [48].

The Kolmogorov–Smirnov Analysis shares some important characteristics with MIA, as both attacks can be used with an identity leakage function, and therefore without precise knowledge about the implementation and leakage. It can provide better results for weak signals than MIA due to its noise robustness [55].

3.6. Differential Deep Learning Analysis (DDLA)

The differential deep learning analysis [42] attack is performed in these steps:

1. Assume an arbitrary leakage function $\hat{\mathrm{L}} \in \mathcal{L}$.
2. Enumerate (sub)key guesses $\hat{k} \in \mathcal{K}$.
3. Compute an intermediate value $v_2 = f_{\hat{k}}(x_i), \forall \hat{k}, x_i$ (and the previous state v_1 if using the Hamming distance).
4. Create labeled training datasets $\{(o_{x_i}, \hat{\mathrm{L}}(v_1, f_{\hat{k}}(x_i)))\}_{\hat{k}}, \forall \hat{k}, o_{x_i}$. Note that the same limitations as described in Section 3.4.1 apply.
5. Perform deep learning classifier training for every dataset.
6. Select the key guess $\hat{k}$ with the best DL training metrics.

Unlike the previously described attacks in this section, the differential deep learning analysis is typically used in a multivariate fashion, not univariate. In other words, the attack is not performed at a single sampling point or all the sampling points in the trace independently. Instead, the classifier is fed with the multivariate vectors corresponding to the entire encryption.

The differential deep learning analysis is a partitioning attack, like all the previously presented attacks except CPA. A key-dependent partitioning of the data is created and then the distinguishability of the partitions is examined by using the classifier. For the correct key guess, the classifier should be able to learn distinctive features of differently labeled data. When a wrong guess is made, the traces are randomly distributed across labels, and therefore the training metrics should be significantly worse than for the correct guess. Different training metrics are proposed for the final selection of the key, e.g., by using sensitivity analysis [42].

Various deep-learning architectures, such as a multilayer perceptron or a convolutional network may be used for the classifier. Translation-invariance property of convolutional networks can be exploited to attack desynchronized traces [42,57], whereas previously described attacks would require synchronization of the traces during preprocessing, e.g., by using autocorrelation, due to their univariate nature. A distinct disadvantage of using the machine-learning-based blackbox approach is limited explainability of the results [58].

4. Profiled Attacks

Profiled attacks assume the attacker has a fully controlled identical copy of the device under attack at her disposal. She is capable of observing the device's side channels during execution of the identical cryptographic implementation. Moreover, she is able to feed the implementation with arbitrary inputs and keys. Her attack is tailored for a specific device and therefore more effective and efficient than a nonprofiled attack.

A profiled attack consists of two phases:

1. Profiling phase, during which an empirical model of the leakage is created by using the identical copy of the device under attack.
2. Attack phase, during which observations of the device under attack are evaluated by using the previously profiled model.

Unlike non-profiled attacks presented in Section 3, all the presented profiled attacks are multivariate, i.e., full traces $\mathbf{O}(t)$ are considered, where the dimension (t) corresponds to a number of sampling points within a single trace.

4.1. Template Attack (TA)

The template attack [19,59] is performed in these steps:

1. Consider an arbitrary leakage function $\hat{\mathrm{L}} \in \mathcal{L}$.
   Profiling phase
2. Measure a profiling set of traces $o_{(x_i,k_i)}(t)$ using desired (typically, but not necessarily random uniform) inputs (x_i, k_i).
3. Compute an intermediate value $v_2 = f_{k_i}(x_i), \forall k_i, x_i$ (and the previous state v_1 if using Hamming distance).
4. Let $L_0, \dots, L_l$ be subsets of $\mathcal{L}$ so that the set $\{L_0, \dots, L_l\}$ is a partitioning of $\mathcal{L}$. The elements $L_j, j = 0, \dots, l$, are called atoms.
5. Associate measurements $o_{(x_i,k_i)}$ whose inputs (x_i, k_i) leak L_j to O_j:

$$O_j = \{o_{(x_i,k_i)} \,|\, \hat{\mathrm{L}}(v_1, f_{k_i}(x_i)) \in L_j\}. \tag{19}$$

6. Select points of interest t_i within the measurements $o_{(x_i,k_i)}(t)$, e.g., by using sum of differences of average traces of each O_j set or by using principal component analysis. From this moment on, restrict the measurements to these points only.
7. Create empirical models, so-called templates, T_j, e.g., Gaussian probability estimates, characterizing leakage induced by atoms L_j using traces in set O_j.
   Attack phase
8. Measure an attack set of traces $o_{x_i}(t)$ using desired plain texts x_i.
9. Enumerate (sub)key guesses $\hat{k} \in \mathcal{K}$.
10. Compute an intermediate value $v_2 = f_{\hat{k}}(x_i), \forall \hat{k}, x_i$ (and the previous state v_1 if using Hamming distance).
11. Associate measurements whose inputs x_i leak L_j under key guess $\hat{k}$ to $O_j^{\hat{k}}$:

$$O_j^{\hat{k}} = \{o_{x_i} \,|\, \hat{\mathrm{L}}(v_1, f_{\hat{k}}(x_i)) \in L_j\}. \tag{20}$$

12. Compute probabilities $Pr(\mathbf{O} = o_{x_i} \,|\, \mathbf{L} \in L_j), o_{x_i} \in O_j^{\hat{k}}$ that measurements in $O_j^{\hat{k}}$ leak L_j using the templates T_j.
13. Select the key guess $\hat{k}$ with the highest overall probability (product of posterior probabilities) of the predicted leakage.

The template attack provides the attacker with a very powerful and universal tool. The empirical templates T_j are typically multivariate Gaussian models [19]. A creation of the Gaussian template is demonstrated in the following examples.

Example: Attacking Rijndael/AES Using Hamming Weight and Gaussian Templates

1. Assume the Hamming weight leakage function $\hat{\mathrm{L}}$ and intermediate value $f_{\hat{k}}(x_i) = \mathrm{Sbox}(x_i \oplus \hat{k})$.
   Profiling phase
2. Measure a profiling set of traces $o_{(x_i,k_i)}(t)$, by using random uniform inputs (x_i, k_i).
3. Partition measurements $o_{(x_i,k_i)}(t)$ into nine groups according to the Hamming weight of S-box output:

$$O_j = \{o_{(x_i,k_i)}(t) \,|\, \hat{\mathrm{L}}(f_{k_i}(x_i)) = j\}. \tag{21}$$

4. Select sampling points of interest $t_1, \ldots, t_m$ using sum of differences of average traces:
   (a) Compute the average measurement $M_j(t)$ for every group O_j.
   (b) Compute the sum of the absolute pairwise differences of these average power traces: $\sum_{i,j} |M_i(t) - M_j(t)|$ and select the most deviate points, preferably in different clock cycles.

   Reduce the dimensionality of $\mathbf{O}(t)$ and of the average traces $M_j(t)$ to these selected points only.
5. Define noise measurements as

$$N_j = \{n_{(x_i,k_i)}(t) \mid n_{(x_i,k_i)}(t) = o_{(x_i,k_i)}(t) - M_j(t) \wedge o_{(x_i,k_i)}(t) \in O_j\}, \tag{22}$$

   and consider N_j samples from variable $\mathbf{N}_j$.
6. Compute the noise covariance matrices Σ_j between all the points of the interest for every group N_j:

$$\Sigma_j = \begin{pmatrix} \mathrm{Var}(\mathbf{N}_j(t_1)) & \ldots & \mathrm{Cov}(\mathbf{N}_j(t_1), \mathbf{N}_j(t_m)) \\ \vdots & \ddots & \vdots \\ \mathrm{Cov}(\mathbf{N}_j(t_m), \mathbf{N}_j(t_1)) & \ldots & \mathrm{Var}(\mathbf{N}_j(t_m)) \end{pmatrix}. \tag{23}$$

7. $T_j = (M_j, \Sigma_j)$ is a Gaussian template characterizing leakage $\mathsf{L} = j$, i.e., Hamming weight of the S-box output.
   Attack phase
8. Measure/capture an attack set of measurements $o_{x_i}(t)$, by using random uniform plaintexts x_i.
9. Enumerate byte subkey guesses $\hat{k} \in \mathcal{K} = \{0, 1, \ldots, 255\}$ and compute the intermediate value $v_2 = f_{\hat{k}}(x_i), \forall \hat{k}, x_i$.
10. Partition the traces $o_{x_i}(t)$ into nine groups according to the predicted Hamming weight, for every subkey guess:

$$O_j^{\hat{k}} = \{o_{x_i}(t) \mid \hat{\mathsf{L}}(f_{\hat{k}}(x_i)) = j\}. \tag{24}$$

11. For every measurement $o_{x_i}(t)$ in group $O_j^{\hat{k}}$, evaluate the probability of it belonging in the designated group by evaluating the template $T_j = (M_j, \Sigma_j)$:
    (a) Compute hypothetical noise vector $n_{x_i}(t) = o_{x_i}(t) - M_j(t)$.
    (b) Compute the probability of observing $n_{x_i}(t)$ by using the multivariate Gaussian probability distribution:

$$p_j(n_{x_i}(t)) = \frac{1}{\sqrt{(2\pi)^N |\Sigma_j|}} \exp(-\frac{1}{2}\, n_{x_i}(t)^\top\, \Sigma_j^{-1}\, n_{x_i}(t)\,), \tag{25}$$

    where N is number of points of the interest, $|\Sigma_j|$ is the determinant of Σ_j, and Σ_j^{-1} is its inversion.
12. Select the key guess $\hat{k}$ with maximum overall probability of the measurements being partitioned in the correct groups.

Example: Attacking Rijndael/AES Using a Single Measurement

1. Assume identity leakage function $\hat{\mathsf{L}}$ and intermediate value $f_{\hat{k}}(x_i) = \hat{k}$. Assume an atom L_j for every subkey value, i.e., 256 atoms.
   Profiling phase
2. Measure a large amount of traces by using uniform plain texts and keys and create 256 Gaussian templates.
   Attack phase

3. Measure a trace using a uniform plaintext and evaluate it against all the templates. Select the key guess with the highest probability.

Computing the probabilities as described above may lead to numerical instabilities, which can be solved by using logarithms of probabilities instead [28].

The creation of the model (i.e., the templates) requires a large amount of measurements in comparison to the actual attack. Efficient and effective templates may require further evaluation due to potential overfitting; templates too specific for the attacker's copy might not work on the device under attack [19].

Various ways to improve the efficiency of the template attack are described in [60], such as methods for the dimensionality reduction/selection of points of the interest, usage of pooled covariance matrices, combining multiple traces, etc.

Many use cases and scenarios are possible by using the extend-and-prune approach, i.e., starting with small parts of information and increasingly extending the attack. Thanks to the profiling phase, the attack phase is very effective and efficient.

4.2. Machine Learning-Based Attacks

Machine learning (ML) algorithms are algorithms that learn to solve a problem without being explicitly programmed to do so. In this context, "to learn" can be perceived as "to build an empirical model using training data", whereas "to solve" can be perceived as "to evaluate real data using the model". A profiled side-channel attack as described in this section can be reduced to a classifying task, which is thoroughly studied in the context of supervised machine learning [61].

Profiled machine learning-based attacks are typically performed in a similar fashion as the template attack, and can often be classified as one. They also share many of its advantages and disadvantages. The crucial difference is in the choice of the empirical model; instead of Gaussian, a machine learning-based classifier is used.

A support vector machine (SVM) is a machine learning algorithm commonly used for classification, based on creating an optimal hyperplane between different classes. It was shown to be more efficient than a Gaussian template attack in some aspects [62–64], performing better on noisy measurements and requiring a smaller profiling set. However, selection of the algorithm parameters, such as the kernel function, may have a significant impact on its performance [62]. Both binary and multi-class SVM classifiers were successfully used to attack Rijndael's S-box output [65]. Other common classifier choices are decision trees, random forests [64,66], and others.

Neural network-based deep learning classifiers are a popular choice in side-channel security [67,68]. The non-profiled variant of the attack is presented in Section 3.6. Both multilayer perceptron [69,70] and convolutional neural network [57,71] architectures are suitable for a profiling attack. Whereas a multilayer perceptron classifier must be fed with aligned power traces, the location and scale invariant convolutional neural network can extract the features itself and therefore it is capable of processing misaligned or jittered measurements without prior preprocessing [57]. It is capable of exploiting both univariate and multivariate leakage, as well as utilizing both Hamming weight/distance or identity training labels [72].

Hyperparameters of the neural network model include the network architecture (number of nodes in a layer, number of layers, activation function) and learning parameters (number of epochs, batch size, optimization algorithm, learning rate). Unfortunately, there does not seem to be the best model for every scenario ("no free lunch" theorem). Finding a suitable model is a nontrivial task; however, it is crucial for a successful attack. Class imbalance may present a significant obstacle [73], especially when Hamming weight is used (e.g., only a single word value $x \in \mathbb{B}^n$ leads to $\mathrm{HW} = 0$, in contrast to $\mathrm{HW} = \frac{n}{2}$). Even though the neural network can be fed the whole unprocessed and even misaligned traces, it holds that the higher is the dimension of the data, the higher is the attack complexity and the larger training sets are required [66]. Similarly to the template attack, both underfitting and overfitting the of model during the learning phase may lead to an unsuccessful attack;

the former cannot generalize the observations, whereas the latter learns non-relevant details and noise [72].

5. Side-Channel Attack-Related Metrics

Several metrics related to side-channel attacks are presented in this section. Experimental metric success rate and Guessing entropy are presented in Section 5.1. Theoretical metrics' confusion coefficients and distinguishing margins, and their relationship to differential cryptoanalysis are presented in Section 5.2.

Let $\hat{k} \in \mathcal{K}$ be a (sub)key guess during an attack and let $k^* \in \mathcal{K}$ be the real (secret) (sub)key. Define a distinguisher $\mathcal{D}^{\hat{k}}(o_{x_1}, \ldots, o_{x_q}; x_1, \ldots, x_q)$ as an absolute value of the statistic that is used to distinguish the correct key during the attack. For example, a difference of means or t-value in case of DPA (Section 3.1), a correlation coefficient in case of CPA (Section 3.3), a mutual information in case of MIA (Section 3.4), a Kolmogorov–Smirnov distance in case of KSA (Section 3.5), probabilities in case of DDLA (Section 3.6), and TA (Section 4.1). Assume that the higher the value of $\mathcal{D}_{\hat{k}}$ is, the higher is the probability of the correct key $\hat{k}$.

5.1. Success Rate and Guessing Entropy

Success rate [74] and guessing entropy [75,76] are experimental metrics allowing a comparison of different attacks on the same implementation. A simplified definition of these metrics is presented in this subsection, omitting two parameters: time complexity τ and memory complexity m [74].

Assume all the key guesses $\hat{k}$ are sorted according to the value of $\mathcal{D}^{\hat{k}}$ in descending order, i.e., the most probable candidate is positioned first, the second most probable candidate is positioned second, etc. Let $\#(\hat{k})$ be a position of the guess $\hat{k}$. Success rate is then defined as the probability of the correct key guess k^* being on the first position:

$$\mathrm{SR}(q) = \Pr(\#(k^*) = 1), \tag{26}$$

where q is the number of measurements available. In other words, it is the probability of the attack revealing the correct key. The n-th order Success rate is defined as

$$\mathrm{Succ}^n(q) = \Pr(\#(k^*) \leq n). \tag{27}$$

Guessing entropy is a related metric defined as the expected position of the correct key within the previously mentioned sorted guesses:

$$\mathrm{GE}(q) = \mathbb{E}(\#(k^*)), \tag{28}$$

where q is a number of measurements available. Whereas success rate characterizes the probability of the attack being successful, guessing entropy characterizes the amount of the remaining work of the attacker when the attack fails to reveal the correct key.

In order for the values of success rate or guessing entropy to be trustworthy, a large number of independent experiments must be performed [74].

5.2. Confusion Coefficient and Distinguishing Margin

Consider a single-bit intermediate value $f_k(x)$ (as in a DPA attack). Let k^* be the correct key and $k \in \mathcal{K}$ be any key hypothesis. The confusion coefficient $\kappa(k^*, k)$ is then defined as a probability of the bit $f_k(x)$ having a different value given two different keys [77]:

$$\kappa(k^*, k) = \Pr(f_{k^*}(x) \neq f_k(x)). \tag{29}$$

It reaches minimum when the two keys are the same: $\kappa(k,k) = 0$, and maximum $\kappa(k^*, k) = 1$ iff $\exists k \neq k^*, \forall x : f_{k^*}(x) = \overline{f_k(x)}$. Assuming $f_k(x) = \mathrm{S}(x \oplus k)$, the confusion coefficient is directly linked to cryptoanalytical metrics of the boolean S-box $S : \mathbb{B}^n \rightarrow \mathbb{B}^m$, namely to its differential uniformity Δ_S:

$$\Delta_S = \max_{a \in \mathbb{B}^m, k \in \mathbb{B}^n} |\{x \in \mathbb{B}^n \mid S(x) \oplus S(x \oplus k) = a\}|. \tag{30}$$

Considering $m = 1$ ($f : \mathbb{B}^n \rightarrow \mathbb{B}^m$ being equivalent to $\{f_i : \mathbb{B}^n \rightarrow \mathbb{B}\}_{i=1}^m$) [78]:

$$2^{-n}\Delta_S - \frac{1}{2} = \max_{k \neq k^*} |\kappa(k^*, k) - \frac{1}{2}|. \tag{31}$$

Let b be one bit of a sensitive variable $f_k(x)$ for a perfectly secret encryption algorithm. Then b is equiprobable, i.e., $\Pr(b = 1) = \Pr(b = 0) = \frac{1}{2}$ [79]. Assume that the bit b is the one under attack. In such a case, the DPA/CPA and KSA/iKSA distinguishers can be rewritten in following closed-form expressions [78]:

$$\mathcal{D}^k_{\text{CPA}} = \mathcal{D}^k_{\text{DPA}} = \frac{2}{\sqrt{1 + 1/\text{SNR}}} \cdot |\kappa(k^*, k) - \frac{1}{2}|, \tag{32}$$

$$\mathcal{D}^k_{\text{KSA}} = 2\mathcal{D}^k_{\text{iKSA}} = (2\Phi(\sqrt{\text{SNR}}) - 1) \cdot |\kappa(k^*, k) - \frac{1}{2}|, \tag{33}$$

where $\Phi(x)$ is a cumulative distribution function of the standard noise $\mathcal{N}(0, 1)$. These equations describe the relationship between the distinguisher and noise. Notice that for large noise, the first multiplicand in both equations tends to zero.

Let the distinguishing margin (distance to the nearest rival) be the distance between the correct key k^* distinguisher value $\mathcal{D}^{k^*}$ and the maximum incorrect key guess $\hat{k}$ distinguisher value [80]:

$$\text{DM} = \mathcal{D}^{k^*} - \max\{\mathcal{D}^{\hat{k}} | \hat{k} \neq k^*\}. \tag{34}$$

The distinguishing margin characterizes the ability of the attacker to distinguish the correct key, i.e., her ability to make the attack succeed. For the Kolmogorov–Smirnov distinguisher, it can be explicitly expressed in terms of confusion coefficient, and therefore differential uniformity [78]:

$$\text{DM}_{\text{KSA}} = \lambda\,(\frac{1}{2} - \max_{k \neq k^*} |\kappa(k^*, k) - \frac{1}{2}|) = \lambda(1 - 2^{-n}\Delta_S). \tag{35}$$

This equation demonstrates that the attack becomes easier as the distance between κ and $\frac{1}{2}$ becomes smaller. It also provides a direct link between S-box properties, i.e., its differential uniformity, and its susceptibility to side-channel attacks; the harder the differential cryptoanalysis, the easier the side-channel analysis.

6. Countermeasures against Attacks

Countermeasures against side-channel attacks can be categorized into two basic groups [28]:

- Hiding, whose main objective is to "hide" the sensitive variable leakage, ideally to entirely remove the data dependency of the $\mathbf{L} \rightarrow \mathbf{O}$ channel. Hiding countermeasures generally focuses on the signal-to-noise ratio (recall Equation (32) and Equation (33)). Hiding countermeasures can sometimes be further classified as (1) hiding in amplitude, and (2) hiding in time. Shuffling, which randomizes the algorithm flow, is sometimes considered a separate category, as it is implemented on the algorithm level; however, its effect is similar to that of hiding.
- Masking randomizes the processed data $\mathbf{W}$ while still providing correct results, therefore making it hard (ideally impossible) for the attacker to predict any intermediate values. The aim is to make the $\mathbf{W} \rightarrow \mathbf{L}$ channel appear random, ideally to remove the data dependency altogether. Unlike hiding countermeasures, masking typically re-

quires a source of fresh randomness. The security of the masking schemes is therefore dependent on the used random generator.

Correctly employing the presented countermeasures does not result in an absolutely secure implementation. The objective is to make the attack infeasible in a real-world scenario, typically by increasing either the number of measurements necessary or the computational cost of the attack over a limit of resources practically available to the attacker. Similar to the classic cryptoanalysis, this limit lowers in time as the available computational capacity increases. Real-world attack examples [16] in 2021 show that protection against extreme numbers of traces (hundreds of millions or more) is necessary. Attacks on protected implementations are presented later at Section 7.

In implementation terms, the countermeasures can further be categorized into three groups [81]:

- Secure logic styles, which incorporate custom logic gate libraries, designed to minimize the data dependency. These countermeasures inherently introduce a large overhead, and their implementation is typically expensive. Countermeasures in this category are generally "hiding in amplitude".
- Additional modules, which are incorporated into the cryptographic design. The basic advantage of these modules is their universal applicability and lower cost compared to that of the secure logic style. Countermeasures in this category are generally hiding countermeasures, either in amplitude or in time.
- Cryptographic module modifications, which aim to alter the encryption itself. While offering a lower overhead than the previously mentioned categories, they are often algorithm-specific and may present unforeseen weaknesses. This category contains, most importantly, masking. However, some hiding countermeasures fall within the category as well.

Additionally, some types of cryptography were believed to be resistant to SCA by their nature. For example, chaos cryptography [82,83] was expected to be hard to attack because of its unpredictable behaviour [84]. Nevertheless, these beliefs were disproven, e.g., by a study showing that the chaos-based S-Boxes are similarly vulnerable to SCA as the AES ones [85]. Similarly, ARX-based cryptography [86,87] was expected to be more resistant to SCA, as there is no highly nonlinear element (S-Box) whereas later research showed the opposite [88,89].

6.1. Secure Logic Styles

The logic gates are typically designed in static CMOS technology, as illustrated in Figure 1). Various logic styles aim to compensate for their data-dependent behavior, often combining concepts of differential and dynamic logic [24]. Differential logic uses complementary signals (e.g., A and $\overline{A}$) at gate input and output, implementing differential pull-down networks to equalize the power consumption of different transitions. Two-phase dynamic logic introduces clock-driven pre-charge, where the load capacitance is artificially charged. Combination of the two styles is often referred to as dynamic differential logic or dual-rail pre-charge logic [81].

Various countermeasures based on secure logic styles were proposed in the literature. Sense amplifier-based logic (SABL) [12] is an example of dynamic differential logic. It aims to ensure constant consumption of all input and output transitions at the cost of complicated design (custom logic gates, "domino" logic), under the assumption that all interconnections and capacitances are symmetrical (balanced). Simple dynamic differential logic (SDDL) [13] uses ordinary CMOS gates, implementing the differential logic by using De Morgan's law and the pre-charge using AND gates. Wave dynamic differential logic (WDDL) [13] extends the SDDL by limiting the used logic to AND and OR gates, thanks to which a "precharge wave" is introduced to reduce overhead. Furthermore, WDDL promises to be glitch-resistant, as opposed to SDDL, where data-dependent hazards may compromise the countermeasure. Similarly to sense amplifier-based logic, SDDL and WDDL require symmetrical interconnections and capacitances. Other extensions of dynamic differential logic are available in the literature [90–92].

Adiabatic logic [93] was originally designed for low-power applications with the aim of reusing energy efficiently instead of it being discharged. It is powered by a clock-controlled power source, typically trapezoidal. Various logic styles which make use of adiabatic logic were proposed with the aim of hiding leakage [94,95].

Other logic style examples include randomized multitopology logic [96] or use of asynchronous logic styles [97–99].

6.2. Additional Modules

Unlike countermeasures based on secure logic styles, the countermeasures presented in this subsection put no (or minor) assumptions on the cryptographic module.

Measured SNR can be effectively lowered by employing noise generators inside the cryptographic device. Different primitives can be utilized to build the generator, including shift registers, block RAMs, switch boxes [14] or ring oscillators [100]. The design of the actual cryptographic primitive can be used for correlated noise generation [101,102].

Current sense-shunt loop-back can be used for active current flattening to hide the leakage [103,104]. Similarly, the current can be randomized by using a variable current source [105]. Decoupling-based countermeasures are based on powering the cryptographic core from internal capacitors [106–108] in order to hide the instantaneous consumption.

Hiding in time is typically achieved by employing a specific clock signal or by altering the cryptographic algorithm. Isolated clock network can be used to deny the attacker the possibility of using the global clock network for synchronization [109]. The clock signal can be randomized [14]. Alternatively, dummy operations and data can be inserted randomly during the computation [110–112]. Partial dynamic reconfiguration can be used to shuffle the algorithm execution to hide the leakage in both time and amplitude [15].

6.3. Masking

Unlike previously presented countermeasures, masking [8] requires a detailed knowledge of the cryptographic algorithm. Its implementation is modified so that all intermediate values are masked by using a random value, making it difficult for an attacker to predict the leakage and therefore mount an attack. A relevant function (group law) is chosen for the masking according to the values domain, e.g., an exclusive or (XOR) in case of Galois field—then the masking is called boolean. When the sensitive value is multiplied by a mask, the masking is called multiplicative. Unlike boolean masking, multiplicative masking is inherently unable to mask a zero value [113].

Unless stated otherwise, boolean masking is considered in this subsection. The sensitive value x is split into $d+1$ shares x_i, where

$$x = \bigoplus_{i=0}^{d} x_i. \tag{36}$$

The splitting is done by generating d uniform random masks $x_1, \dots, x_d$ and by putting $x_0 = x \oplus x_1 \oplus \dots \oplus x_d$. The number of masks d is then called a masking order. Implementation secured with d-order masking should ideally be secure against attacks up to d-th order [8,114] (as defined further in Section 7). However, the desired security level is often not reached in practice [115,116] due to unforeseen imperfections.

A cryptographic algorithm typically consists of several linear and nonlinear operations, some of which must be altered to function properly when the variable is split. Masking of a linear operation f is trivial because all of the shares can be processed independently:

$$f(x) = f(x_0 \oplus \dots \oplus x_d) = f(x_0) \oplus \dots \oplus f(x_d). \tag{37}$$

There are different approaches to dealing with the nonlinear operations. Considering substitution-permutation network-based ciphers, substitution boxes (S-boxes) are typically the nonlinear operations.

Pre-computed masked S-boxes were originally proposed for first-order masked software implementations [9] and later adapted for hardware [14,117]. The concept is further described in Section 6.3.1.

A more efficient approach suitable for hardware Rijndael/AES implementations splits the S-box into an inversion and an affine operation and masks the inversion by using a multiplicative mask [118]. Even lower overhead can be achieved when the S-box computation is performed in a composite field [119–121]. However, these hardware masking schemes were later shown to be vulnerable against first-order side-channel attacks due to data-dependent glitches occuring during the S-box computation [115]. An example of glitch-induced leakage in a masked AND gate is shown in [122, section 4.1].

This issue is solved by glitch-resistant masking schemes, such as domain-oriented masking [11] or threshold implementation [10,122], which is further described in Section 6.3.2. Lower overhead of these schemes can be once again achieved by using a composite field S-box computation [123,124].

6.3.1. Pre-Computed Masked Substitution Boxes

Pre-computed masked S-boxes were originally proposed for first-order masked software implementations [9] and later utilized in hardware (FPGA) by using block RAM [14] or more efficient CFGLUT [117] primitives. In the following paragraphs, the concept will be described as used for PRESENT [125] encryption. PRESENT is a lightweight substitution-permutation network-based cipher with a block size of 64 bits and possible key sizes of 80 or 128 bits. Each round consists of a round key addition (XOR), a non-linear substitution layer (4-bit S-boxes applied 16 times in parallel), and a linear permutation layer. After 31 rounds, the 32nd round key is finally added to produce the ciphertext.

Assuming the PRESENT encryption algorithm accepts plain text pt masked by XORing a random mask m:

$$state' := pt \oplus m, \tag{38}$$

where $state'$ is the masked cipher state, three round operations/layers must be taken into account and altered appropriately so that equation

$$state = state' \oplus m \tag{39}$$

holds, allowing the ciphertext to be obtained using $state'$.

The first layer, round key addition, i.e., XOR, is a commutative and associative operation:

$$state' \oplus rk = (state \oplus rk) \oplus m. \tag{40}$$

Therefore, addition of the round key rk does not require any further alteration since the output of the layer is already equal to the valid cipher state masked by m.

The last layer, the permutation layer, is a linear transformation P, used to permute bits of the cipher state. The output of the layer is therefore equal to the valid cipher state masked by a permuted mask:

$$P(state') = P(state) \oplus P(m), \tag{41}$$

which means the mask that would need to be subtracted to obtain the valid cipher state changes to $P(m)$.

The middle layer is a non-linear substitution layer S. The validity of the output is assured by altering the substitution look-up table into a masked substitution layer S'

$$S'(state') := S(state' \oplus m) \oplus P^{-1}(m), \tag{42}$$

which realizes the original substitution upon masked input value and outputs the substitution result masked by m processed with inverse permutation P^{-1}. This approach countermands the mask alteration performed by the Permutation layer, since

$$P(state \oplus P^{-1}(m)) = P(state) \oplus P(P^{-1}(m)). \tag{43}$$

Therefore, S' is the only alteration which must be performed for Equation (39) to hold.

In this example, the mask m is used through entire encryption, allowing usage of a single precomputed substitution layer. However, special care must be taken when the round state is written to a CMOS register holding the previous round state. Assuming the Hamming distance leakage model, the mask m would get subtracted:

$$\mathrm{HD}(x \oplus m, y \oplus m) = \mathrm{HW}(x \oplus y \oplus m \oplus m) = \mathrm{HD}(x, y). \tag{44}$$

One possible solution to this problem is a combination with a register precharge hiding countermeasure [117], where the working register is doubled and the encryption context is interleaved with random data.

6.3.2. Threshold Implementation

Threshold implementation [10,122] is a glitch-resistant masking scheme suitable for both hardware and software implementations [126].

According to the selected masking order d, the input is first split into $d+1$ shares as described in Equation (36). Linear operations during computation are performed on each share independently as described in Equation (37). Each non-linear operation f is split into $d+1$ shared functions $f_0, \dots, f_d$ over which the following properties are defined: correctness, non-completeness, and uniformity.

Correctness property assures that the correct result of f can be obtained after the computation:

$$\bigoplus_{i=0}^{d} f_i(x_0, \dots, x_d) = f(\bigoplus_{i=0}^{d} x_i). \tag{45}$$

Non-completeness property requires each function f_i to be independent of at least one share of each input variable, e.g.,

$$\begin{gathered} f_0(x_1, x_2, \dots, x_d), \\ f_1(x_0, x_2, \dots, x_d), \\ \dots \\ f_d(x_0, x_1, \dots, x_{d-1}). \end{gathered} \tag{46}$$

For the masking scheme to protect against higher-order attacks, the property must be extended to the d-th order non-completeness [127]: any combination of up to d shared functions f_i must be independent of at least one share of each input variable.

Similarly, as the inputs x_i are uniformly shared, which is assured by generating uniform masks, the uniformity property requires the output of the shared functions f_i to be uniformly shared as well. Unlike previous properties which can be explicitly validated, uniformity is typically checked by using an exhaustive enumeration and conditional probability examination. Since uniformity is often hard to achieve directly, remasking with a fresh randomness may be necessary after the non-linear stage [128].

To eliminate the propagation of glitches, assure non-completeness when consecutive non-linear operations are considered, and possibly split a single non-linear operation, pipeline registers must be used between the stages. At least $d+1$ shares are required to implement a function of algebraic degree d (e.g., Rijndael/AES S-box has algebraic degree 7). Splitting the non-linear stage (e.g., decomposing the function or computing the S-box in a composite field) may result in functions of a smaller algebraic degree; therefore, a smaller number of shares and lower overall overhead [123,129].

7. Attacks on Protected Implementations

Approaches to attacking protected implementations are presented in this section. Attacks on hiding countermeasures are summarized in Section 7.1 and attacks on masking are explained in Section 7.2.

Most of the presented techniques are typically performed as a pre-processing step before mounting an attack. Moment-based attacks (recall Section 3) on masking can be computed in an online and parallel fashion [130], sparing computing resources. Machine learning-based attacks may even be mounted on protected implementations in a same fashion as when attacking unprotected implementations [42].

7.1. Attacks on Hiding

Different approaches were proposed to deal with hiding in time. Simple time shifts can be overcome by using autocorrelation, i.e., a correlation of a signal with a delayed copy of itself. More generally, a pattern near the sensitive operation can be used with matching techniques known from digital signal processing [28] to identify the time shift and align traces appropriately. These methods are also useful when there is no dependable synchronization signal for measurements (trigger).

When the leakage is spread in time in a more chaotic manner, e.g., by clock jitter, a sliding window attack may be used, where a finite number of (consecutive or not) time samples is summed/integrated to a single value [110]. When attacking implementations with a hiding in time countermeasure in place, this attack results in better signal-to-noise ratio. However, because the noise is integrated as well, it is still less efficient than a direct attack on an unprotected implementation [110].

Another example of an attack on hiding in time is the use of elastic alignment attack [131], which utilizes dynamic time warping techniques [132] to create well-aligned traces.

Machine-learning based attacks were shown to successfully break through hiding in time countermeasures, most importantly convolutional neural networks, thanks to their location-scale invariance properties [57].

Similarly, different techniques were proposed for attacking hiding in amplitude countermeasures. Differential logic (such as WDDL) without proper place and route constraints can be successfully attacked by using electromagnetic analysis [133], as the attacker is able to measure leakage from only a small part the chip. Various approaches to filter out excessive noise (such as that created by noise generators) were also proposed [134–137], e.g., based on wavelet transform.

7.2. Attacks on Masking

Consider a moment-based non-profiled attack (e.g., DPA or CPA). With a masking countermeasure in place, the intermediate sensitive variable $f_k(x_i)$ is split into d shares (recall Equation (36)). Side-channel attack targeting this intermediate value, therefore, must consider d mutually independent leakages. Such an attack is then referred to as a higher-order, or d-th order, attack [6]. The d leakages may manifest themselves at different times, resulting in a multivariate higher-order attack. Similarly, when the leakages manifest at the same time, a univariate higher-order attack is mounted.

The combining function C is used to combine multiple key-independent noisy distributions to produce a single key-dependent distribution, which is then exploited by the attack. Two different combining functions are presented in this subsection: absolute difference combining and product combining. The centralized absolute difference combining between two time samples $o_{x_i}(t_1), o_{x_i}(t_2)$ with mean values $\mu_{o(t_1)}, \mu_{o(t_2)}$ is defined as [138]

$$\mathsf{C}(o_{x_i}(t_1), o_{x_i}(t_2)) = |\,(o_{x_i}(t_1) - \mu_{o(t_1)}) - (o_{x_i}(t_2) - \mu_{o(t_2)})\,|, \tag{47}$$

and the centralized product combining between arbitrary number of samples is defined as [8]

$$\mathsf{C}(o_{x_i}(t_1), \ldots, o_{x_i}(t_d)) = \prod_{k \in \{1,\ldots,d\}} (o_{x_i}(t_k) - \mu_{o(t_k)}). \tag{48}$$

The centralization, i.e., subtracting the mean, normalizes the Gaussian noise and minimizes bias during the combining [139]. Optimal leakage function $\hat{\mathsf{L}}$ to use with combined leakage is then combining function specific.

Assume attacking a first-order masking scheme, i.e., a second-order attack, using Hamming weight model. Denote the targeted intermediate variable $z = f_k(x_i) \in \mathbb{B}^n$, and let it be split in two shares $s_0, s_1 \in \mathbb{B}^n$:

$$s_0 = z \oplus m, \quad s_1 = m, \tag{49}$$

where $m \in \mathbb{B}^n$ is a uniform independent mask. Model the observed leakage during processing of each share as

$$\mathsf{O}_0 = \delta_0 + \mathrm{HW}(s_0) + B_0, \quad \mathsf{O}_1 = \delta_1 + \mathrm{HW}(s_1) + B_1, \tag{50}$$

where δ_0, δ_1 are constant parts of the leakage, and $B_0, B_1 \sim \mathcal{N}(0, \sigma)$ are zero-centered Gaussian noise. For Hamming distance model, assume usage of $z' = z \oplus v_1^0 \oplus v_1^1$, $m' = m \oplus v_1^1$ instead of z, m, where v_1^0, v_1^1 are previous states of z, m, respectively. The optimal leakage prediction function $\hat{\mathsf{L}}$ for the centered absolute difference combining is then [139]

$$2^{1-\mathrm{HW}(z)}\,\mathrm{HW}(z)\binom{\mathrm{HW}(z)-1}{\lfloor\frac{\mathrm{HW}(z)}{2}\rfloor}, \tag{51}$$

i.e., a non-affine function of the Hamming weight. The optimal leakage prediction function for the centered product combining is [139]

$$-\frac{1}{2}\mathrm{HW}(z) + \frac{n^2+n}{4} + \frac{n}{2}(\delta_1+\delta_2) + \delta_1\delta_2, \tag{52}$$

i.e., a linear function of the Hamming weight. The centered product combining is therefore well-suited for a correlation attack (CPA), where simple $\mathrm{HW}(z)$ predictions will show up as a negative correlation. Assuming very noisy observations, the centered product combining function leads to a more efficient attack than the absolute difference combining [139]. Moreover, a higher-order attack using centered product combining can be computed in a one-pass and parallel fashion [130], instead of pre-processing the data.

As mentioned earlier, the higher-order attack can be either univariate or multivariate. Multivariate attack is usually used when attacking masked software implementations [138] and may require a prior points-of-the-interest analysis (recall the template attack in Section 4.1); otherwise, it may become very expensive in terms of both computational power and memory. Univariate attack is suitable when the implementation leaks information about all of the shares in the same time instant [140], which is usually the case for hardware implementations. In such cases, the time sample will be combined with itself, e.g., $(o_{x_i} - \mu_o)^2$ assuming a second-order attack and product combining. Notice that the combining result is equal to the second central moment.

Combining the samples also results in an amplification of the noise [8]. The amount of side-channel information necessary for a successful attack grows exponentially with the masking order. Assuming variance of a single observation is σ^2, variance of k combined samples is approximately $(\sigma^2)^k$, and to distinguish between two distributions with different means and $(\sigma^2)^k$ variance, approximately $(2\sigma^2)^k$ samples are necessary [8]. Therefore a sufficient noise level is necessary for masking countermeasures to be secure [141].

Mutual information analysis is, unlike DPA or CPA, "naturally higher-order" in the sense of examining the entire underlying distribution instead of statistical moments [49]. For a multivariate attack, there are different methods of combining multiple time samples: (1) considering them a d-dimensional vector, (2) computing multivariate mutual information, or (3) computing total correlation. Multi-dimensional probability density function must then be estimated.

Machine learning-based attacks were also shown capable of exploiting higher-order leakage [42,142] and successfully breaking masked implementations. Compared with CPA, a machine-learning based attack might not require any alterations, and it may be performed on both unprotected and protected implementations with no adjustments [42].

8. Leakage Assessment

Leakage assessment methodology examines whether the implementation leaks information. A naïve technique of testing vulnerability against side-channel attacks would be mounting all the known attacks. Leakage-assessment methodologies offer a more general and less computationally and time-demanding approach. Similarly to the non-profiled attacks, the methods presented in this section are typically based on partitioning the measurements and examining their distinguishability. Contrary to the attack scenario, the evaluator in this scenario has full control over the implementation. The described tests can be categorized as either specific or non-specific tests [143,144].

The specific tests [143] typically evaluate measurements of random uniform plain text encryptions with a fixed key. They partition the measurements into two or more groups according to a selected intermediate value and leakage function. For example, assuming single-bit Hamming weight leakage (similarly to DPA in Section 3.1), the measurements are partitioned into two groups according to the value of a bit of S-box output. Considering Rijndael/AES, this intermediate value provides 128 different partitionings (one for each bit of the cipher context). Other options for the intermediate value include round output or XOR of round input and output. Distinguishability of the groups then suggests the possibility of presence of leakage exploitable by targeting the selected intermediate value.

The non-specific tests [143,144] do not target a specific intermediate value. Instead, e.g., distinguishability between two groups containing measurements of an encryption of either random uniform plain text, or of pre-selected fixed plaintext, is tested. Such tests are referred to as random vs. fixed tests. The other choice is a fixed vs. fixed test. In such tests, both groups must be measured in a randomly interleaved fashion during a single evaluation to prevent false results, e.g., due to environmental noise or varying device temperature [144]. Distinguishability of the two groups once again suggests information leakage. Non-specific tests are more sensitive and general than specific tests, and they provide only limited information about the leakage origin.

Various statistical tools can be used to test the distinguishability of the groups. Methodologies based on Welch's t-test and Pearson's χ^2 test are described in Sections 8.1 and 8.2. A deep learning-based approach is described in Section 8.3.

The measurement setup plays a crucial role in leakage evaluation. Test equipment (e.g., oscilloscope) with sufficient bandwidth, sampling rate and resolution must be used [143], as these and other parameters have a direct impact on potential attack success [30]. A pre-amplifier may be used to ensure the full range of the ADC is utilized.

Relevant pre-processing should also be performed, especially when evaluating secured implementations (see Section 7). The role of evaluator is always creative and non-trivial in the sense that the evaluator should consider all possible techniques the attacker may use to increase the chance of attack success. Results of the presented methods must be interpreted carefully, with possible false positives and false negatives in mind [141].

8.1. Welch's t-Test

A two-tailed Welch's t-test can be used to examine a null hypothesis that two groups' means are equal, and can be successfully used in leakage assessment [143,144]. The univariate statistic t is computed for the two groups, in every sampling point independently:

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}}, \tag{53}$$

where $\bar{X}_1$, $\bar{X}_2$ are sample means, s_1^2, s_2^2 are sample standard deviations, and N_1, N_2 are cardinalities of the first and the second group, respectively. The number of degrees of freedom v can be estimated by using

$$v \approx \frac{(\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2})^2}{\frac{(\frac{s_1^2}{N_1})^2}{N_1-1} + \frac{(\frac{s_2^2}{N_2})^2}{N_2-1}}. \tag{54}$$

Under the null hypothesis, the statistic t follows Student's t-distribution with v degrees of freedom. The null hypothesis is rejected according to the distribution and selected significance level α. For sufficiently large n, the t-distribution can be satisfactorily approximated by normal distribution. In side-channel analysis, the threshold ±4.5 or ±5 for the t-value is often considered [141,143], which roughly corresponds to significance level $\alpha \leq 10^{-5}$. Rejecting the null hypothesis suggests that the two groups have different means, and therefore an information leakage. Not rejecting the null hypothesis suggests nothing; most importantly, it does not suggest there is no leakage.

The Welch's *t*-test is a univariate moment-based statistic, similar to statistics used in DPA or CPA attacks (Sections 3.1 and 3.3). The measurements therefore must be aligned. To evaluate leakage exploitable by higher-order attacks, e.g., when evaluating a higher-order masking scheme, relevant (pre-)processing must be performed [144], similarly to the attacks. This includes a use of either univariate or multivariate combining function as described in Section 7.2.

8.2. χ^2 Test

Pearson's χ^2 test of independence tests a null hypothesis that two or more variables are independent, and are well-suited for leakage assessment [145]. Unlike the *t*-test, χ^2 is a nonparametric test: instead of statistical moments, whole underlying distributions are considered. In this subsection, the univariate test is described first, as in case of the *t*-test, i.e., the test is performed at every sampling point independently.

A two-row ($r = 2$) contingency table F is created by using histograms of both groups (assuming aligned histograms, i.e., the same range and width of bins), where the number of columns c corresponds to the number of bins. Columns containing only zeros should be eliminated to decrease number of degrees of freedom. Let $F_{i,j}$ be the frequency of each cell, and N be the number of all measurements. The expected frequency of each cell $E_{i,j}$ is then computed as

$$E_{i,j} = \frac{(\sum_{k=0}^{c-1} F_{i,k}) \cdot (\sum_{k=0}^{r-1} F_{k,j})}{N}, \tag{55}$$

the χ^2 statistic x as

$$x = \sum_{i=0}^{r-1} \sum_{j=0}^{c-1} \frac{(F_{i,j} - E_{i,j})^2}{E_{i,j}}, \tag{56}$$

and the number of degrees of freedom v as

$$v = (c-1) \cdot (r-1). \tag{57}$$

Under the null hypothesis, the statistic x follows χ^2 distribution with v degrees of freedom. The null hypothesis is rejected according to the distribution and selected significance level α, similarly to Welch's *t*-test (Section 8.1). Once again, rejection of the null hypothesis suggests information leakage.

Because χ^2 is a nonparametric test, univariate higher-order leakage is considered inherently. To extend the test to a multivariate case, either the combining function can be utilized (as in case of the *t*-test), or a multivariate histogram can be used. The χ^2 test also enables more than two groups to be used in the test, and it can be also used in an attack scenario similar to the *t*-test in DPA [145].

8.3. Deep Learning Leakage Assessment

The distinguishability of the two groups can also be successfully tested by using a deep learning-based classifier [146]. Assuming there is exploitable leakage, the classifier should be able to learn distinctive features of measurements in each group.

First, the measurements get standardized by subtracting the mean value and then dividing it by the standard deviation, at every sampling point independently. Henceforth, for the classifier, the measurements are considered multivariate vectors. The data are split into training and evaluating sets. The leakage assessment only examines leakage in measurements used in the training stage.

Under a *null hypothesis* that the classifier did not recognize and learn any features, the number of its correct guesses on the evaluating set should follow binomial distribution with probability $p = \frac{1}{2}$. The null hypothesis is rejected once again according to the distribution and selected significance level.

Deep learning leakage assessment provides a powerful tool thanks to its multivariate nature, ability to identify distinctive features, and its detection sensitivity, which outperforms both Welch's t-test and χ^2 test [146]. It displays similar characteristics as machine learning-based attacks (Sections 3.6 and 4.2).

9. Discussion

In this paper, we provided an insight into the side-channel leakage origin and measurements (Section 2) and we presented both non-profiled (Section 3) and profiled (Section 4) attacks as well as attacks on protected implementations with different countermeasures (Section 7).

The non-profiled attacks remain the most powerful in the sense that the attacker can reveal sensitive information, such as cipher keys, with only a little knowledge about the implementation. Correlation power analysis (CPA) is the most effective and efficient attack, assuming the device under attack leaks side-channel information in a "well-behaved and the most common manner", i.e., the leakage follows a linear Hamming weight/distance leakage model and the observation channel follows the normal distribution. On the other hand, mutual information analysis (MIA) relaxes all these assumptions and it is capable of revealing sensitive information without further knowledge about the leakage model. Unlike CPA, which exploits statistical moments such as mean or (co)variance, the MIA works with probability density estimates, thus considering all the information available. However, its generality is typically at the cost of efficiency when simpler attacks are possible to mount. Contrary to the CPA, its effectivity and efficiency strongly depend on the probability density estimation approach and its parameters. Considering attacks on protected implementation, the MIA may become more handy compared to the CPA, as it allows for easy multivariate combining and it inherently considers higher-order univariate leakage, thus not being as susceptible to noise amplification. Therefore, in some cases of protected hardware implementations, it may be able to reveal sensitive information much more efficiently than the CPA [141]. Other non-profiled attacks, such as Kolmogorov–Smirnov analysis (KSA), remain mostly of theoretical interest. Namely, the KSA was studied and used to demonstrate a direct relationship between susceptibility to side-channel attacks and differential cryptoanalysis. Most recently, non-profiled machine learning attacks have emerged, allowing the exploitation of the advantages of neural networks as discussed in the next paragraph for profiled attacks.

The profiled attacks require the attacker to have a fully controlled copy of the device under attack at her disposal. The attacker then examines the leakage characteristics and tailors the attack to the specific device. Under this assumption, the profiled attacks become much more effective and efficient than the non-profiled attacks; where hundreds of measurements are necessary for the non-profiled attack to succeed, the profiled attack may reveal sensitive information with as little as a single measurement. The template attack uses multivariate Gaussian distribution to model the leakage and to mount the attack. Recently, many machine learning algorithms have been used to model leakage. Their main advantage, most prominently in the case of convolutional neural networks, is a scale and translation invariance and their ability to attack protected implementations without further adjustments of the attack. The main disadvantage of the machine learning attacks lies currently in their limited explainability.

In the Section 5, we presented different metrics related to side-channel analysis. The most prominent and widely used metrics are the success rate and the closely related guessing entropy, which both demonstrate the practical effectivity and efficiency of the side-channel attack. We further presented more theoretical metrics such as the distinguishing margin and the confusion coefficient, which were used to demonstrate the aforementioned relationship between physical and theoretical attacks on cryptography: the more difficult it is to break a cipher by using differential cryptanalysis, the easier it is to break it by using side-channel analysis.

We also presented countermeasures (Section 6) against side-channel attacks. These are typically divided into two categories: hiding and masking. While hiding aims to conceal the leaking information in noise, masking aims to randomize the working variables by splitting them into multiple variables and thus making the attack significantly more difficult. Attacking the split variable typically requires more sophisticated attacks (including sample combining), which often result in noise amplification and exponential growth of the attack complexity. It is therefore beneficial to use both hiding and masking countermeasures simultaneously, as the additional noise introduced by the hiding countermeasures significantly boosts the security of masking countermeasures. However, most countermeasures come with non-negligible overhead (in time or area or both) and their implementation often comes at high costs. The selection of appropriate countermeasures and their implementation is therefore highly use-case-dependent with criticality and cost in mind.

Lastly, we presented leakage assessment strategies in Section 8. These are used to evaluate protected implementations and their security by both manufacturers and certification laboratories. As mounting all the possible attacks is infeasible, the leakage assessment uses more general approaches such as the non-specific tests to evaluate whether statistically significant side-channel leakage can be detected or not. In the case of the detected leakage, there is still no evidence of its practical exploitability. But even more importantly, when no leakage is detected, this still cannot be used as proof of security. While the leakage assessment provides beneficial insights, a thorough examination of the secured implementation by a skilled engineer is always necessary.

10. Conclusions

Side-channel analysis has become a widely recognized threat in the last twenty years. It has evolved from simple attacks such as simple or differential power analysis to a complex research field of its own, having a direct impact even on the semiconductor manufacturing technology itself. Nowadays, devices used in both commercial and government sectors must have appropriate certifications to prove themselves secure against side-channel attacks.

In this survey, we described non-invasive vertical side-channel attacks, including machine learning-based attacks, countermeasures against such attacks, and leakage-assessment methodologies. We provided a taxonomy of both attacks and countermeasures with respect to both their theoretical background and the historical context, and we described the advantages and disadvantages of different approaches. In addition to the matter described in this paper, the side-channel analysis further includes horizontal attacks, invasive attacks, or active attacks such as fault injection, which are out of the scope of this paper. Because the countermeasures against all of these attacks are typically expensive and their effectiveness is being put into question with each newly presented attack, further research in the field of side-channel analysis is essential.

Future work in the field of side-channel analysis most importantly consists of security evaluation of protected implementations and security verification. Moreover, as new algorithms are still being proposed (e.g., post-quantum algorithms, light-weight cryptography), their side-channel security must be evaluated, attack vectors must be identified and proper countermeasures proposed. A general approach to side-channel security remains a great challenge due to a consistent back-and-forth between attackers and security engineers. Due to the general inability to foresee all possible threats and weaknesses, the security (including the side-channel security) will always remain a never-ending process instead of the final product.

Author Contributions: Conceptualization, formal analysis, investigation, writing—original draft preparation, visualization, P.S.; supervision, project administration, funding acquisition, V.M. and M.N. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the grant No. SGS20/211/OHK3/3T/18 of CTU Student Grant Competition.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors thank Jan Onderka for proofreading and discussions.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

Cryptography and Side-Channel Analysis related

AES	Advanced Encryption Standard
CPA	Correlation Power Analysis
DDLA	Differential Deep Learning Analysis
DLLA	Deep Learning Leakage Assessment
DM	Distinguishing Margin
DPA	Differential Power Analysis
GE	Guessing Entropy
HD	Hamming Distance
HW	Hamming weight
ID	Identity
iKSA	Inter-class Kolmogorov-Smirnov Analysis
KSA	Kolmogorov-Smirnov Analysis
MIA	Mutual Information Analysis
PPA	Partitioning Power Analysis
RSA	Rivest–Shamir–Adleman cryptosystem
SR	Success Rate
TA	Template Attack

Digital Design and Electronics related

AC	Alternating Current
ADC	Analog-to-Digital Converter
ARM	Advanced RISC (Reduced Instruction Set Computer) Machines
ASIC	Application-Specific Integrated Circuit
CLK	Clock
CMOS	Complementary Metal–Oxide–Semiconductor
DC	Direct Current
EM	Electromagnetic
FPGA	Field-Programmable Gate Array
GND	Ground
HLS	High-Level Synthesis
MOSFET	Metal–Oxide–Semiconductor Field-Effect Transistor
NMOS	N-channel Metal–Oxide–Silicon Transistor
PMOS	P-channel Metal–Oxide–Silicon Transistor
RAM	Random Access Memory
RTL	Register-Transfer Level
SABL	Sense Amplifier-based Logic
SDDL	Simple Dynamic Differential Logic
SNR	Signal-to-Noise Ratio
VHDL	VHSIC (Very High-Speed Integrated Circuits Program) Hardware Description Language
WDDL	Wave Dynamic Differential Logic
XOR	Exclusive OR

Miscellaneous

CNN	Convolutional Neural Network
CPU	Central Processing Unit
DL	Deep Learning
GPU	Graphics Processing Unit
ML	Machine Learning
MLP	Multilayer Perceptron
NIST	National Institute of Standards and Technology
OpenCL	Open Computing Language
OpenMP	Open Multi-Processing
SVM	Support Vector Machine

References

1. Sicari, S.; Rizzardi, A.; Grieco, L.A.; Coen-Porisini, A. Security, privacy and trust in Internet of Things: The road ahead. *Comput. Netw.* **2015**, *76*, 146–164.
2. Daemen, J.; Rijmen, V. The block cipher Rijndael. In Proceedings of the International Conference on Smart Card Research and Advanced Applications, Louvain-la-Neuve, Belgium, 14-16 September 1998; Springer: Berlin/Heidelberg, Germany, 1998; pp. 277–284.
3. *Federal Information Processing Standards Publication 197*; Advanced Encryption Standard. National Institute of Standards and Technology: Gaithersburg, MD, USA, 2001.
4. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126.
5. Kocher, P.C. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 1996; Springer: Berlin/Heidelberg, Germany, 1996; pp. 104–113.
6. Kocher, P.; Jaffe, J.; Jun, B. Differential power analysis. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 15-19 August 1999; Springer: Berlin/Heidelberg, Germany, 1999; pp. 388–397.
7. Quisquater, J.J.; Samyde, D. Electromagnetic analysis (ema): Measures and counter-measures for smart cards. In *Smart Card Programming and Security*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 200–210.
8. Chari, S.; Jutla, C.S.; Rao, J.R.; Rohatgi, P. Towards sound approaches to counteract power-analysis attacks. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 15-19 August 1999; Springer: Berlin/Heidelberg, Germany, 1999; pp. 398–412.

9. Messerges, T.S. Securing the AES finalists against power analysis attacks. In Proceedings of the International Workshop on Fast Software Encryption, New York, NY, USA, 10-12 April 2000; Springer: Berlin/Heidelberg, Germany, 2000; pp. 150–164.
10. Nikova, S.; Rijmen, V.; Schläffer, M. Secure hardware implementation of nonlinear functions in the presence of glitches. *J. Cryptol.* **2011**, *24*, 292–321.
11. Gross, H.; Mangard, S.; Korak, T. Domain-Oriented Masking: Compact Masked Hardware Implementations with Arbitrary Protection Order. In Proceedings of the 2016 ACM Workshop on Theory of Implementation Security, Vienna, Austria, 24 October 2016; p. 3.
12. Tiri, K.; Akmal, M.; Verbauwhede, I. A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards. In Proceedings of the 28th European Solid-State Circuits Conference, Florence, Italy, 24-26 September 2002; pp. 403–406.
13. Tiri, K.; Verbauwhede, I. A logic level design methodology for a secure DPA resistant ASIC or FPGA implementation. In Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, Paris, France, 16-20 February 2004; Volume 1, pp. 246–251.
14. Güneysu, T.; Moradi, A. Generic side-channel countermeasures for reconfigurable devices. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Nara, Japan, 28 September - 1 October 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 33–48.
15. Mentens, N.; Gierlichs, B.; Verbauwhede, I. Power and fault analysis resistance in hardware through dynamic reconfiguration. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Washington, D.C., USA, 10-13 August 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 346–362.
16. Lisovets, O.; Knichel, D.; Moos, T.; Moradi, A. Let's take it offline: Boosting brute-force attacks on iPhone's user authentication through SCA. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2021**, *2021*, 496–519.
17. den Boer, B.; Lemke, K.; Wicke, G. A DPA attack against the modular reduction within a CRT implementation of RSA. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Redwood Shores, CA, USA, 13-15 August 2002; Springer: Berlin/Heidelberg, Germany, 2002; pp. 228–243.
18. Brier, E.; Clavier, C.; Olivier, F. Correlation power analysis with a leakage model. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Cambridge, MA, USA, 11-13 August 2004; Springer: Berlin/Heidelberg, Germany, 2004; pp. 16–29.
19. Chari, S.; Rao, J.R.; Rohatgi, P. Template attacks. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Redwood Shores, CA, USA, 13-15 August 2002; Springer: Berlin/Heidelberg, Germany, 2002; pp. 13–28.
20. Schellenberg, F.; Gnad, D.R.; Moradi, A.; Tahoori, M.B. An inside job: Remote power analysis attacks on FPGAs. In Proceedings of the 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 19-23 March 2018; pp. 1111–1116.
21. Zhao, M.; Suh, G.E. FPGA-based remote power side-channel attacks. In Proceedings of the 2018 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 20-24 May 2018; pp. 229–244.
22. Standaert, F.X. Introduction to side-channel attacks. In *Secure Integrated Circuits and Systems*; Springer: New York, NY, USA, 2010; pp. 27–42.
23. Pant, S. Design and Analysis of Power Distribution Networks in VLSI Circuits. Ph.D. Thesis, The University of Michigan, Ann Arbor, MI, USA, 2008.
24. Rabaey, J.M. *Digital Integrated Circuits: A Design Perspective*; Pearson Education: Upper Saddle River, NJ, USA, 1996.
25. Horowitz, P.; Hill, W.; Robinson, I. *The Art of Electronics*; Cambridge University Press: Cambridge, UK, 1989; Volume 2.
26. Gaubert, P.; Teramoto, A. Carrier mobility in field-effect transistors. In *Different Types of Field-Effect Transistors: Theory and Applications*; InTech, Rijeka, Croatia, 2017; pp. 2–25.
27. Rabaey, J.M.; Chandrakasan, A.P.; Nikolić, B. *Digital Integrated Circuits: A Design Perspective*; Pearson Education, Incorporated: Saddle River, NJ, USA, 2003.
28. Mangard, S.; Oswald, E.; Popp, T. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2008; Volume 31.
29. Moradi, A. *Advances in Side-Channel Security*; Ruhr-Universität Bochum: Bochum, Germany, 2015.
30. O'Flynn, C.; Chen, Z. Synchronous sampling and clock recovery of internal oscillators for side channel analysis and fault injection. *J. Cryptogr. Eng.* **2015**, *5*, 53–69.
31. Camurati, G.; Poeplau, S.; Muench, M.; Hayes, T.; Francillon, A. Screaming channels: When electromagnetic side channels meet radio transceivers. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, Canada, 15-19 October 2018; pp. 163–177.
32. Afzali-Kusha, A.; Nagata, M.; Verghese, N.K.; Allstot, D.J. Substrate noise coupling in SoC design: Modeling, avoidance, and validation. *Proc. IEEE* **2006**, *94*, 2109–2138.
33. Camurati, G.; Francillon, A.; Standaert, F.X. Understanding screaming channels: From a detailed analysis to improved attacks. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2020**, *2020*, 358–401.
34. Gnad, D.R.; Oboril, F.; Kiamehr, S.; Tahoori, M.B. Analysis of transient voltage fluctuations in FPGAs. In Proceedings of the 2016 International Conference on Field-Programmable Technology (FPT), Xi'an, China, 7-9 December 2016; pp. 12–19.

35. Ramesh, C.; Patil, S.B.; Dhanuskodi, S.N.; Provelengios, G.; Pillement, S.; Holcomb, D.; Tessier, R. FPGA side channel attacks without physical access. In Proceedings of the 2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), Boulder, CO, USA, 29 April - 1 May 2018; pp. 45–52.
36. Gierlichs, B.; Batina, L.; Tuyls, P.; Preneel, B. Mutual information analysis. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Washington, D.C., USA, 10-13 August 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 426–442.
37. Standaert, F.X.; Malkin, T.G.; Yung, M. A formal practice-oriented model for the analysis of side-channel attacks. *IACR e-Print Arch.* **2006**, *134*, 2.
38. Messerges, T.S.; Dabbish, E.A.; Sloan, R.H. Examining smart-card security under the threat of power analysis attacks. *IEEE Trans. Comput.* **2002**, *51*, 541–552.
39. Oswald, E.; Mangard, S.; Herbst, C.; Tillich, S. Practical second-order DPA attacks for masked smart card implementations of block ciphers. In Proceedings of the Cryptographers' Track at the RSA Conference, San Jose, CA, USA, 13-17 February 2005; Springer: Berlin/Heidelberg, Germany, 2006; pp. 192–207.
40. Johnson, N.L.; Kemp, A.W.; Kotz, S. *Univariate Discrete Distributions*; John Wiley & Sons: Hoboken, NJ, USA, 2005; Volume 444.
41. Liu, H.; Qian, G.; Goto, S.; Tsunoo, Y. AES key recovery based on Switching Distance model. In Proceedings of the 2010 Third International Symposium on Electronic Commerce and Security, Nanchang, China, 29-31 July 2010; pp. 218–222.
42. Timon, B. Non-profiled deep learning-based side-channel attacks with sensitivity analysis. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2019**, *2019*, 107–131.
43. Bevan, R.; Knudsen, E. Ways to enhance differential power analysis. In Proceedings of the International Conference on Information Security and Cryptology, Seoul, Korea, 28-29 November 2002; Springer: Berlin/Heidelberg, Germany, 2002; pp. 327–342.
44. Canovas, C.; Clédière, J. What do S-boxes say in differential side channel attacks? *IACR Cryptol. ePrint Arch.* **2005**, *2005*, 311.
45. Akkar, M.L.; Bevan, R.; Dischamp, P.; Moyart, D. Power analysis, what is now possible... In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, 3-7 December 2000; Springer: Berlin/Heidelberg, Germany, 2000; pp. 489–502.
46. Le, T.H.; Clédière, J.; Canovas, C.; Robisson, B.; Servière, C.; Lacoume, J.L. A proposition for correlation power analysis enhancement. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Yokohama, Japan, 10-13 October 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 174–186.
47. Batina, L.; Gierlichs, B.; Lemke-Rust, K. Comparative evaluation of rank correlation based DPA on an AES prototype chip. In Proceedings of the International Conference on Information Security, Taipei, Taiwan, 15-18 September 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 341–354.
48. Veyrat-Charvillon, N.; Standaert, F.X. Mutual information analysis: How, when and why? In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Lausanne, Switzerland, 6-9 September 2009; Springer: Berlin/Heidelberg, Germany, 2009; pp. 429–443.
49. Batina, L.; Gierlichs, B.; Prouff, E.; Rivain, M.; Standaert, F.X.; Veyrat-Charvillon, N. Mutual information analysis: A comprehensive study. *J. Cryptol.* **2011**, *24*, 269–291.
50. Silverman, B.W. *Density Estimation for Statistics and Data Analysis*; CRC Press: Boca Raton, FL, USA, 1986; Volume 26.
51. Lemke-Rust, K.; Paar, C. Gaussian mixture models for higher-order side channel analysis. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Vienna, Austria, 10-13 September 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 14–27.
52. Whitnall, C.; Oswald, E. A comprehensive evaluation of mutual information analysis using a fair evaluation framework. In Proceedings of the Annual Cryptology Conference, Santa Barbara, CA, USA, 14-18 August 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 316–334.
53. Standaert, F.X.; Gierlichs, B.; Verbauwhede, I. Partition vs. comparison side-channel distinguishers: An empirical evaluation of statistical tests for univariate side-channel attacks against two unprotected cmos devices. In Proceedings of the International Conference on Information Security and Cryptology, Seoul, Korea, 3-5 December 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 253–267.
54. Whitnall, C.; Oswald, E.; Standaert, F.X. The myth of generic DPA... and the magic of learning. In Proceedings of the Cryptographers' Track at the RSA Conference, San Francisco, CA, USA, 25-28 February 2014; Springer: Cham, Switzerland, 2014; pp. 183–205.
55. Whitnall, C.; Oswald, E.; Mather, L. An exploration of the kolmogorov-smirnov test as a competitor to mutual information analysis. In Proceedings of the International Conference on Smart Card Research and Advanced Applications, Leuven, Belgium, 14-16 September 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 234–251.
56. Maghrebi, H.; Rioul, O.; Guilley, S.; Danger, J.L. Comparison between side-channel analysis distinguishers. In Proceedings of the International Conference on Information and Communications Security, Hong Kong, China, 29-31 October 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 331–340.
57. Cagli, E.; Dumas, C.; Prouff, E. Convolutional neural networks with data augmentation against jitter-based countermeasures. In Proceedings of the International Conference on Cryptographic Hardware and Embedded Systems, Taipei, Taiwan, 25-28 September 2017; Springer: Cham, Switzerland, 2017; pp. 45–68.

58. van der Valk, D.; Picek, S.; Bhasin, S. Kilroy was here: The first step towards explainability of neural networks in profiled side-channel analysis. In Proceedings of the International Workshop on Constructive Side-Channel Analysis and Secure Design, Lugano, Switzerland, 1–3 April 2020; Springer: Cham, Switzerland, 2020; pp. 175–199.
59. Rechberger, C.; Oswald, E. Practical template attacks. In Proceedings of the International Workshop on Information Security Applications, Jeju Island, Korea, 23-25 August 2004; Springer: Berlin/Heidelberg, Germany, 2004; pp. 440–456.
60. Choudary, O.; Kuhn, M.G. Efficient template attacks. In Proceedings of the International Conference on Smart Card Research and Advanced Applications, Berlin, Germany, 27-29 November 2013; Springer: Cham, Switzerland, 2013; pp. 253–270.
61. Kotsiantis, S.B.; Zaharakis, I.; Pintelas, P. Supervised machine learning: A review of classification techniques. *Emerg. Artif. Intell. Appl. Comput. Eng.* **2007**, *160*, 3–24.
62. Hospodar, G.; Gierlichs, B.; De Mulder, E.; Verbauwhede, I.; Vandewalle, J. Machine learning in side-channel analysis: A first study. *J. Cryptogr. Eng.* **2011**, *1*, 293.
63. Heuser, A.; Zohner, M. Intelligent machine homicide. In Proceedings of the International Workshop on Constructive Side-Channel Analysis and Secure Design, Darmstadt, Germany, 3-4 May 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 249–264.
64. Lerman, L.; Bontempi, G.; Markowitch, O. Power analysis attack: An approach based on machine learning. *Int. J. Appl. Cryptogr.* **2014**, *3*, 97–115.
65. Bartkewitz, T.; Lemke-Rust, K. Efficient template attacks based on probabilistic multi-class support vector machines. In Proceedings of the International Conference on Smart Card Research and Advanced Applications, Graz, Austria, 28-30 November 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 263–276.
66. Lerman, L.; Poussier, R.; Markowitch, O.; Standaert, F.X. Template attacks versus machine learning revisited and the curse of dimensionality in side-channel analysis: Extended version. *J. Cryptogr. Eng.* **2018**, *8*, 301–313.
67. Benadjila, R.; Prouff, E.; Strullu, R.; Cagli, E.; Dumas, C. Deep learning for side-channel analysis and introduction to ASCAD database. *J. Cryptogr. Eng.* **2020**, *10*, 163–188.
68. Hettwer, B.; Gehrer, S.; Güneysu, T. Applications of machine learning techniques in side-channel attacks: A survey. *J. Cryptogr. Eng.* **2020**, *10*, 135–162.
69. Martinasek, Z.; Zeman, V. Innovative method of the power analysis. *Radioengineering* **2013**, *22*, 586–594.
70. Martinasek, Z.; Malina, L.; Trasy, K. Profiling power analysis attack based on multi-layer perceptron network. In *Computational Problems in Science and Engineering*; Springer: Cham, Switzerland, 2015; pp. 317–339.
71. Maghrebi, H.; Portigliatti, T.; Prouff, E. Breaking cryptographic implementations using deep learning techniques. In Proceedings of the International Conference on Security, Privacy, and Applied Cryptography Engineering, Hyderabad, India, 14-18 December 2016; Springer: Cham, Switzerland, 2016; pp. 3–26.
72. Kubota, T.; Yoshida, K.; Shiozaki, M.; Fujino, T. Deep learning side-channel attack against hardware implementations of AES. *Microprocess. Microsyst.* **2020**, *87*, 103383.
73. Picek, S.; Heuser, A.; Jovic, A.; Bhasin, S.; Regazzoni, F. The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2019**, *2019*, 1–29.
74. Standaert, F.X.; Malkin, T.G.; Yung, M. A unified framework for the analysis of side-channel key recovery attacks. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, 26-30 April 2009; Springer: Berlin/Heidelberg, Germany, 2009; pp. 443–461.
75. Massey, J.L. Guessing and entropy. In Proceedings of the IEEE International Symposium on Information Theory, Trondheim, Norway, 27 June - 1 July 1994 ; p. 204.
76. Köpf, B.; Basin, D. An information-theoretic model for adaptive side-channel attacks. In Proceedings of the 14th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 31 October - 2 November 2007; pp. 286–296.
77. Fei, Y.; Luo, Q.; Ding, A.A. A statistical model for DPA with novel algorithmic confusion analysis. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Leuven, Belgium, 9-12 September 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 233–250.
78. Heuser, A.; Rioul, O.; Guilley, S. A theoretical study of Kolmogorov-Smirnov distinguishers. In Proceedings of the International Workshop on Constructive Side-Channel Analysis and Secure Design, Paris, France, 13-15 April, 2014; Springer: Cham, Switzerland, 2014; pp. 9–28.
79. Katz, J.; Lindell, Y. *Introduction to Modern Cryptography*; CRC Press: Boca Raton, FL, USA, 2020.
80. Whitnall, C.; Oswald, E. A fair evaluation framework for comparing side-channel distinguishers. *J. Cryptogr. Eng.* **2011**, *1*, 145–160.
81. Mayhew, M.; Muresan, R. An overview of hardware-level statistical power analysis attack countermeasures. *J. Cryptogr. Eng.* **2017**, *7*, 213–244.
82. Matthews, R. On the derivation of a “chaotic” encryption algorithm. *Cryptologia* **1989**, *13*, 29–42.
83. Murillo-Escobar, M.A.; Cruz-Hernández, C.; Abundiz-Pérez, F.; López-Gutiérrez, R.M. Implementation of an improved chaotic encryption algorithm for real-time embedded systems by using a 32-bit microcontroller. *Microprocess. Microsyst.* **2016**, *45*, 297–309.
84. Majumder, B.; Hasan, S.; Uddin, M.; Rose, G.S. Chaos computing for mitigating side channel attack. In Proceedings of the 2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), Washington, D.C., USA, 30 April - 4 May 2018; pp. 143–146.

85. Açikkapi, M.Ş.; Özkaynak, F.; Özer, A.B. Side-channel analysis of chaos-based substitution box structures. *IEEE Access* **2019**, *7*, 79030–79043.
86. Beaulieu, R.; Shors, D.; Smith, J.; Treatman-Clark, S.; Weeks, B.; Wingers, L. The SIMON and SPECK lightweight block ciphers. In Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, 7-11 June, 2015; pp. 1–6.
87. Aumasson, J.-P.; Bernstein, D.J. SipHash: A fast short-input PRF. In Proceedings of the International Conference on Cryptology in India, Kolkata, India, 9–12 December 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 489–508.
88. Joseph, M.; Sekar, G.; Balasubramanian, R. Side channel analysis of SPECK. *J. Comput. Secur.* **2020**, *28*, 655–676.
89. Olekšák, M.; Miškovský, V. Correlation Power Analysis of SipHash. In Proceedings of the 2022 25th International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS), Prague, Czech Republic, 6-8 April 2022; pp. 84–87.
90. Bucci, M.; Giancane, L.; Luzzi, R.; Trifiletti, A. Three-phase dual-rail pre-charge logic. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Yokohama, Japan, 10-13 October 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 232–241.
91. Baddam, K.; Zwolinski, M. Divided backend duplication methodology for balanced dual rail routing. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Washington, D.C., USA, 10-13 August 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 396–410.
92. Razafindraibe, A.; Robert, M.; Maurine, P. Improvement of dual rail logic as a countermeasure against DPA. In Proceedings of the 2007 IFIP International Conference on Very Large Scale Integration, Atlanta, GA, USA, 15-17 October 2007 ; pp. 270–275.
93. Moon, Y.; Jeong, D.K. An efficient charge recovery logic circuit. *IEICE Trans. Electron.* **1996**, *79*, 925–933.
94. Sana, P.K.; Satyam, M. An energy efficient secure logic to provide resistance against differential power analysis attacks. In Proceedings of the 2010 International Symposium on Electronic System Design, Bhubaneswar, India, 20-22 December 2010; pp. 61–65.
95. Choi, B.D.; Kim, K.E.; Chung, K.S.; Kim, D.K. Symmetric adiabatic logic circuits against differential power analysis. *ETRI J.* **2010**, *32*, 166–168.
96. Avital, M.; Dagan, H.; Keren, O.; Fish, A. Randomized multitopology logic against differential power analysis. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2014**, *23*, 702–711.
97. Bouesse, G.F.; Renaudin, M.; Dumont, S.; Germain, F. DPA on quasi delay insensitive asynchronous circuits: Formalization and improvement. In Proceedings of the Design, Automation and Test in Europe, Munich, Germany, 7-11 March 2005; pp. 424–429.
98. Bouesse, F.; Sicard, G.; Renaudin, M. Path swapping method to improve DPA resistance of quasi delay insensitive asynchronous circuits. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Yokohama, Japan, 10-13 October, 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 384–398.
99. Bouesse, F.; Renaudin, M.; Sicard, G. Improving DPA resistance of quasi delay insensitive circuits using randomly time-shifted acknowledgment signals. In *Vlsi-Soc: From Systems To Silicon*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 11–24.
100. Zhu, N.; Zhou, Y.; Liu, H. Counteracting leakage power analysis attack using random ring oscillators. In Proceedings of the 2013 International Conference on Sensor Network Security Technology and Privacy Communication System, Harbin, China, 18-19 May 2013; pp. 74–77.
101. Kamoun, N.; Bossuet, L.; Ghazel, A. Correlated power noise generator as a low cost DPA countermeasures to secure hardware AES cipher. In Proceedings of the 2009 3rd International Conference on Signals, Circuits and Systems (SCS), Medenine, Tunisia, 6-8 November 2009; pp. 1–6.
102. Alipour, A.; Papadimitriou, A.; Beroulle, V.; Aerabi, E.; Hély, D. On the performance of non-profiled differential deep learning attacks against an AES encryption algorithm protected using a correlated noise generation based hiding countermeasure. In Proceedings of the 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 9-13 March 2020; pp. 614–617.
103. Ratanpal, G.B.; Williams, R.D.; Blalock, T.N. An on-chip signal suppression countermeasure to power analysis attacks. *IEEE Trans. Dependable Secur. Comput.* **2004**, *1*, 179–189.
104. Muresan, R.; Gregori, S. Protection circuit against differential power analysis attacks for smart cards. *IEEE Trans. Comput.* **2008**, *57*, 1540–1549.
105. Hubert, G.T. Current Source for Cryptographic Processor. US Patent 7,571,492, 4 August 2009.
106. Shamir, A. Protecting Smart Cards from Power Analysis with Detachable Power Supplies. US Patent 6,507,913, 14 January 2003.
107. Tokunaga, C.; Blaauw, D. Securing encryption systems with a switched capacitor current equalizer. *IEEE J. Solid-State Circuits* **2009**, *45*, 23–31.
108. Mayhew, M.; Muresan, R. On-chip nanoscale capacitor decoupling architectures for hardware security. *IEEE Trans. Emerg. Top. Comput.* **2014**, *2*, 4–15.
109. Pedersen, B.B. Programmable Logic Device with Improved Security. US Patent 8,255,702, 28 August 2012.
110. Clavier, C.; Coron, J.S.; Dabbous, N. Differential power analysis in the presence of hardware countermeasures. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Worcester, MA, USA, 17-18 August 2000; Springer: Berlin/Heidelberg, Germany, 2000; pp. 252–263.
111. Bucci, M.; Luzzi, R.; Guglielmo, M.; Trifiletti, A. A countermeasure against differential power analysis based on random delay insertion. In Proceedings of the 2005 IEEE International Symposium on Circuits and Systems, Kobe, Japan, 23-26 May 2005; pp. 3547–3550.

112. Jeřábek, S.; Schmidt, J.; Novotný, M.; Miškovský, V. Dummy rounds as a DPA countermeasure in hardware. In Proceedings of the 2018 21st Euromicro Conference on Digital System Design (DSD), Prague, Czech Republic, 29-31 August 2018; pp. 523–528.
113. Fumaroli, G.; Martinelli, A.; Prouff, E.; Rivain, M. Affine masking against higher-order side channel analysis. In Proceedings of the International Workshop on Selected Areas in Cryptography, Ontario, Canada, 12-13 August 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 262–280.
114. Prouff, E.; Rivain, M. Masking against side-channel attacks: A formal security proof. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, 26-30 May 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 142–159.
115. Mangard, S.; Pramstaller, N.; Oswald, E. Successfully attacking masked AES hardware implementations. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Edinburgh, UK, 29 August - 1 September 2005; Springer: Berlin/Heidelberg, Germany, 2005; pp. 157–171.
116. Moos, T.; Moradi, A.; Schneider, T.; Standaert, F.X. Glitch-resistant masking revisited. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2019**, *2019*, 256–292.
117. Sasdrich, P.; Moradi, A.; Mischke, O.; Güneysu, T. Achieving side-channel protection with dynamic logic reconfiguration on modern FPGAs. In Proceedings of the 2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), Washington, D.C., USA, 5-7 May 2015; pp. 130–136.
118. Akkar, M.L.; Giraud, C. An implementation of DES and AES, secure against some attacks. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Paris, France, 14-16 May 2001; Springer: Berlin/Heidelberg, Germany, 2001; pp. 309–318.
119. Trichina, E.; Korkishko, T.; Lee, K.H. Small size, low power, side channel-immune AES coprocessor: Design and synthesis results. In Proceedings of the International Conference on Advanced Encryption Standard, Bonn, Germany, 10-12 May 2004; Springer: Berlin/Heidelberg, Germany, 2004; pp. 113–127.
120. Oswald, E.; Mangard, S.; Pramstaller, N.; Rijmen, V. A side-channel analysis resistant description of the AES S-box. In Proceedings of the International Workshop on Fast Software Encryption, Paris, France, 21-23 February 2005; Springer: Berlin/Heidelberg, Germany, 2005; pp. 413–423.
121. Canright, D.; Batina, L. A very compact "perfectly masked" S-box for AES. In Proceedings of the International Conference on Applied Cryptography and Network Security, New York, NY, USA, 3-6 June 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 446–459.
122. Nikova, S.; Rechberger, C.; Rijmen, V. Threshold implementations against side-channel attacks and glitches. In Proceedings of the International Conference on Information and Communications Security, Raleigh, NC, USA, 4-7 December 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 529–545.
123. Moradi, A.; Poschmann, A.; Ling, S.; Paar, C.; Wang, H. Pushing the limits: A very compact and a threshold implementation of AES. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, 15-19 May 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 69–88.
124. Bilgin, B.; Gierlichs, B.; Nikova, S.; Nikov, V.; Rijmen, V. A more efficient AES threshold implementation. In Proceedings of the International Conference on Cryptology in Africa, Marrakesh, Morocco, 28-30 May 2014; Springer: Cham, Switzerland, 2014; pp. 267–284.
125. Bogdanov, A.; Knudsen, L.R.; Leander, G.; Paar, C.; Poschmann, A.; Robshaw, M.J.; Seurin, Y.; Vikkelsoe, C. PRESENT: An ultra-lightweight block cipher. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Vienna, Austria, 10-13 September 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 450–466.
126. Sasdrich, P.; Bock, R.; Moradi, A. Threshold implementation in software. In Proceedings of the International Workshop on Constructive Side-Channel Analysis and Secure Design, Singapore, 23-24 April 2018,; Springer: Cham, Switzerland, 2018; pp. 227–244.
127. Bilgin, B.; Gierlichs, B.; Nikova, S.; Nikov, V.; Rijmen, V. Higher-order threshold implementations. In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, China, 7-11 December 2014; Springer: Berlin/Heidelberg, Germany, 2014; pp. 326–343.
128. Bilgin, B.; Gierlichs, B.; Nikova, S.; Nikov, V.; Rijmen, V. Trade-offs for threshold implementations illustrated on AES. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2015**, *34*, 1188–1200.
129. Poschmann, A.; Moradi, A.; Khoo, K.; Lim, C.W.; Wang, H.; Ling, S. Side-channel resistant crypto for less than 2300 GE. *J. Cryptol.* **2011**, *24*, 322–345.
130. Schneider, T.; Moradi, A.; Güneysu, T. Robust and one-pass parallel computation of correlation-based attacks at arbitrary order. In Proceedings of the International Workshop on Constructive Side-Channel Analysis and Secure Design, Graz, Austria, 14-15 April 2016; Springer: Cham, Switzerland, 2016; pp. 199–217.
131. van Woudenberg, J.G.; Witteman, M.F.; Bakker, B. Improving differential power analysis by elastic alignment. In Proceedings of the Cryptographers' Track at the RSA Conference, San Francisco, CA, USA, 14-18 February 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 104–119.
132. Chu, S.; Keogh, E.; Hart, D.; Pazzani, M. Iterative deepening dynamic time warping for time series. In Proceedings of the 2002 SIAM International Conference on Data Mining, SIAM, Arlington, VA, USA, 11-13 April 2002; pp. 195–212.

133. Sauvage, L.; Guilley, S.; Danger, J.L.; Mathieu, Y.; Nassar, M. Successful attack on an FPGA-based WDDL DES cryptoprocessor without place and route constraints. In Proceedings of the 2009 Design, Automation & Test in Europe Conference & Exhibition, Nice, France, 20-24 April 2009; pp. 640–645.
134. Le, T.H.; Clédière, J.; Servière, C.; Lacoume, J.L. Noise reduction in side channel attack using fourth-order cumulant. *IEEE Trans. Inf. Forensics Secur.* **2007**, *2*, 710–720.
135. Souissi, Y.; Elaabid, M.A.; Debande, N.; Guilley, S.; Danger, J.L. Novel applications of wavelet transforms based side-channel analysis. In Proceedings of the Non-Invasive Attack Testing Workshop, Nara, Japan, 26-27 September 2011.
136. Debande, N.; Souissi, Y.; El Aabid, M.A.; Guilley, S.; Danger, J.L. Wavelet transform based pre-processing for side channel analysis. In Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture Workshops, Vancouver, BC, Canada, 1-5 December 2012; pp. 32–38.
137. Ai, J.; Wang, Z.; Zhou, X.; Ou, C. Improved wavelet transform for noise reduction in power analysis attacks. In Proceedings of the 2016 IEEE International Conference on Signal and Image Processing (ICSIP), Beijing, China, 13-15 August 2016; pp. 602–606.
138. Messerges, T.S. Using second-order power analysis to attack DPA resistant software. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Worcester, MA, USA, 17-18 August 2000; Springer: Berlin/Heidelberg, Germany, 2000; pp. 238–251.
139. Prouff, E.; Rivain, M.; Bevan, R. Statistical analysis of second order differential power analysis. *IEEE Trans. Comput.* **2009**, *58*, 799–811.
140. Waddle, J.; Wagner, D. Towards efficient second-order power analysis. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Cambridge, MA, USA, 11-13 August 2004; Springer: Berlin/Heidelberg, Germany, 2004; pp. 1–15.
141. Standaert, F.X. How (not) to use welch's *t*-test in side-channel security evaluations. In Proceedings of the International Conference on Smart Card Research and Advanced Applications, Montpellier, France, 12-14 November 2018; Springer: Cham, Switzerland, 2018; pp. 65–79.
142. Gilmore, R.; Hanley, N.; O'Neill, M. Neural network based attack on a masked implementation of AES. In Proceedings of the 2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), Washington, D.C., USA, 5-7 May 2015; pp. 106–111.
143. Goodwill, G.; Jun, B.; Jaffe, J.; Rohatgi, P. A testing methodology for side-channel resistance validation. In Proceedings of the NIST Non-Invasive Attack TESTING workshop, Nara, Japan, 26-27 September 2011; Volume 7, pp. 115–136.
144. Schneider, T.; Moradi, A. Leakage assessment methodology. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Saint-Malo, France, 13-16 September 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 495–513.
145. Moradi, A.; Richter, B.; Schneider, T.; Standaert, F.X. Leakage detection with the x2-test. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2018**, *2018*, 209–237.
146. Moos, T.; Wegener, F.; Moradi, A. DL-LA: Deep Learning Leakage Assessment. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2021**, *2021*, 552–598.

Short Biography of Authors

Petr Socha was born in Liberec, Czech Republic, in 1993. He received the Bachelor's degree in Computer Engineering and the Master's degree in Design and Programming of Embedded Systems from the Czech Technical University in Prague, Faculty of Information Technology, in 2016 and 2019, respectively. He is a Research Scientist and a Ph.D. Candidate at the Czech Technical University in Prague. He is the author of 11 conference papers and four journal articles. His research interests include cryptographic hardware, side-channel security, embedded systems, and digital design. He teaches several university subjects and he also teaches IT at an elementary school. He is an external reviewer for the Microprocessors and Microsystems journal. Mr. Socha's awards for his scientific work include the Stanislav Hanzl award (CTU in Prague, 2018) and the Josef Hlávka award (Hlávka Foundation, 2020). Mr. Socha is a member of the IEEE and IACR societies.

Vojtěch Miškovský is an assistant professor at Faculty of Information Technology at Czech Technical University in Prague, where he graduated in 2015 and defended his doctoral dissertation in 2020. His research interests include embedded systems, digital design, hardware security and dependable architectures. He is an author or co-author of 14 journal and conference papers. He is an external reviewer for two journals, Microprocessors and Microsystems and IEEE Transactions on Computers.

Martin Novotný graduated in electrical engineering from the Czech Technical University in Prague, the Czech Republic, in 1992. He received his Ph.D. degree in information security from Ruhr-University Bochum, Germany, in 2009. He is an Assistant Professor and the Head of the Embedded Security Lab at the Czech Technical University in Prague. He serves as a member of the editorial board of Microprocessors and Microsystems journal and a program committee member in several international conferences focusing on cryptography and digital design. He was a program co-chair of DSD 2017, program chair of DSD 2018, and a general chair of CARDIS 2019 conferences. He is an author or co-author of 70+ journal and conference papers and book chapters. His research interests include arithmetic units, hardware for cryptography and cryptanalysis, efficient implementations of cryptographic algorithms, and embedded systems. Dr. Novotný is a member of the IACR society.