Microelectronics Journal 166 (2025) 106866

Contents lists available at ScienceDirect

Microelectronics Journal

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

Exploiting ShiftRow vulnerabilities in addition-chain masked AES: A novel wireless side-channel attack approach

Bijia Cao, Huanyu Wang *, Tuo Deng, Dalin He, Zitian Huang, Junnian Wang

Hunan University of Science and Technology, Xiangtan, Hunan, China

ARTICLE INFO

Keywords:
Wireless side-channel attack
Remote attack
AES
Rivain-Prouff masking
ShiftRows
Deep learning

ABSTRACT

The novel Wireless Side-Channel Attack (WSCA, A contactless side-channel analysis method using unintended EM emissions from target devices.), also known as screaming channel, presents a significant threat to widely deployed IoT edge devices due to its non-contact nature. This attack method has already made significant progress on implementations of unprotected AES. However, when it comes to the case with the presence of masking, the latest research results show that WSCAs require an impractical number of traces (nearly 300K) to barely recover the AES key remotely, which may still be far away from the optimum. In this paper, we go one step further to propose a *ShiftRow*-based WSCA framework to bypass the theoretical strength of the addition-chain based masking approach. By exploring all potential attack points of AES-128 with Rivain-Prouff (RP, A countermeasure against high-order side-channel attacks based on additive chain technology.) masking scheme, our experiments show that targeting on the *ShiftRow* procedure can significantly reduce the protective impact of RP masking on the algorithm. By collaboratively employing two deep-learning models, we successfully compromise an nRF52 SoC implementation of RP-masked AES-128, achieving an attack that is around 80% more efficient than the current state-of-the-art methods. In addition, we further exploit to which extent different combinations of attack points can help the attack on RP-masked AES implementations.

1. Introduction

As the domains of the Internet of Things (IoT) and Artificial Intelligence (AI) continue to evolve rapidly, they are significantly accelerating the pace of global digital transformation. These technological advances have created unprecedented challenges in the area of information security. As a result, the landscape of cybersecurity is not only experiencing the evolution of traditional mathematical-based cryptographic attacks but is also confronting emerging threats from physical side-channel attacks.

Unlike the conventional cryptography, a side-channel attack does not exploit vulnerabilities within the cryptographic algorithm itself. Instead, such attacks capitalize on the correlation between physical measurements taken during the computation process and the device's internal state [1]. This correlation permits attackers to infer the internal operational state of the device, thereby enabling unauthorized information.

Following Kocher's pioneering introduction of differential power analysis in [2], side-channel attacks, particularly power analysis, have rapidly evolved. Concurrently, the swift advancement of machine learning and deep learning techniques has significantly enhanced side-channel attacks [3], providing robust capabilities for analyzing and recognizing subtle leakages during cryptographic processes, thereby increasing the efficiency and success rate of these attacks [4]. At the same time, numerous studies have proved that despite various protective measures adopted by cryptographic systems, deep learning techniques still play a crucial role [5]. Particularly, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) are able to analyze and identify complex data produced by the physical cryptographic implementations with countermeasures [6]. On the past decade, power consumption [7] and electromagnetic emissions [8] have become the most widely exploited and efficient two side channels. However, these approaches are limited by proximity, requiring nearly zero distance for adversaries to capture side-channel information emitted by victim devices, which is very inconsistent with the reality of actual attacks [9].

In 2018, Wireless Side-Channel Attacks (WSCAs), also known as screaming channels, are introduced in [10]. These attacks enable adversaries to compromise devices without direct contact, significantly increasing the feasibility of real-world attacks. [10] finds that the Advanced Encryption Standard (AES, A widely-used symmetric encryption algorithm designed to protect the confidentiality of electronic data.) implementation on a mixed-signal chip may unintentionally couple the side-channel leakage from the core during the execution of instructions

* Corresponding author.
E-mail address: huanyu@hnust.edu.cn (H. Wang).

https://doi.org/10.1016/j.mejo.2025.106866
Received 14 October 2024; Received in revised form 14 May 2025; Accepted 1 September 2025
Available online 10 October 2025

with signals transmitted by the antenna on the chip, and further be emitted to the air. By analyzing these transmitted signals, the adversary can easily recover the senstive data of the chip remotely. Since then, wireless side-channel attacks have been comprehensively developed. By building upon the work in [10], the template attack introduced in [11] successfully employs 5K traces collected from a distance of 15 m, with 1K repetitions, to extract a subkey of AES-128. In 2021, with the help of deep-learning techniques, [12] successfully extracts the AES key from 10K traces without repetitions at 15 m distance to the victim device. Afterwards, Wang et al. introduce a new profiling strategy in [13], which involves training models on 'clean' traces rather than utilizing traces captured remotely from the profiling device. This represents a four-orders-of-magnitude enhancement compared to the template attack presented in [11]. As a relatively recent development in side-channel attacks, there has been limited exploration into WSCAs against protected systems, with masking being the predominant defensive strategies. In 2024, [14] presents the first WSCA on AES implementations protected by the addition-chain masking scheme remotely. However, the results indicate that WSCAs are largely ineffective under such protective measures, necessitating an unrealistically large number of traces (approximately 300 K) to recover the AES key. This inefficacy is attributed by the resilience of the addition-chain based masked Substitution Box (*SBox*), which significantly complicates the attack and makes the current WSCA approaches may still be far away from the optimum.

In this paper, we go one step further to explore how to bypass the theoretical resistance of addition-chain masked AES implementations in WSCAs scenarios. We exploit to which extent different potential attack points can be used by WSCAs to compromise an Nordic Semiconductor nRF52832 development tool kit implementation of AES-128 with the Rivain-Prouff masking scheme. Due to its characteristic of provable security against any attack of order lower than $d + 1$, RP masking scheme becomes one of the most popular addition-chain based approaches in security-sensitive circuits [15]. In addition, we further propose a *ShiftRow*-based WSCA framework to overcome the protections offered by the addition-chain based masking. By employing a dual deep-learning model analysis strategy, we successfully compromise an nRF52 SoC implementation of RP-masked AES-128 by using 80% less traces than the state-of-the-art methods [14] under the same condition. In summary, our contributions are as follows.

(1) Firstly, we extensively explore all potential attack points on RP-masked AES implementations within WSCAs and experimentally demonstrate that *ShiftRow*, instead of the well-protected *SBox* procedure, serves as a more effective target for attacking addition-chain masked AES implementations.
(2) Next, we introduce a *ShiftRow*-based deep-learning side-channel attack framework in the context of WSCAs, which successfully compromises RP-masked AES implementations remotely with an 83% increase in attack efficiency.
(3) Afterwards, we integrate various attack points into ensembles and investigate to which extent different combinations can further help the attack.

The rest of this paper is organized as follows. Section 2 reviews the theoretical background and related works of the paper. Section 3 describes our experimental setup and the selection of deep learning models. Section 4 presents the analysis process of the captured traces and the search of attack points. Sections 5 and 6 cover the proposed method and experimental results. Section 7 concludes the paper and discusses the future work.

2. Background

This section primarily introduces the AES encryption algorithm and explores the RP masking scheme. Afterwards, this section explains the application of deep learning techniques in side-channel attacks and the principles and mechanisms of emerging wireless side-channel attack methods.

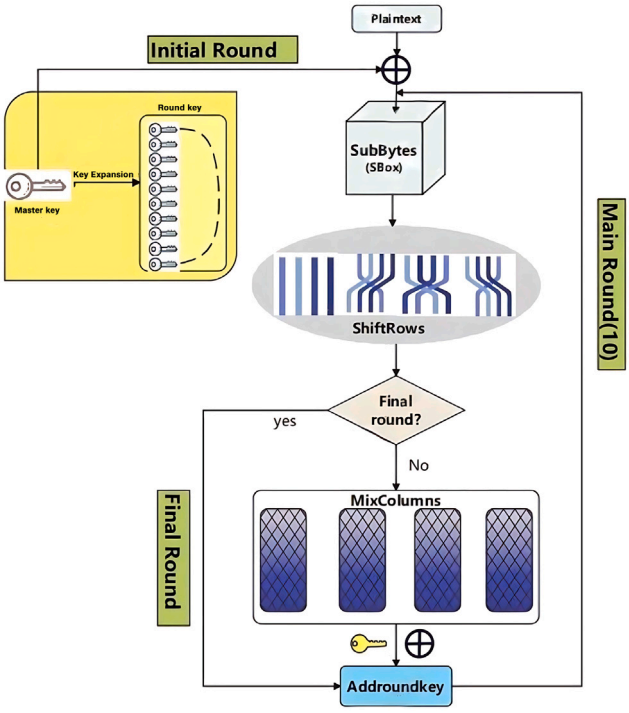


Fig. 1. An illustration of the AES encryption process.

2.1. AES

The core design of the Rijndael encryption algorithm is based on Galois Field (GF, A finite field containing a finite number of elements.) theory, specifically GF(2^8). In this finite field, there are 256 elements contained, each of which can be regarded as a polynomial. The algorithm not only performs efficient arithmetic operations, but also maintains the closure property and feasibility of inverse operations [16]. This design greatly enhances its resistance to attacks based on mathematical analysis. Due to its superior security and efficiency, Rijndael was selected by the National Institute of Standards and Technology (NIST) in 2001 as the basis for the AES [16], a block cipher algorithm, which is a symmetric encryption type.

In the standard implementation of AES, the block size is fixed to 128 bits, while key lengths of 128, 192, or 256 bits are supported. This paper focuses on the AES-128 where a 128-bit key is used to encrypt blocks of equal length. The encryption process of AES-128, as shown in Fig. 1, includes key expansion and ten rounds of iterative encryption. In the key expansion phase, the initial key is expanded into a series of subkeys (round keys) that are used in each subsequent encryption round. Throughout the encryption process, apart from the first round that only performs the Addroundkey step and the last round that does not execute MixColumns, each round includes the following four steps: SubBytes, ShiftRows, MixColumns, and Addroundkey [17]. The SubBytes procedure is implemented by utilizing a lookup table called SBox, the construction of which relies on the multiplicative inverse of the elements in GF(2^8), which is a key source of the algorithm's nonlinear characterization.

The security of the AES algorithm relies heavily on the nonlinear properties of the SBox and the diffusion properties of the MixColumns procedure [18]. These designs not only increase the complexity of the encryption process but also help the algorithm to resist advanced cryptanalytic techniques such as linear and differential cryptanalysis. Through these meticulously designed multi-layer security measures, AES provides robust data protection, ensuring the secure transmission and storage of information globally.

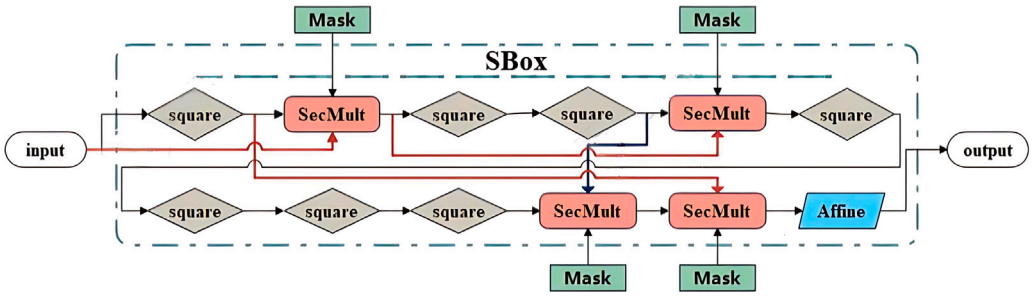


Fig. 2. An illustration of the masked SBox operation flow under the RP masking scheme based on addition chains.

However, despite the superior design of AES algorithms, the emergence of side-channel attacks poses a challenge to their implementations, its nonlinear nature may cause inadvertent physical leakage of information, thus putting the entire encryption system at risk, and there attacks can exploit the unique leakage patterns generated during nonlinear operations.

2.2. AES with the RP masking scheme

Side-channel attacks can bypass the strength of traditional encryption algorithms by exploiting minute leaks in physical operations to obtain sensitive information. Since these physical leakages are difficult to completely eliminate with current technology, effectively resisting side-channel attacks has become a crucial and unavoidable task in the field of information security.

In addition to physical isolation and shielding, the introduction of randomness is one of the widely used countermeasures. Masking is one of the main randomization techniques. It combines sensitive data with randomly generated mask values to reduce the dependence of sensitive variables on physical subtle leakage, and the actual data is modified by the mask to protect the key.

The basic principle of the masking scheme is to divide the sensitive variable x into $d+1$ shares $x_0, x_1, \ldots, x_d$, and ensure they satisfy the following relationship:

$$x = x_0 \oplus x_1 \oplus \cdots \oplus x_d \tag{1}$$

Among them, $x_1, x_2, \ldots, x_d$ are referred to as random masks, and x_0 is referred to as the masked value. When each sensitive variable involves d random masks, the masking is referred to as d-order. Theoretically, when d-order masking techniques are applied, any effective attack requires at least $d+1$ orders of complexity [15].

Typically, during AES encryption, the security of the algorithm greatly depends on its nonlinear characteristics [2]. However, these nonlinear characteristics often become the main target of side-channel attacks, with the SubBytes as the core nonlinear operation being a focal point for such attacks. There are three main reasons for this. Firstly, this operation is usually realized through table lookups, which require memory accesses, thus potentially exposing the memory access patterns associated with the input data. Secondly, due to the nature of nonlinear operations, different input data can lead to significant differences in processing time and power consumption, which can be captured and identified by side-channel analysis techniques. Finally, nonlinear functions generally exhibit high data dependency, meaning the output not only depends on the input values but may also depend on specific input bits in a complex manner. This data dependency increases the likelihood of inferring internal states through side-channel attacks.

The RP masking scheme is the first securely verified method based on addition chains. It supports masks of any order and can effectively resist side-channel attacks [15]. This scheme particularly demonstrates its cost-effectiveness and security advantages when handling the SubBytes of the SBox. By utilizing addition chain techniques, the traditional lookup table-based SBox is transformed into a more secure and efficient masked SBox. This new type of masked SBox does not rely on predefined lookup tables but instead achieves the functionality of the SBox through a series of carefully designed squaring and multiplication operations. This significantly reduces the risk of side-channel attacks leaking sensitive information through memory access patterns or power consumption analysis. Algorithm 1 shows the pseudocode of the specific operations for the RP masking scheme. The nonlinear multiplication operation **SecMult()** uses the Ishai–Sahai–Wanger (ISW) scheme [19], while the pseudocode for the function describing the operation of adding random masks **RefreshMasks()** is shown in Algorithm 2.

Algorithm 1 Pseudo-code of the SBox operation in AES-128 with RP masking scheme [14]

// **SecSBox**
// **in**: share x_i satisfying $\oplus i x_i = x$
// **out**: share y_i satisfying $\oplus i y_i = \mathrm{SBox}(x)$
for $i = 0$ to d **do**
  $z_i \leftarrow x_i^2$ ▶ $\oplus i z_i = x^2$
end for
RefreshMasks$(z_0, z_1, \ldots, z_d)$
$(y_0, y_1, \ldots, y_d) \leftarrow$ **SecMult**$((z_0, z_1, \ldots, z_d), (x_0, x_1, \ldots, x_d))$
for $i = 0$ to d **do**
  $w_i \leftarrow y_i^4$ ▶ $\oplus i w_i = x^{12}$
end for
RefreshMasks$(w_0, w_1, \ldots, w_d)$
$(y_0, y_1, \ldots, y_d) \leftarrow$ **SecMult**$((y_0, y_1, \ldots, y_d), (w_0, w_1, \ldots, w_d))$
for $i = 0$ to d **do**
  $y_i \leftarrow y_i^4$ ▶ $\oplus i y_i = x^{15}$
end for
$(y_0, y_1, \ldots, y_d) \leftarrow$ **SecMult**$((y_0, y_1, \ldots, y_d), (y_0, y_1, \ldots, y_d))$
$(y_0, y_1, \ldots, y_d) \leftarrow$ **SecMult**$((y_0, y_1, \ldots, y_d), (z_0, z_1, \ldots, z_d))$
for $i = 0$ to d **do**
  $y_i \leftarrow \mathrm{Af}(y_i)$
end for
if $d \mod 2 = 1$ **then**
  $y_0 \leftarrow y_0 \oplus 0x63$
end if

Algorithm 2 Pseudo-code of refreshing masks in AES-128 with the RP masking scheme [14]

// **RefreshMasks**
// **in**: share x_i satisfying $\oplus i x_i = x$
// **out**: share x_i satisfying $\oplus i x_i = x$
for $i = 0$ to d **do**
  mask $\leftarrow$ random()
  $x_0 \leftarrow x_0 \otimes$ mask
  $x_i \leftarrow x_i \otimes$ mask
end for

In this scheme, polynomial expansion is used to define the core operations of the SBox, with the key introduction of randomized masking to enhance security. The addition of random masks can reduce the dependency between inputs and outputs, breaking the direct association during data processing. Moreover, since the nonlinear operations are

replaced by using arithmetic procedures instead of using the lookup table, this method further mitigates the risks posed by physical execution differences of procedures, such as slight variations in execution time and power consumption. Consequently, this scheme effectively reduces the risk of side-channel leakage and enhances the overall security of the AES implementions. The illustration of the masked SBox in the RP scheme is shown in Fig. 2. This paper focuses on studying the effects and methods of wireless side-channel attacks on implementations of AES under the protection of the addition chain-based RP masking scheme.

2.3. Deep learning side-channel attacks

Since 2013, when Martinasek and Zdenek used a basic deep learning model, MLP, for side-channel attacks [20], deep learning side-channel attacks have rapidly developed. Deep learning, due to its complex and opaque internal mechanisms, is often likened to a "black box". Researchers can only observe the input and output results, while the intermediate process is completed by multiple layers of non-linear transformations within the model. This "black box" characteristic aligns well with the need in side-channel attacks to infer internal information by analyzing external leakages. The advantage of this method is that, through deep learning, attackers can effectively recover keys using physical leakage information without fully understanding the internal structure of the encryption algorithm.

Deep learning-based side-channel attacks typically consist of two phases: the profiling phase and the attack phase. In the profiling phase, researchers train deep neural networks to find the subtle relationships between leakage information and the key. Specifically, the deep learning model learns and extracts relevant patterns between side-channel information and the key by training on large amounts of data. In the attack phase, attackers use the patterns learned by the trained model to classify new leakages, thereby obtaining sensitive information.

Deep learning has been widely applied in side-channel attacks. Various deep learning architectures have been applied to side-channel analysis, and different models exhibit distinct performance during attacks. [20] uses a simple three-layer MLP model to analyze the power consumption trace and successfully classified the first byte of the key. [21] explores and demonstrated the attack effects of various deep learning methods, including CNN, LSTM, MLP, and stacked autoencoder (AE) models, on unprotected and protected AES implementations. [8] shows how to select the hyperparameters of deep learning models, while [22] introduces the attention mechanism, which can better focus on the parts of the leakage related to the key. In addition, [23] proposes a categorical cross entropy loss function for quantifying the network classification error, while [24] proposes an optimization method that minimizes loss by dynamically adjusting the learning rate. [25] introduces federated learning to improve the attack efficiency by performing model-level aggregation on locally trained models.

2.4. Wireless side-channel attacks

Traditional side-channel attack methods, such as power and electromagnetic analysis, typically require direct physical contact to the target device to collect the necessary traces. However, such physical contact is often difficult to achieve in practical scenarios, posing a limitation to side-channel attacks. The emerging wireless side-channel attack effectively addresses this limitation. This paper focuses on studying the effects and methods of wireless side-channel attacks on AES implementations under the protection of the addition chain-based RP masking scheme. Through this research, we aim to develop more effective attack strategies to address the impracticality of wireless side-channel attacks when countermeasures are in place.

In [10], researchers discovered that during the execution of the AES algorithm on an ARM Cortex M4 CPU, physical leakages might inadvertently couple with radio carrier signals on a mixed-signal circuit board. By capturing and analyzing these signals, attackers can remotely recover the key. Radio Frequency (RF) integrated circuits (also known as mixed-signal circuits) contain both analog and digital components and are highly favored due to their broad market demand. Since the RF module in the analog part is highly sensitive to noise, when digital circuits and analog circuits are integrated on the same silicon chip, the noise generated by the digital part executing instructions may be unintentionally amplified [26]. This means that during encryption operations, the RF module of the analog circuit might be affected by side-channel information leakage from the digital circuit, amplifying these signals and transmitting them through a wireless transmission channel, enabling remote trace capturing.

The principle of WSCAs, as illustrated in Fig. 3, is as follows. Typically, a chip is divided into a digital circuit section and an analog circuit section. When the encryption module in the digital section performs encryption operations, the inadvertently leaked side-channel information is easily modulated by the square wave noise generated during the system clock operation of the CPU core. When the modulated signal is transmitted from the digital circuit section to the analog circuit section, baseband coupling occurs due to differences in circuit structure. In the analog circuit section, these signals further undergo capacitive coupling with the baseband signals of the Voltage-Controlled Oscillator (VCO). Finally, the RF module transmits these signals to a high frequency, customized by the wireless transmission protocol, and sends them out via the antenna [14]. This effect primarily results in amplitude modulation due to capacitive coupling (at the circuit level). In addition, substrate coupling can also be caused by impact ionization current (at the device level) and resistive coupling (at the chip level) [27]. In this way, we can collect the necessary traces using a radio receiver without physically contacting the victim device.

To collect such traces, the center frequency of the receiver needs to be set to an appropriate value, otherwise it will be difficult to collect the true wireless side channel information. The following is the reason for selecting the value of the center frequency. Firstly, the clock signal generated by the square wave is expressed in the time domain and frequency domain as shown in formula (2).

$$\begin{aligned} s(t) &= \sum_{n=-\infty}^{+\infty} S_n e^{i2n\pi f_s t} \\ S(f) &= \sum_{n=-\infty}^{+\infty} S_n \delta(f - nf_s) \end{aligned} \tag{2}$$

Among them, $S_n = \frac{\sin(n\pi\alpha)}{n\alpha}$ is the coefficient of the Fourier series, f_s is the clock frequency of the square wave, α represents the duty cycle of the square wave, and δ is the pulse function. However, in reality, square wave noise is not an ideal square wave, so the even terms of the Fourier series are not completely zero.

The leaked side channel signal $e(t)$ during the encryption process is modulated by square wave noise. The modulated signal $c(t) = e(t) * s(t)$ is expressed in the frequency domain as follows.

$$C(f) = E(f) * S(f) = \sum_{n=-\infty}^{+\infty} S_n E(f - nf_s) \tag{3}$$

In the analog circuit part, the modulated signal is transmitted by the RF module to the high frequency defined by the wireless transmission protocol due to the substrate coupling phenomenon, and is transmitted through the antenna. At this time, the signal $m(t)$ is expressed as formula (4).

$$m(t) = \sum_{n=-\infty}^{+\infty} S_n c(t) e^{i2\pi(nf_s + f_c)t} \tag{4}$$

After Fourier transform, the frequency domain representation of the signal $M(f)$ is as follows.

$$M(f) = \sum_{n=-\infty}^{+\infty} S_n C(f - nf_s - f_c) \tag{5}$$

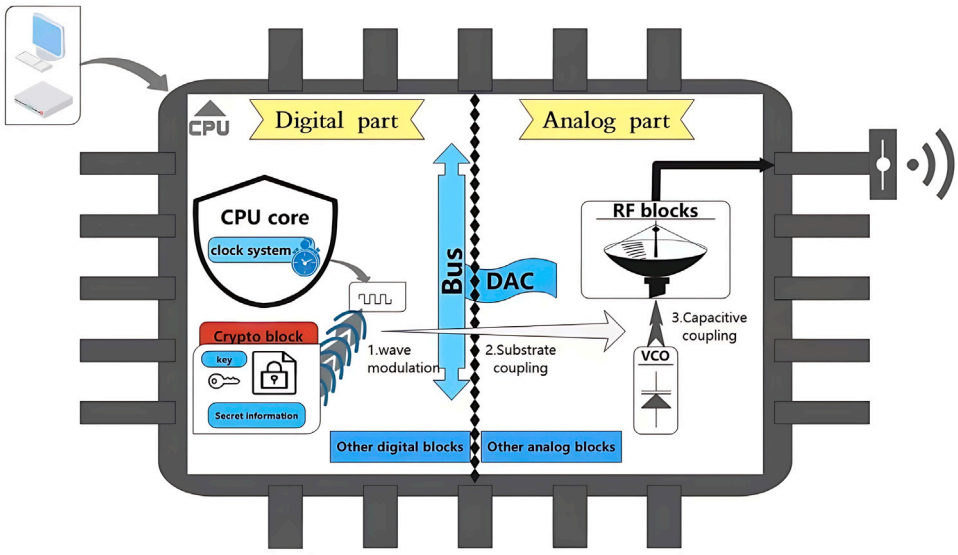


Fig. 3. An illustration of how traces can be captured remotely through wireless channel.

where f_c is the carrier frequency and $e^{i2\pi f_c t}$ represents the radio carrier.

At the radio receiver, if the center frequency is set to $f_{\text{center}} = Nf_s + f_c$, the time domain $r(t)$ and frequency domain $R(f)$ of the received signal are expressed as shown in formula (6). The side channel signal $e(t)$ can be recovered using a low-pass filter [14].

$$\begin{aligned} r(t) &= \sum_{n\neq N} S_n e(t) e^{i2\pi(n-N)f_s t} + S_N e(t) \\ R(f) &= \sum_{n\neq N} S_n E(f-(n-N)f_s) + S_N E(f) \end{aligned} \tag{6}$$

3. Experimental setup

The experimental process is divided into two stages: the collection stage and the attack stage. In the experiment, we use Nordic semiconductor nRF52832 Soc devices to encrypt plaintext and continuously send data. Meanwhile, on the receiving end, we use the Ettus USRP N210 Software Defined Radio (SDR) to capture amplitude-modulated electromagnetic traces as shown in Fig. 4. Subsequently, we select an appropriate deep learning architecture for side-channel attacks.

3.1. Traces collection

At the transmitter side, we select the nRF52 development kit as the victim device. This is a versatile single-board based on the Arm Cortex-M4 CPU, installed on the Nordic nRF52 DK board, supporting Bluetooth 5 with a data transmission rate of 2Mbps and a floating-point unit running at a frequency of 64 MHz. The kit is suitable for developing low-power Bluetooth, NFC, ANT, and 2.4 GHz proprietary protocols on the nRF52832 SoC. We implement a provably secure additive chain-based random masking scheme (RP masking) AES-128 encryption algorithm on the nRF52832 SoC. During the experiment, we use identical devices, called d1 and d2, respectively.

3.2. Trace processing

On the receiving side, we use the Ettus Research USRP N210 SDR as the receiver, connected to a VERT2450 vertical antenna with a gain of 3dBi to receive signals and collect electromagnetic traces. The center frequency of the receiver is set to $2f_{\text{clock}} + f_{\text{Bluetooth}} = 2.528$ GHz, where $f_{\text{Bluetooth}} = 2.4$ GHz. The sampling frequency is set to 5 MHz. To ensure data accuracy, the two devices are placed in a quiet environment 10 cm apart. After obtaining the sampled signals, a low-pass filter with a cutoff frequency of 5 KHz was used to process the signals to obtain suitable traces.

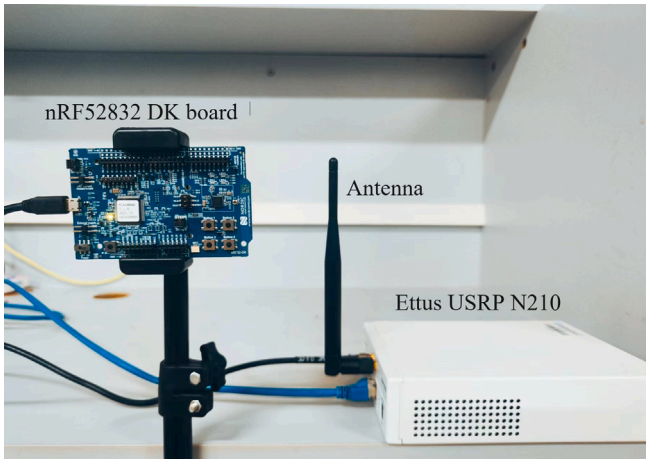


Fig. 4. Trace collection experimental setup.

3.3. Model architecture

We collect a large amount of trace data and use deep learning models to reveal the correlation between the traces and the key. This method fully leverages the advantages of modern deep learning, overcoming the limitations of traditional analysis methods. The Multilayer Perceptron (MLP) model, as a widely used deep learning architecture, has shown significant advantages in side-channel attacks.

The MLP is a type of feedforward neural network with high flexibility and adaptability. Through multiple hidden layers, the MLP can process and transform input data at multiple levels, capturing complex nonlinear relationships. Trace data is usually high-dimensional, containing a large amount of temporal information and signal features. The MLP is particularly effective in handling this high-dimensional data because it can iteratively extract and transform features layer by layer, gradually reducing the data dimensionality and finally extracting key features related to the key.

Compared to CNN or RNN, the MLP has a relatively simple structure and lower computational requirements, making it more suitable for attack scenarios with limited computational resources. Overall, the application of the MLP model in side-channel attacks not only efficiently utilizes computational resources but also accurately extracts key

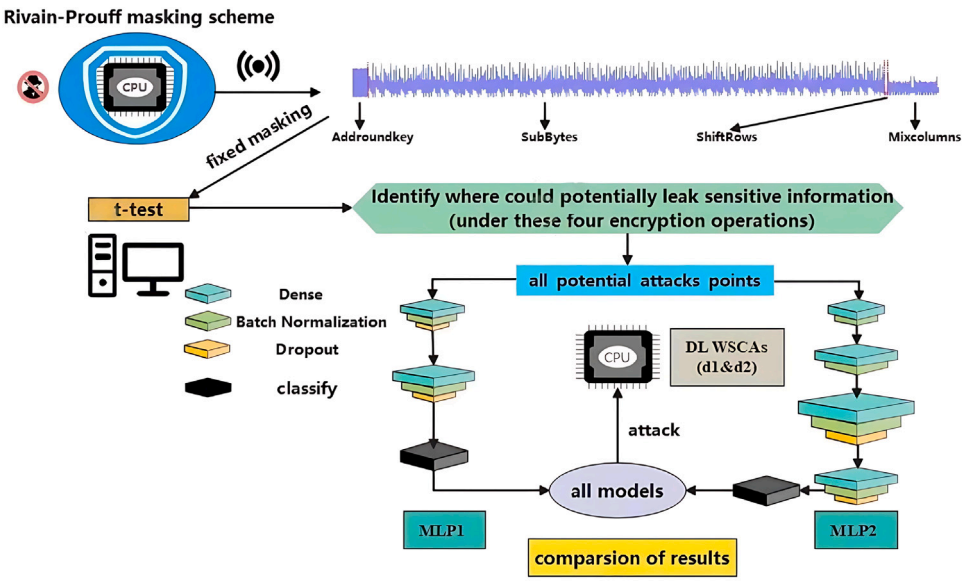


Fig. 5. An illustration of the attack strategy.

features from trace data, significantly improving the success rate of the attack. This method not only addresses the shortcomings of traditional methods in handling complex high-dimensional data but also maintains efficiency and accuracy under resource constraints, fully demonstrating the potential and practicality of the MLP model in side-channel attacks.

In this experiment, we conduct a lot of model training and parameter adjustment. Eventually, we choose two different MLP model architectures, one with a deeper neuron layer and the other with a shallower neuron layer, to compare the results and analyze their security performance in different attack scenarios. These two MLP model architectures of different complexity were trained on traces respectively.

3.4. Attack strategy

Based on the addtion-chain RP masking scheme, AES-128 encryption is effectively protected. As noted in [14], both deep learning-based electromagnetic side-channel attacks and deep learning-based wireless side-channel attacks require an unrealistically large number of traces to barely succeed in recovering the key when facing devices protected by RP masking. In this paper, we propose an attack strategy targeting devices protected by the RP masking scheme. First, we collect wireless side-channel traces from a distance of 10 cm for leakage detection. Due to the effective protection of the RP masking scheme, we adopt a fixed mask approach to identify potentially leaked trace segments. Through repeated experimentation, we identified two suitable model architectures and trained several models using the leaked trace segments. Subsequently, we employed these trained models to attack both the original device (d1) and a similar device (d2), and compared the attack results. After numerous attack attempts, we ultimately determine that a *ShiftRow*-based wireless side-channel attack strategy is the most suitable approach to breaking through RP masking protection. The specific strategy is illustrated in Fig. 5.

4. Trace analysis

From the captured signals, we extract the traces related to the AES execution with RP mask scheme protection. By using the correlation coefficient calculated between the pre-processed trace segment template and the captured signal, we can accurately determine the first round of encryption operation of each AES execution block. Thus, we are able to locate the starting point of the encryption operation.

4.1. Trace processing

In side-channel attack analysis, the collected traces need to go through a series of preprocessing steps to improve data quality and degenerate the level of noise and interference. In [13,14], researchers use "clean" traces captured through coaxial cables to train deep learning models. This is because coaxial cables can directly connect transmitters and receivers, transmit high-frequency oscillation signals without radiating to the outside, and enable the receiver to directly receive RF signals from the chip's RF module. Specifically, in [13], the collected traces are aligned, scaled, and averaged. Alignment helps us to accurately synchronize traces. Scaling is necessary because the traces captured by the coaxial cable are not in the same range as the traces captured at the actual operating distance. Averaging helps to improve trace quality and reduce noise. In Ref. [14], due to the use of RP random masks, it is impossible to average and reduce noise on the traces when training the deep learning model.

Unlike the traces collection for model training using coaxial cables as mentioned in [14], our research directly utilizes traces captured at a distance of 10 centimeters for training deep learning models. This approach makes our model's generalization capacity closer to real-world application scenarios, despite potentially facing more noise and interference. Although we lack the "clean" traces from coaxial cables, scaling is still necessary due to our experiment involving cross-device studies. Scaling adjusts traces from different sources to the same scale, facilitating comparison and analysis, while improving the accuracy and consistency of the processing. Specifically, we use the min–max scaling method [28] to map the amplitude of all traces to the range $[0, 1]$. Given a set of traces T, each trace $T = (\tau_1, \ldots, \tau_m) \in \mathcal{R}^m$ is converted to $T' = (\tau'_1, \ldots, \tau'_m) \in \mathcal{I}^m$ for every i in the range $\{1, \ldots, m\}$. The scaling is performed as shown in formula (7):

$$\tau'_i = \frac{\tau_i - \tau_{\min}}{\tau_{\max} - \tau_{\min}} \tag{7}$$

where $\tau_{\min}$ and $\tau_{\max}$ are the minimum and maximum data points in T.

Fig. 6 is the plot of an example trace after the preprocessing, which is a segment containing 7000 points. This segment represents the average of 1000 traces collected from the profilling device d1 (the average is used here only to better observe the overall shape of the trace). From the figure, we can observe the distribution of various encryption operations after trace preprocessing. The range of the round key addition in the first round is approximately between 500 and 700. The distribution of the encryption operations in the second

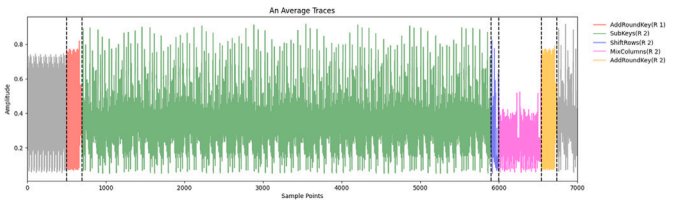


Fig. 6. The shape of the trace averaged from 1k traces collected from the D1 device with the RP masking protection.

round is as follows: the range of the SubBytes operation in the SBox is approximately from 700 to 5900, the ShiftRows operation ranges from approximately 5900 to 6000, the MixColumns operation ranges from approximately 6000 to 6540, and the Addroundkey operation ranges from approximately 6540 to 6740. Additionally, the traces left by the RP mask are clearly visible. For example, after the masked S-box processing, in the ShiftRows and MixColumns operations, two similar trace segments indicate that the intermediate value has been split into two shares x0 and x1 under the influence of the mask. The trace segments in the ShiftRows operation are shorter than in the MixColumns, but since the operation range is approximate, it causes blurred boundaries, making it difficult to clearly divide them into two segments, although they have actually been split into two shares.

4.2. Leakage detection

In side-channel attacks, the process of identifying the intervals where side-channel information is leaked to recover keys and sensitive information is known as leakage detection. The purpose of leakage detection is to identify and analyze the physical weaknesses in the system during encryption operations that could be exploited by attackers. From Fig. 6, we can see that the captured traces consist of 7000 sample points representing the execution of the first round of AES-128 with the first-order RP mask protection. To reduce time and resource consumption, we use leakage detection to identify Points of Interest (POI) within the traces, allowing more effective attacks.

In existing leakage detection methods, Test Vector Leakage Assessment (TVLA, A security method analyzing side-channel leakage under varying inputs via statistical tests (e.g., t-test) to detect sensitive information leaks like cryptographic keys.), based on the well-known Welch's t-test, has become one of the most widely used statistical techniques for detecting first-order leakage. We selected the Hamming weight (HW, A count of 1s in a binary representation, used in side-channel attacks to model leakage (e.g., power/EM) correlation with key-dependent intermediate values like AES S-box outputs.) as the leakage model, fundamentally grounded in the linear correlation between the intensity of the side channel signal and the number of '1' bits in registers when hardware devices process data. Within the framework, the Hamming weight is used to quantify the characteristics of intermediate values associated with both the cryptographic key and the input data. We divided the traces into two subsets based on the Hamming weight of the value at the attack point: traces with HW(labels) > t as t_0 and traces with HW(labels) < t as t_1. The threshold is set to 4 based on two principles: First, the Hamming weight ranges from 0 to 8 (for 8-bit data), and the median value of 4 balances the sample sizes of both groups to avoid statistical bias. Secondly, if masking protections are flawed (e.g. insufficient randomness), the Hamming weight distribution of key-dependent intermediate values may exhibit asymmetry around the threshold of 4 (e.g., a significant deviation in the proportion of the high group from the theoretical random distribution). By amplifying the signal-to-noise ratio by comparing power consumption differences between groups, this approach effectively exposes vulnerabilities in masked implementations.

The t-test detects leakage by determining whether there is a significant difference between the means of two samples. The key steps are setting up hypotheses, calculating the test statistic, finding the critical value, and making a decision. The null hypothesis is that the means of the two subsets are equal. When the null hypothesis is rejected, it indicates that the sample means are significantly different, implying that leakage has been detected. In TVLA, to quantify leakage, we calculate the Sum of Squared Differences (SOST) of pairwise t-tests for two trace subsets t_0 and t_1. The formula (8) for the SOST values of the two trace subsets is as follows.

$$SOST = \left(\frac{\overline{X}_0 - \overline{X}_1}{\sqrt{\frac{s_0^2}{n_0} + \frac{s_1^2}{n_1}}} \right)^2 \tag{8}$$

where $\overline{X}_i$ represents the mean of trace subsets t_i, s_i represents the variance of T_i, n_i represents the number of traces in T_i, $i \in \{0, 1\}$.

4.3. Selection of attack points

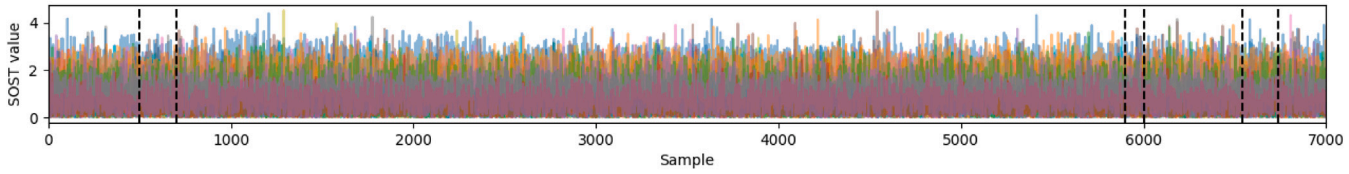
When performing the t-test, we use the value of the SBox input $S_{in} = (p_j \oplus k_j)$ in the first round and the value of the SBox output $S_{out} = SBox(p_j \oplus k_j)$ in the second round of AES-128 as the label (depending on the key or the intermediate value of the input). We use 5k traces captured at a distance of 10 cm and divides into two groups, where the threshold is 4, $HW(\text{label}) > 4$ and $HW(\text{label}) < 4$. The test results are shown in Fig. 7.

As can be seen from Fig. 7, the RP random masking scheme can effectively protect the AES algorithm. No matter whether the label we use, we cannot see any leakage signs on the trace. Since the mask value is randomly generated during each encryption operation, the traces generated by the same plaintext or ciphertext in different encryption processes are different. Specifically, the mask randomizes the intermediate calculation values in the AES algorithm, making it difficult to infer the key or other sensitive information by analyzing the traces.

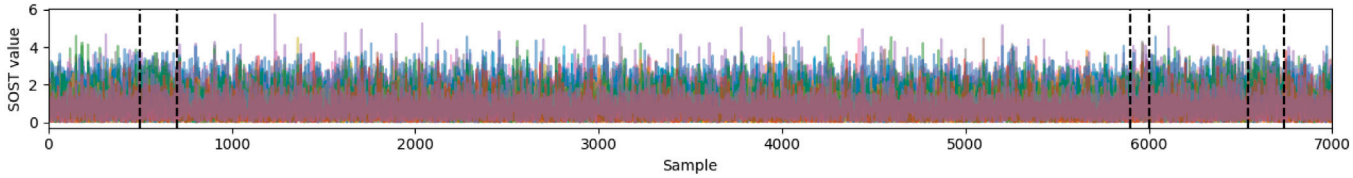
Although mask randomization increases the complexity of leakage detection, we construct a controlled experimental environment to bypass protection limitations by fixing the RP mask (set to 0 during the experiment). To better determine the leakage interval, we collected 5000 wireless signal traces from device d1 at a distance of 10 cm and fixed the RP mask to 0 during the experiment. First, fixing the mask reduces its protective effect on the AES encryption algorithm. Second, when collecting traces, traditional averaging noise reduction methods can be used to improve the signal-to-noise ratio (SNR) – this is because when the mask is fixed, the intermediate values (e.g., S-box outputs) depend only on the key and plaintext, eliminating the interference of randomness on the Hamming weight distribution. Due to these operations, these traces have a higher SNR, making leakage easier to detect. According to the Hamming weight leakage model, the traces are divided into two subsets: $HW(\text{label}) > 4$ and $HW(\text{label}) < 4$. Although this method assumes that the attacker can control the mask value (inconsistent with real-world attacks), its goal is to provide a baseline reference to locate leakage points. In practical scenarios, attackers can use this information to focus on attacking weak points in the protection of the RP mask.

At the same time, in the selection of labels, in addition to the S_{in} and S_{out} of SBox, a new label, S_0, is added. The reason is as follows. Under the RP masking scheme, the output value of the S-box in the first round of AES-128, $S_{\text{out}} = SBox(p_j \oplus k_j)$, can be considered as divided into two shares, S_0 and S_1. S_0 represents the value XORed with the mask, and S_1 represents the mask. Since the mask is set to 0, the second share can be represented as $S_1 = SBox(0) = 99$. Therefore, the first share can be calculated by formula (9) as follows.

$$S_0 = \text{SBox}(p_j \oplus k_j) \oplus 99 \tag{9}$$

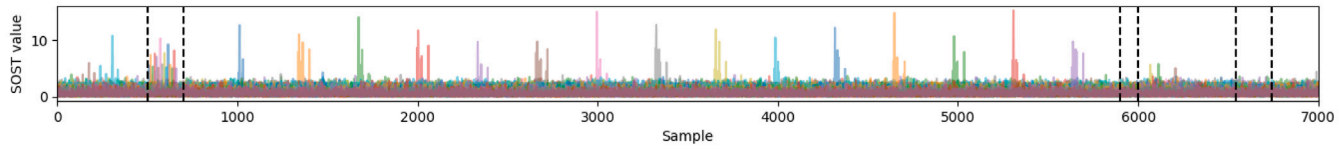


(a) The t-test results when the label is S_{in}

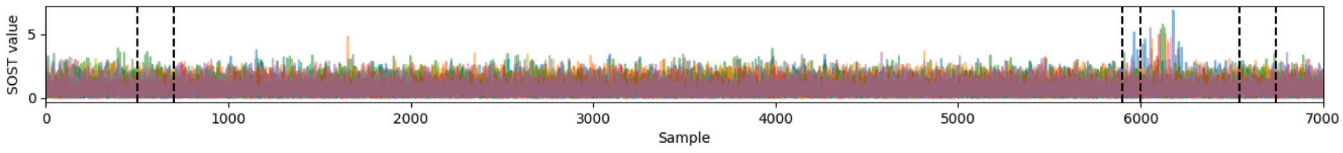


(b) The t-test results when the label is S_{out}

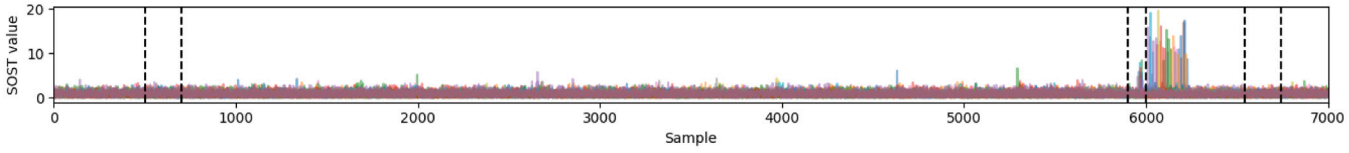
Fig. 7. The t-test results of 5000 traces collected from the D1 device at a distance of 10 cm under the RP masking scheme with random masking.



(a) The t-test results when the label is S_{in}



(b) The t-test results when the label is S_{out}



(c) The t-test results when the label is S_0

Fig. 8. The t-test results of 5000 traces collected through a coaxial cable (at 0 cm) under the RP masking scheme with the mask fixed to 0.

Therefore, S_0 is known and can be used as a label. In this case, we discover significant leakage.

The leakage evaluation is shown in Figs. 8. From the top subplots of Figs. 8, it can be observed that when the label is S_{in}, the Addroundkey operation in the first round's Addroundkey and the SubBytes operations of the SBox in the second round are both leaked, and the leakage is obvious, with a value of about 7–8. From the middle subplots of Figs. 8, it can be seen that when the label is S_{out}, the ShiftRows and MixColumns in the second round have a slight leakage, but the value is about 5. As the number of traces increases, the leakage is expected to become more pronounced. From the bottom subplots of Figs. 8, it can be seen that when the label is S_0, both the ShiftRows and MixColumns operations have very obvious leakage, with a value of up to 20. This shows that the wireless electromagnetic side channel attack has a strong potential. The traces collected from a distance through the wireless side channel can obtain a large amount of key-related information, and this zero-contact attack method seems to be more realistic.

Fixing the mask to 0 and applying averaging are both aimed at improving the signal-to-noise ratio of the traces, allowing for a better determination of the leakage interval. From the above observations, it can be concluded that with 5000 traces, the traces at 10 cm with random mask protection do not show significant leakage. However, the traces with the mask fixed to 0 and averaged 100 times show a clear leakage interval. We boldly speculate that when the number of traces increases indefinitely, even with RP random mask protection, leakage still occurs at these same positions. In other words, when the number of traces is insufficient, the leakage in these areas can be considered negligible.

Based on the above results, we have identified four attack points (trace segments), specifically the trace segment of Addroundkey in the first round within the interval [491:663], the trace segment of SubBytes in the second round within the interval [1000:1080], the trace segment of ShiftRows within the interval [5958:5996], and the trace segment of MixColumns within the interval [5999:6535]. These are the four attack points we have finalized. Please note that the clean traces of the coaxial cable are only used for leakage analysis detection.

5. Experimental method

By performing a t-test leakage assessment on the traces with the mask fixed to 0, we identify four specific leakage intervals. Afterwards, we collect trace sets data T1 and T2 from target devices d1 and d2 at a distance of 10 cm. During encryption, the plaintext is randomly generated $P_j \in (0,1)^{128}$ while the key is fixed $K \in (0,1)^{128}$. For the

Table 1
Summary of Model 1 Architecture.

Layer type	Out shape
Batch Normalization	(None, 172) (S1); (None, 38) (S2) (None, 38) (S3); (None, 536) (S4)
Dense	(None, 256)
Batch Normalization	(None, 256)
Dropout	(None, 256)
Dense	(None, 512)
Batch Normalization	(None, 512)
Dropout	(None, 512)
Output (Dense)	(None, 256)

Table 2
Summary of Model 2 Architecture.

Layer type	Out shape
Batch Normalization	(None, 172) (S1); (None, 38) (S2) (None, 38) (S3); (None, 536) (S4)
Dense	(None, 256)
Batch Normalization	(None, 256)
Dense	(None, 512)
Batch Normalization	(None, 512)
Dense	(None, 1024)
Batch Normalization	(None, 1024)
Dropout	(None, 1024)
Dense	(None, 512)
Batch Normalization	(None, 512)
Dropout	(None, 512)
Output (Dense)	(None, 256)

kth byte of the 16-byte plaintext P_j, it is denoted as $P_{j,k}$, where $K \in \{0, 1, \ldots, 15\}$. We conduct WSCAs based on deep learning on these four leakage points, and compare the attack effects of each leakage point to find the appropriate attack point and attack method. Additionally, we compare the results by utilizing methods such as ensembling to enhance the attack effectiveness.

5.1. Model training

During the experiment, we collected 300k trace data T1 at 10 cm from the d1 device and 200k trace data T2 at 10 cm from the d2 device. We divided the trace data T1 from the d1 device into a training set $T1_{\text{train}}$ (250k) and a test set $T1_{\text{test}}$ (50k) in a ratio of 5:1. The 200k trace data T2 is entirely used as a cross-device test set to evaluate the model's generalization ability and the effectiveness of cross-device attacks.

During the model training process, we use the training set $T1_{\text{train}}$ (250k) to train the model, of which 20% is used for verification. The specific trace segments used for training are $\mathcal{S}1$, $\mathcal{S}2$ [1000:1080], $\mathcal{S}3$ [5958:5996], and $\mathcal{S}4$ [5999:6535], which correspond to the side-channel leakages generated by the encryption operations of Addroundkey, SubBytes, ShiftRows, and MixColumns, respectively. When selecting training labels, we used the label S_{in} for $\mathcal{S}1$ [491:663] and $\mathcal{S}2$ [1000:1080] and the label S_{out} for $\mathcal{S}3$ [5958:5996] and $\mathcal{S}4$ [5999:6535]. This is because, although the label S_0 showed higher leakage for the ShiftRows and MixColumns operations under t-test detection, the mask was no longer fixed to that point, making S0 unsuitable as a label for model training.

Additionally, since our target of attack is protected not only by random masking (RP) but also by radio electromagnetic signal collection at a distance of 10 cm, there is a tendency for overfitting during training. To mitigate this, we implemented a learning rate decay mechanism, which reduces overfitting without significantly decreasing accuracy. After extensive parameter tuning and training, we ultimately selected two $\mathcal{MLP}$ model architectures that were most suitable for this protection mechanism, referred to as $\mathcal{MLP}1$ and $\mathcal{MLP}2$. The model architectures are shown in Tables 1 and 2.

Therefore, we used the leakage trace segments $\mathcal{S}1$, $\mathcal{S}2$, $\mathcal{S}3$, and $\mathcal{S}4$ generated by the four basic encryption operations to conduct model training under the $\mathcal{MLP}1$ and $\mathcal{MLP}2$ architectures, with label selection for each trace segment as described above. The initial learning rate was set to 0.0008, and the training process lasted for 100 epochs, ultimately resulting in the generation of 8 trained models. The target of the attack was the second byte of the subkey. It should be noted that the leakage interval for the second round of SubBytes is quite long. Since we are attacking the second byte of the subkey, to avoid wasting computational resources, we only intercept the interval corresponding to the leakage of the second byte.

The selection of two types of model architectures (one deep, one shallow) is aimed at better evaluating the attack effectiveness of these four attack points based on deep learning. Generally speaking, the higher the leakage, the more effective the side-channel attack. However, under RP random mask protection, conducting t-test examinations reveals no leakage in the device. Although we can observe leakage locations when the mask is fixed to 0, we cannot conclude the extent of the leakage based on the t-test results (Figs. 8). In actual attacks, when dealing with devices protected by RP random masks based on additive chains, the experimental results of this paper will offer suggestions on which specific attack points and methods to choose for side-channel attacks. This is also why we chose the same deep learning model architecture to train models for the four segments.

5.2. Attack phase

In the study of side channel attacks, Guessing Entropy (GE, A metric quantifying side-channel attack efficiency via the average guesses needed to recover the key.) is a widely used and recognized evaluation metric. It indicates how many attempts an attacker needs to make on average to successfully recover the correct key, and is used to measure the difficulty of the attack. A lower guessing quotient means that the attacker can find the correct key faster and the attack is more likely to succeed.

When performing a side channel attack, assign a probability to each possible key and sort these probabilities in descending order. The position of the correct key k^* in this order is its rank. Since the attack in this paper is on subkeys, each subkey K_j should be guessed separately and the Partial Guess Entropy (PGE, An extension of Guessing Entropy (GE) that quantifies the weighted average number of guesses required to recover the correct key in side-channel attacks, given the attacker's knowledge of the key probability distribution.), rather than GE, should be used as the estimation metric [29]. The ranking of subkey K_j is shown in formula (10).

$$\text{Rank}(K_j; \mathcal{T}) = \left| \left\{ K'_j \in \mathcal{K} : \Pr[K_j \mid \mathcal{X}, \mathcal{T}] < \Pr[K'_j \mid \mathcal{X}, \mathcal{T}] \right\} \right| \tag{10}$$

$\mathcal{K}$ represents the set of all possible keys, $\Pr[K_j \mid \mathcal{X}, \mathcal{T}]$ represents the probability of the key K_j given the side-channel information $\mathcal{X}$ and the guess set $\mathcal{T}$. And the ranking of key K_j is the number of keys that are more likely than it. Its Empirical Guess Entropy (EGE) is as shown in formula (11).

$$\text{GE} = \frac{1}{N} \sum_{q=1}^{N} \text{Rank}(K^*; \mathcal{T}_q) \tag{11}$$

where N is the number of trials, $\text{Rank}K^*; \mathcal{T}_q$ is the correct ranking of the qth time. EGE is a statistical summary of Rank, used to represent the average difficulty of guessing. In the above, we used two different $\mathcal{MLP}$ model architectures for model training for four different leakage intervals. After training eight models, we let them calculate the ranking of the model for the correct key on the T1 test set collected from the d1 device and the T2 test set collected from the d2 device, and obtained the average number of traces required in the attack, in order to evaluate the effectiveness of different leakage intervals and the attack performance under two model architectures. During the attack,

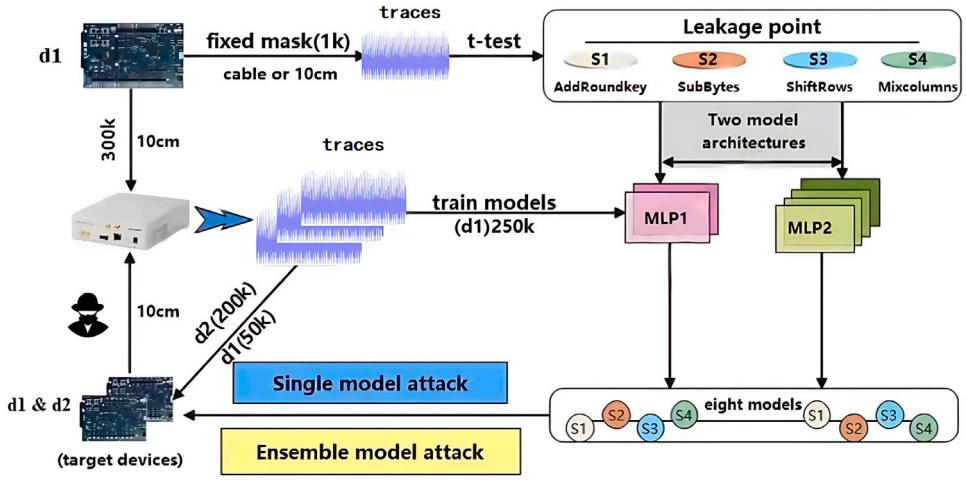


Fig. 9. Entire experimental process of wireless side-channel attack on devices protected by RP masking scheme.

the smaller the average number of traces required, the more effective the attack method. The entire experimental process is shown in Fig. 9.

At the same time, in order to compare with the results in [14], most of our results do not use the PGE to obtain the average number of traces required for the attack, but use the number of termination traces where the average guessing quotient of the true subkey reaches 0 in most experiments as the number of traces to evaluate the attack effect. When performing rank evaluation, we conduct 150 experiments to get the average. In each experiment, the traces in the test set are randomly shuffled, and the trained model is used to estimate the probability of each key for the given traces. The probabilities are then ranked. If in more than half of the experiments the correct key ranks as 0, we consider the attack successful. The number of traces at this time is regarded as the number of traces required for the attack.

5.3. Model ensembling

In addition to performing individual side-channel attacks on each leakage interval, we also combine models trained on leakage intervals with similar attack effectiveness into ensembles. Model ensemble typically enhances robustness and accuracy, reducing potential classification errors of individual models by combining different models. We will integrate models trained on these four leakage intervals, the models trained on $S1$, $S2$, $S3$, and $S4$, the ensemble can improve prediction accuracy.

In reality, the traces collected for training the model and the traces finally collected from the victim device generally cannot come from exactly the same source. In this case, the attack is more difficult. The idea of ensemble is to find the best idea and find the best method to improve the effect of side channel attacks.

6. Experimental results

This section describes the attack results of eight models trained under four leakage intervals and two different model architectures, including attacks on the original device and across devices. Through this comprehensive evaluation, we can understand the performance of models trained under different leakage intervals, their generalization ability to unseen devices, and their overall ability to exploit side channel information in different environments.

Table 3
Number of traces required for each of the eight models trained under the two architectures to successfully recover the key by attacking device d1.

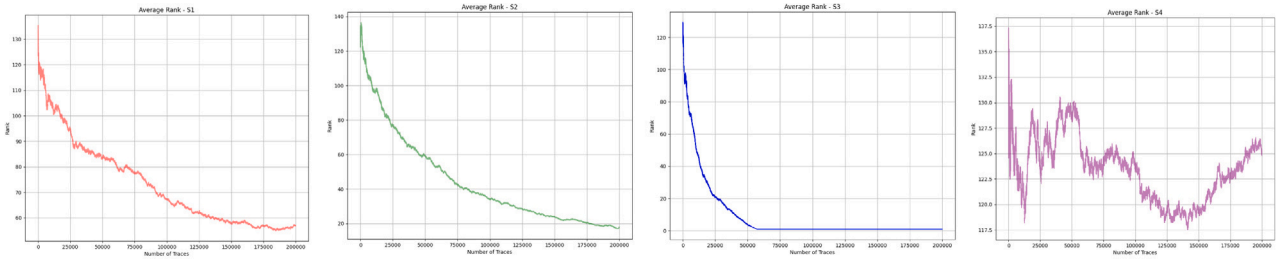
Leakage	Model	Traces
S1 (491–663)	$\mathcal{MLP}1$	39
	$\mathcal{MLP}2$	7
S2 (1000–1080)	$\mathcal{MLP}1$	28
	$\mathcal{MLP}2$	8
S3 (5958–5996)	$\mathcal{MLP}1$	22
	$\mathcal{MLP}2$	5
S4 (5999–6535)	$\mathcal{MLP}1$	20
	$\mathcal{MLP}2$	6

6.1. Results on d1

Although the target device d1 is protected by a mask based on an addition chain (RP mask scheme), the same-device attack is relatively simple. Only a few to dozens of trajectories are needed to successfully recover the key. Table 3 shows the number of traces of the model attacking the original device d1 under the deep learning model architectures $\mathcal{MLP}1$ and $\mathcal{MLP}2$.

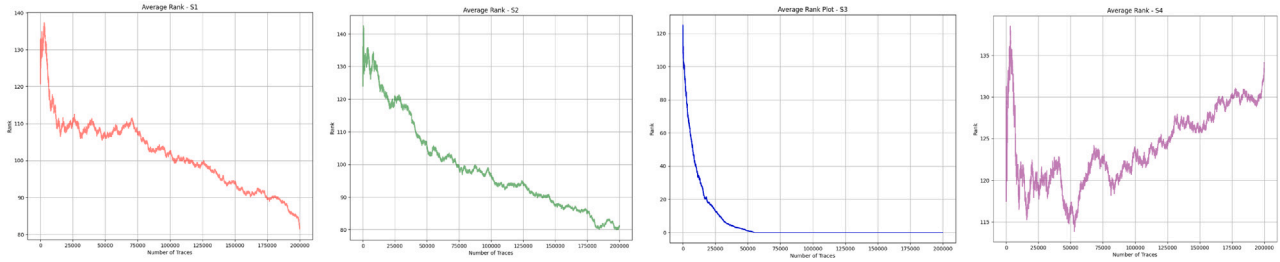
From the perspective of model architecture, Table 3 shows that when conducting wireless side channel attacks (collected at 10 cm), the model trained under the $\mathcal{MLP}2$ model architecture is more suitable for attacking AES encryption with the RP mask scheme protection, and only less than ten traces are needed to recover the key. Although the $\mathcal{MLP}1$ model architecture can also attack successfully, it requires 20–40 traces. In the case of protection, models with more neurons and deeper layers can better grasp the potential association between the key and the trace, but they also learn some useless features and lead to overfitting. The $\mathcal{MLP}2$ "small-large-small" model structure can reduce overfitting while learning more essential connections and features between the key and the side channel leakage compared to the shallower $\mathcal{MLP}1$ model architecture.

Based on the number of traces required for successful attacks in each leakage interval, the $S3$ and $S4$ leakage intervals resulting from the ShiftRows and MixColumns operations show better attack outcomes. As noted in Section 4, with a fixed mask value of 0, the leakage positions and effects vary. The t-test result shows the best results at 10 cm, specifically for the MixColumns operation with the label S_0 at the $S4$ interval. However, since the mask cannot be fixed in a real attack and is inevitably random, this label cannot be used. For other labels, the leakage from ShiftRows and MixColumns operations is not



(a) Addroundkey (S1) **(b)** SubBytes (S2) **(c)** ShiftRows (S3) **(d)** MixColumns (S4)

Fig. 10. PGE results of four models trained under the $\mathcal{MLP}1$ Architecture.



(a) Addroundkey (S1) **(b)** SubBytes (S2) **(c)** ShiftRows (S3) **(d)** MixColumns (S4)

Fig. 11. PGE results of four models trained under the $\mathcal{MLP}2$ Architecture.

significant or even noticeable. The attack results for each segment do not quite align with the leakage findings, which might be related to whether the mask is fixed. We hypothesize that when the RP mask is random, during the AES encryption process under this masking scheme, the truly vulnerable points for side-channel attacks are not the conventional points such as the input or output of the SBox, but rather the information leakage from the ShiftRows or MixColumns operations.

6.2. Results on cross device attack (d2)

In [14], the authors selected the leakage interval $S4$ generated by the column obfuscation operation for the attack when studying how to break through the wireless side channel attack with RP mask protection. In [14], two strategies are adopted. One strategy is a single-step strategy, where both share0 and share1, which are the parts of the intermediate value divided under the RP mask scheme, are separately input into the network for training, and then perform XOR on the results to obtain the intermediate value of the key. The other strategy is to throw a whole segment directly into the network for training, and the deep learning model directly completes the d+1 order attack internally. However, considering the attack results, the first strategy requires about 290k traces to successfully attack the device, while the second strategy cannot achieve a successful attack even with 300k traces, only lowering the rank to around 22. Similarly, the attackers also conduct attacks on other devices protected with RP masking at the 0 cm position (via coaxial cable). The first strategy required approximately 192k traces to succeed, while the second strategy needed about 168k traces. It can be seen that under the defense of random mask based on addition chain, the difficulty of cross-device attack can be seen, and long-distance collection is even more difficult. Fig. 10 shows the attack results using these four leakage intervals for the attack under the $\mathcal{MLP}1$ model architecture. Fig. 11 shows the attack results using these four leakage intervals as attack points under the $\mathcal{MLP}2$ model architecture.

In Figs. 10 and 11, the cross-device test set T2 consists of 200k traces. From the attack result charts, it can be seen that when performing cross-device attacks, when using the models trained with $S1$, $S2$, and $S4$ under the two $\mathcal{MLP}$ model architectures $\mathcal{MLP}1$ and $\mathcal{MLP}2$, the key cannot be successfully recovered and sensitive information cannot be obtained under 200k traces. However, when using the model trained with the $S3$ trace segment to attack, the key is not only successfully recovered, but also the attack efficiency is greatly improved compared with the researchers in [14]. Under the $\mathcal{MLP}1$ model architecture, although the rank of key recovery cannot be directly reduced to 0 (that is, the attacker directly finds the correct key), when the number of traces reaches 55k, the rank of the key can drop to 1. In other words, in this case, the attacker only needs to make two guesses, the first guess is the key ranked 0, and if it is incorrect, the second guess is the key ranked 1. Under the $\mathcal{MLP}2$ model architecture, we can rank the correct key at the top with around 49k traces, successfully obtaining the key. Compared with [14], we have greatly reduced the number of traces required for the attack and improved the attack efficiency by 83.1%. The specific attack situation is shown in Table 4.

Table 4
Number of traces required for each of the eight models trained under the two architectures to successfully recover the key by attacking d2 device (at 10 cm).

Leakage	Model	# Traces
S1 (491–663)	$\mathcal{MLP}1$	PGE=57/200K
	$\mathcal{MLP}2$	PGE=82/200K
S2 (1000–1080)	$\mathcal{MLP}1$	PGE=18/200K
	$\mathcal{MLP}2$	PGE=81/200K
S3 (5958–5996)	$\mathcal{MLP}1$	PGE=1/55K
	$\mathcal{MLP}2$	49 089
S4 (5999–6535)	$\mathcal{MLP}1$	—
	$\mathcal{MLP}2$	—

As can be seen from Table 4, when using the wireless side channel attack method to attack the device d2 protected by the RP mask scheme at 10 cm, among the 200k traces, only the model trained by the trace segment $S3$ generated by the ShiftRows operation can successfully recover the key and only 49,089 traces are required. The trace segments generated by other encryption operations including Addroundkey, SubBytes, and MixColumns cannot be successful under the two $\mathcal{MLP}$ model architectures. The absence of specific ranking results for segment $S4$ in Table 4 stems from the failure of our attack on device d2 under these two model architectures within 200k traces. As shown in Figs. 10 and 11, the rank curve exhibits no progressive descent, indicating no concrete key recovery results were achieved.

From the t-test results in Section 4.2 (Figs. 7 and 8), when the number of traces is 5k and the mask is fixed to 0, all four operations exhibit varying degrees of information leakage. In contrast, when the mask is random, no leakage is observed. Based on the t-test results with a fixed mask, we identified four potential leakage segments. Using the same two model architectures and appropriate labels, we performed wireless side-channel attacks on the potential leakage segments generated by these four operations. However, in actual cross-device attacks, the results from models trained under both architectures indicate that only the wireless side-channel information from the ShiftRows operation is of relatively high quality. Under the protection of the RP masking scheme, the other operations exhibit almost no information leakage.

This is because the core design goal of the RP masking scheme is to use additive chain masking techniques to focus on protecting nonlinear cryptographic operations (such as the SubBytes operation in AES). Such nonlinear operations are often the critical breakthrough points in traditional side-channel attacks, as their input–output relationships are complex and tend to produce significant Hamming weight variations. The RP scheme employs multiple layers of random masks to algebraically recompose the S-box computation process, making it difficult for attackers to recover key information through first-order statistical means. However, the scheme provides relatively weak protection for linear operations (such as ShiftRows and MixColumns), as it assumes that linear transformations themselves do not introduce additional leakage risks.

The particularity of the ShiftRows operation lies in its mere rearrangement of positions, essentially being a purely linear operation. When the intermediate value carrying the mask enters ShiftRows, the mask propagates along with the byte positions, but its randomness distribution characteristics remain unchanged. For example, after the mask value m passes through ShiftRows, it becomes ShiftRows(m), and the intermediate value $v \oplus m$ becomes ShiftRows(v) $\oplus$ ShiftRows(m). Since the permutation operation does not break the linear superposition relationship between the mask and the key, attackers can directly establish a statistical correlation with the key by analyzing the Hamming weight distribution (such as HW(ShiftRows(v) $\oplus$ ShiftRows(m))) in a fixed time segment.

In contrast, other linear operations (such as MixColumns), though also limited by the RP masking protection assumptions, introduce more complex mask diffusion effects during execution. MixColumns, through matrix multiplication, diffuses a single mask byte into multiple output bytes, significantly reducing the signal-to-noise ratio (SNR) of the leakage signal. For example, after the input mask m passes through MixColumns, it becomes a linear combination of multiple mask components, causing the Hamming weight distribution to approach randomness. Experimental data (Table 4) show that even with 200k traces, attacks targeting MixColumns ($S4$ segment) cannot reduce the guessing entropy, indicating that the leakage quality is insufficient for effective key recovery. This phenomenon is consistent with the conclusions in [14]—only when the attack method is highly customized (such as single-step key guessing), can MixColumns possibly be broken, but the required number of traces (300k) far exceeds the practical feasibility boundary for attacks.

It is worth noting that although the AddRoundKey operation shows leakage under fixed mask conditions (Fig. 7), its actual attack effectiveness is limited by the round-wise mask update mechanism. The RP scheme regenerates the mask in each encryption round, preventing the instantaneous leakage of AddRoundKey from forming a statistical accumulation effect across rounds. Therefore, although the t-test detects potential risks, it is still impossible to recover the key through this operation in practical attacks (Table 4). This further demonstrates the RP scheme's core protection advantage for nonlinear operations and the difference in protection effectiveness for linear operations—ShiftRows, due to its extremely simple operational characteristics, becomes the weak link in the protection chain.

Table 5
Number of traces required for successful key recovery on device d1 after model ensembling trained under the $\mathcal{MLP}1$ architecture.

Ensembling	Leakage	#Traces
Two-Ensemble	S1S2	19
	S1S4	13
	S2S4	13
Three-Ensemble	S1S2S4	10
Four-Ensemble	S1S2S3S4	7

Table 6
Number of traces required for successful key recovery on device d1 after model ensembling trained under the $\mathcal{MLP}2$ architecture.

Ensembling	Leakage	#Traces
Two-Ensemble	S1S2	3
	S1S4	3
	S2S4	4
Three-Ensemble	S1S2S4	3
Four-Ensemble	S1S2S3S4	2

Therefore, in AES encryption with the RP random masking scheme, the true vulnerability and optimal attack point in wireless side-channel attacks should be the leakage generated by the second-round ShiftRows operation, i.e., the $S3$ leakage segment. Experiments demonstrate that the $\mathcal{MLP}2$ model, with its deeper architecture and higher neuron density, significantly enhances the ability to capture non-linear correlations between leakage information and the key, recovering the key with only 49k traces. In contrast, the simpler $\mathcal{MLP}1$ model reduces the rank to 1 with merely 5.5k traces in the same leakage segment, further confirming the weak protection of ShiftRows—the strong key-leakage correlation even allows efficient attacks by shallow models. For other operations (e.g., AddRoundKey, MixColumns), however, the leakage quality degrades drastically due to the complex diffusion effects of RP masking, rendering both model architectures incapable of extracting sensitive information even with 200k traces.

6.3. Ensemble attacks

Among the individual models, the most effective one is clearly the model under the $\mathcal{MLP}2$ architecture trained on $S3$. The other three segments show almost no effective attack capability. Therefore, we integrated the models trained on $S1$, $S2$, and $S4$ to explore whether this integration could provide better attack performance and potentially surpass the single deep learning model trained on $S3$. Our primary focus was on integrating trace segments that could not successfully recover the key, using two ensemble methods: two-model ensemble and three-model ensemble. Additionally, we combined all four trace segments into a four-model ensemble to evaluate the impact of the other three segments on the $S3$ segment.

As can be seen from Tables 5 and 6, when attacking the profiling device d1, integrating the three models with slightly inferior effects can improve their attack efficiency and reduce the number of traces required for the attack. In the case of four ensemble, the models trained on the remaining three stages do not drag down the model trained in the $S3$ stage with the best attack effect, but instead further improves the overall attack effectiveness.

When conducting cross-device attacks, the challenge increases significantly. Except for the model trained in $S3$ segment, no other single model can recover the key within 200k traces. Therefore, we need to evaluate whether the other three segments can have a positive impact on the model trained in the $S3$ segment. The results are shown in Tables 7 and 8.

As can be seen from the Tables 7 and 8, during cross-device attacks, the remaining three segments still cannot successfully recover

Table 7
Number of traces required for successful cross-device d2 key recovery after model ensembling trained under the $\mathcal{MLP}1$ architecture.

Ensembling	Leakage	#Traces
Two-Ensemble	S1S2	PGE=21/200K
	S1S4	PGE=70/200K
	S2S4	PGE=47/200K
Three-Ensemble	S1S2S4	PGE=38/200K
Four-Ensemble	S1S2S3S4	PGE=23/200K

Table 8
Number of traces required for successful cross-device d2 key recovery after model ensembling trained under the $\mathcal{MLP}2$ architecture.

Ensembling	Leakage	#Traces
Two-Ensemble	S1S2	PGE=74/200K
	S1S4	—
	S2S4	—
Three-Ensemble	S1S2S4	PGE=94/200K
Four-Ensemble	S1S2S3S4	PGE=55/200K

the key within 200k traces after integration, and there is almost no improvement in the attack effect. Even if integrated with the model trained in the S3 segment, it still cannot succeed. This shows that other models are interfering with its correct judgment of the key. In other words, when conducting cross-device attacks, S3 (the second round of row shift operations) is the most suitable attack point because it can most effectively reveal the core relationship between the key and the side-channel information.

7. Conclusions

We demonstrate a more successful deep learning wireless side channel attack than the first one against AES-128 implementation with RP masking. By performing a t-test on traces with fixed zero mask, four different leakages are found. Then, by attacking these four segments with the MLP model, we conclude that the models trained on these four segments can successfully recover the key when attacking the profiling device, and ensembling can improve the attack efficiency. However, when attacking across devices, the leakage generated by the second round of ShiftRows operations is the real suitable attack point for WSCAs on AES with RP masking. The other leakage segments are not effective when attacking across devices, which is far from the $S3$ segment, and ensembling cannot bring more effect. We can successfully recover the key from the random mask trace within 50k traces at a distance of 10 cm from the victim device. This is an 83.1% improvement in attack efficiency compared to the first study [14].

In the future, more effective ways to disrupt Bluetooth devices with higher-order masking protection at longer distances may be considered.

CRediT authorship contribution statement

Bijia Cao: Literature review, Conceptualization, Data acquisition, Analysis and curation, Writing – original draft. **Huanyu Wang:** Supervision, Methodology, Writing – review & editing. **Tuo Deng:** Data analysis, Result interpretation. **Dalin He:** Data curation, Visualization. **Zitian Huang:** Literature review, Investigation. **Junnian Wang:** Supervision, Methodology, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Dalin He reports financial support was provided by Hunan Province. Wangdong Zeng reports financial support was provided by National Natural Science Foundation of China. Wangdong Zeng reports financial support was provided by Hunan Scientific Committee. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the research grant 2025AQ2024 (the Key R&D Program of Hunan Province) from the Department of Science and Technology of Hunan Province.

Data availability

Data will be made available on request.

References

[1] B. Nassi, O. Vayner, E. Iluz, D. Nassi, J. Jancar, D. Genkin, E. Tromer, B. Zadov, Y. Elovici, Optical cryptanalysis: Recovering cryptographic keys from power LED light fluctuations, in: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, CCS, 2023, pp. 268–280.
[2] P. Kocher, J. Jaffe, B. Jun, Differential power analysis, in: Advances in Cryptology—CRYPTO'99, Springer, 1999, pp. 388–397.
[3] D. Das, A. Golder, J. Danial, S. Ghosh, A. Raychowdhury, S. Sen, X-DeepSCA: Cross-device deep learning side channel attack, in: ACM/IEEE Design Automation Conference, DAC, 2019, pp. 1–6.
[4] Y. Ji, E. Dubrova, A side-channel attack on a masked hardware implementation of CRYSTALS-Kyber, in: Proceedings of the 2023 Workshop on Attacks and Solutions in Hardware Security, ASHES, 2023, pp. 27–37.
[5] K. Kuroda, Y. Fukuda, K. Yoshida, T. Fujino, Practical aspects on non-profiled deep-learning side-channel attacks against AES software implementation with two types of masking countermeasures including RSM, in: Proceedings of the 5th Workshop on Attacks and Solutions in Hardware Security, ASHES, 2021, pp. 29–40.
[6] E. Cagli, C. Dumas, E. Prouff, Convolutional neural networks with data augmentation against jitter-based countermeasures: Profiling attacks without pre-processing, in: International Workshop on Cryptographic Hardware and Embedded Systems, CHES, Springer, 2017, pp. 45–68.
[7] S. Picek, I. Samiotis, J. Kim, A. Heuser, S. Bhasin, A. Legay, On the performance of convolutional neural networks for side-channel analysis, in: Int. Conf. on Security, Privacy, and Applied Crypt. Engineering, Springer, 2018, pp. 157–176.
[8] R. Benadjila, E. Prouff, R. Strullu, E. Cagli, C. Dumas, Deep learning for side-channel analysis and introduction to ASCAD database, J. Cryptogr. Eng. 10 (2) (2020) 163–188.
[9] Z. Hong-Yi, G. Da-Wu, C. Pei, Q. Shi-Pei, L. xiao Wei, Wireless side-channel analysis method based on spectral addition, J. Cryptologic Res. 10 (4) (2023) 862–878.
[10] G. Camurati, S. Poeplau, M. Muench, T. Hayes, A. Francillon, Screaming channels: When electromagnetic side channels meet radio transceivers, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018, pp. 163–177.
[11] G. Camurati, A. Francillon, F.-X. Standaert, Understanding screaming channels: From a detailed analysis to improved attacks, IACR Trans. Cryptogr. Hardw. Embed. Systems (TCHES) (2020) 358–401.
[12] R. Wang, H. Wang, E. Dubrova, Far field EM side-channel attack on AES using deep learning, in: Proceedings of the 4th ACM Workshop on Attacks and Solutions in Hardware Security, ASHES, 2020, pp. 35–44.
[13] R. Wang, H. Wang, E. Dubrova, M. Brisfors, Advanced far field EM side-channel attack on AES, in: Proceedings of the 7th ACM on Cyber-Physical System Security Workshop, 2021, pp. 29–39.
[14] H. Wang, Amplitude-modulated EM side-channel attack on provably secure masked AES, J. Cryptogr. Eng. (2024) 1–13.
[15] M. Rivain, E. Prouff, Provably secure higher-order masking of AES, in: International Workshop on Cryptographic Hardware and Embedded Systems, Springer, 2010, pp. 413–427.
[16] P. FIPS, 197: Advanced encryption standard (AES), Natl. Inst. Stand. Technol. 26 (2001).
[17] B. Schneier, Applied cryptography protocols, Algorithms Source Code C (1995).
[18] J.-S. Coron, E. Prouff, M. Rivain, Side channel cryptanalysis of a higher order masking scheme, in: Cryptographic Hardware and Embedded Systems-CHES 2007: 9th International Workshop, Vienna, Austria, September 10-13, 2007. Proceedings 9, Springer, 2007, pp. 28–44.

[19] Y. Ishai, A. Sahai, D. Wagner, Private circuits: Securing hardware against probing attacks, in: Advances in Cryptology-CRYPTO 2003: 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003. Proceedings 23, Springer, 2003, pp. 463–481.
[20] Z. Martinasek, V. Zeman, Innovative method of the power analysis, Radioengineering 22 (2) (2013) 586–594.
[21] H. Maghrebi, T. Portigliatti, E. Prouff, Breaking cryptographic implementations using deep learning techniques, in: Security, Privacy, and Applied Cryptography Engineering: 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings 6, Springer, 2016, pp. 3–26.
[22] M. Jin, M. Zheng, H. Hu, N. Yu, An enhanced convolutional neural network in side-channel attacks and its visualization, 2020, arXiv preprint arXiv:2009.08898.
[23] I. Goodfellow, Deep learning, 2016.
[24] D.P. Kingma, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.
[25] J. Konečnỳ, H.B. McMahan, D. Ramage, P. Richtárik, Federated optimization: Distributed machine learning for on-device intelligence, 2016, arXiv preprint arXiv:1610.02527.
[26] R.M. Secareanu, S. Warner, S. Seabridge, C. Burke, T.E. Watrobski, C. Morton, W. Staub, T. Teilier, E. Friendman, Physical design to improve the noise immunity of digital circuits in a mixed-signal smart-power system, ISCAS, in: 2000 IEEE International Symposium on Circuits and Systems, vol. 4, IEEE, 2000, pp. 277–280.
[27] S. Bronckers, G. Van der Plas, Y. Rolain, G. Vandersteen, Substrate Noise Coupling in Analog/RF Circuits, Artech House, 2010.
[28] P. Juszczak, D. Tax, R.P. Duin, Feature scaling in support vector data description, in: Proc. Asci, Citeseer, 2002, pp. 95–102.
[29] H. Pahlevanzadeh, J. Dofe, Q. Yu, Assessing CPA resistance of AES with different fault tolerance mechanisms, in: 2016 21st Asia and South Pacific Design Automation Conference, ASP-DAC, IEEE, 2016, pp. 661–666.