ADVANCED LOW-COST ELECTRO-MAGNETIC AND MACHINE LEARNING SIDE-CHANNEL ATTACKS

A Thesis

Submitted to the Faculty

of

Purdue University

by

Josef A. Danial

In Partial Fulfillment of the

Requirements for the Degree

of

MASTER OF SCIENCE IN ELECTRICAL AND COMPUTER ENGINEERING

December 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL

STATEMENT OF THESIS APPROVAL

ACKNOWLEDGMENTS

First, I would like to thank my advisor Shreyas Sen for his insight and guidance in and out of academics.

I would also like to specifically thank Debayan Das for his constant support, encouragement, and willingness to help with any problem.

Additionally, I would like to thank all the members of the SPARC Lab for their help over the past two years.

Finally, I would like to thank my family for their support throughout my entire education.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABBREVIATIONS

SCA	side-channel analysis
ML	machine learning
MSE	Mean Square Error
TVLA	test vector leakage assessment
MTD	Minimum Traces to Disclosure
CPA	Correlation Power Analysis
CEMA	Correlation EM Analysis
DPA	Differential Power Analysis
DEMA	Differential EM Analysis
DNN	Deep Neural Network
CNN	Convolutional Neural Network
SNR	signal to noise ratio
AES	advanced encryption standard
DES	Data Encryption Standard
RSA	Rivest–Shamir–Adleman
PCA	Principle Component analysis
POI	Point of Interest
LDA	Linear Discriminant Analysis
FFT	Fast Fourier Transform
HW	Hamming Weight
HD	Hamming Distance
HLS	High Level Synthesis

ABSTRACT

Danial, Josef A. MS, Purdue University, December 2020. Advanced Low-Cost Electro-Magnetic and Machine Learning Side-Channel Attacks. Major Professor: Shreyas Sen.

Side-channel analysis (SCA) is a prominent tool to break mathematically secure cryptographic engines, especially on resource-constrained devices. SCA attacks utilize physical leakage vectors like the power consumption, electromagnetic (EM) radiation, timing, cache hits/misses, that reduce the complexity of determining a secret key drastically, going from 2^{128} for brute force attacks to 2^{12} for SCA in the case of AES-128. Additionally, EM SCA attacks can be performed non-invasively without any modifications to the target under attack, unlike power SCA. To develop defenses against EM SCA, designers must evaluate the cryptographic implementations against the most powerful side-channel attacks. In this work, systems and techniques that improve EM side-channel analysis have been explored, making it lower-cost and more accessible to the research community to develop better countermeasures against such attacks. The first chapter of this thesis presents SCNIFFER, a platform to perform efficient end-to-end EM SCA attacks. SCNIFFER introduces leakage localization – an often-overlooked step in EM attacks – into the loop of an attack. Following SCNIFFER, the second chapter presents a practical machine learning (ML) based EM SCA attack on AES-128. This attack addresses issues dealing with low signal-to-noise ratio (SNR) EM measurements, proposing training and pre-processing techniques to perform an efficient profiling attack. In the final chapter, methods for mapping from power to EM measurements, are analyzed, which can enable training a ML model with much lower number of encryption traces. Additionally, SCA evaluation of high-level synthesis (HLS) based cryptographic algorithms is performed, along with the study of futuristic neural encryption techniques.

1. INTRODUCTION

Side-channel analysis allows adversaries to recover secret information using far less processing than brute force attacks by taking advantage of additional information from a side-channel such as execution time, power consumption, electromagnetic emissions, or even sound. These side channels give attackers information based on the implementation of a cryptographic algorithm, and can be used to efficiently recover secrets even if the cryptographic algorithm itself is secure. This thesis focuses on attacks using the electromagnetic side-channel, and the closely related power side-channel. In order to create effective defenses against side-channel attacks, designers must subject cryptographic implementations to the most powerful side-channel attacks, thus by improving side-channel attacks, more effective countermeasures can be developed, and the security of cryptography implementations more accurately estimated.

Of critical importance to the practicality of EM and power based side-channel attacks is the signal to noise ratio (SNR) of the EM/power measurements. In the study of side-channel analysis, SNR has a particular meaning: $\mathbf{SNR} = \frac{VAR[Q]}{VAR[N]}$, where Q is the targeted side-channel leakage, and N is the noise. Noise - in the side-channel sense - has two main components: First, the traditional measurement noise due to thermal effects and quantization errors during measurement, and second, algorithmic noise. Algorithmic noise is composed of any variation in the side channel signal originating from the targeted device, but not caused by the specific operation being analyzed. This could be due to other processes executing concurrently, or even a part of the algorithm under analysis that does not provide useful information.

Measurement noise can be addressed relatively easily by averaging, amplification, or the use of high precision measurement equipment. Algorithmic noise on the other hand does not benefit from amplification or high precision measurements, as the source of the noise is the same as the information leakage. Averaging can help reduce some types of algorithmic noise, but not always. This leads to one of the main benefits of EM measurements over power, that being the ability to selectively collect leakage from a particular section of a cryptographic implementation. By measuring EM leakage from a specific location, algorithmic noise can be reduced, as the signal recorded is related to only the informative leakage, and other sources of algorithmic noise are not sensed. While EM measurements do allow for reduced algorithmic noise, they often also result in increased measurement noise. Furthermore, for EM measurements to reduce algorithmic noise, the measurements must be made at the correct position, and locating this position can be difficult and time consuming, but is often overlooked in existing literature. Additionally, while measurement noise is frequently regarded as a non-issue, modern countermeasures specifically inject noise or attenuate leakage signals beyond what can be addressed through amplification or higher precision measurements. While averaging can fix this, it requires more measurements to be made - meaning the countermeasure is effective.

By addressing these issues, more powerful side-channel attacks can be created. The fist chapter introduces `SCNIFFER` a platform for efficiently localizing exploitable EM leakage. This allows attackers to benefit from the reduced algorithmic noise of localized EM measurements without exhaustively searching for a high leakage position. The next chapter presents a practical machine learning based EM attack. In this attack, various processing methods are proposed to address the problem of low SNR EM measurements, allowing powerful ML attacks to use EM measurements. In the final chapter a variety of topics are covered, starting with an additional method to improve EM-ML SCA attacks - power to EM mapping. Following this is an SCA evaluation of cryptographic implementations created through high level synthesis. Finally, neural network based cryptosystems are studied.

1.1 SCNIFFER

The first chapter proposes SCNIFFER: a low-cost, automated EM Side Channel leakage SNIFFing platform to perform efficient end-to-end Side-Channel attacks. Presently, to perform EM SCA on an embedded device, the entire chip is manually scanned and the MTD (Minimum Traces to Disclosure) analysis is performed at each point on the chip to reveal the secret key of the encryption algorithm. However, an automated end-to-end framework for EM leakage localization, trace acquisition, and attack has been missing. Using a leakage measure such as Test Vector Leakage Assessment (TVLA), or the signal to noise ratio (SNR), we propose a greedy gradient-search heuristic that converges to one of the points of highest EM leakage on the chip (dimension: $N \times N$) within $O(N)$ iterations, and then perform Correlational EM Analysis (CEMA) at that point. This reduces the CEMA attack time by $\sim N$ times compared to an exhaustive MTD analysis, and by $> 20\times$ compared to choosing an attack location at random. We demonstrate SCNIFFER using a low-cost custom-built 3-D scanner with an H-field probe ($< \$500$) compared to $> \$50,000$ commercial EM scanners, and a variety of microcontrollers as the devices under attack. The SCNIFFER framework is evaluated for several cryptographic algorithms (AES-128, DES, RSA) running on both an 8-bit Atmega microcontroller and a 32-bit ARM microcontroller to find a point of high leakage and then perform a CEMA at that point.

1.2 Cross Device EM-ML SCA

The second chapter presents a Cross-device Deep-Learning based Electromagnetic (EM-X-DL) side-channel analysis (SCA), achieving $> 90\%$ single-trace attack accuracy on AES-128, even in the presence of significantly lower signal-to-noise ratio (SNR), compared to power SCA. With an intelligent selection of multiple training devices and proper choice of hyperparameters, the proposed 256-class deep neural network (DNN) can be trained efficiently utilizing pre-processing techniques like PCA, LDA, and FFT on the target encryption engine running on an 8-bit Atmel microcontroller. Finally, an efficient end-to-end SCA leakage detection and attack framework using EM-X-DL demonstrates high confidence of an attacker with ¡20 averaged EM traces.

1.3 Additional Side-Channel Attacks

The final chapter investigates three topics: Power to EM mapping, the security of high-level synthesis generated implementations of AES, and neural encryption through adversarial training. Power to EM mapping would allow EM-ML SCA models, such as the one in the previous chapter, to be trained by collecting high SNR power traces and using a mapping to create EM trace. Through this the number of measurements would be reduced. Different methods for finding a mapping are compared, and the performance of generated EM traces in an ML-SCA model is investigated. Finally, neural encryption is studied as a possible target for side channel attacks. The security of a neural cryptosystem is analyzed, and it is to be rather insecure, able to be broken without needing side-channel information.

2. SCNIFFER: LOW-COST, AUTOMATED, EFFICIENT ELECTROMAGNETIC SIDE-CHANEL SNIFFING

2.1 Introduction

As the internet of things (IoT) continues to grow, security of many edge nodes has become critical. With many of these edge nodes being simple microcontrollers, side-channel attacks pose a powerful threat to their security. In the world of cryptography, side-channel attacks have long been identified as a threat to the security of computing and communication systems attempting to provide confidentiality and integrity of sensitive data, since the introduction of Differential Power Analysis in [1]. By analyzing physical side-channel information, such as power consumption, timing, or electromagnetic emissions, cryptographic algorithms that are mathematically secure can be broken efficiently.

EM side-channel analysis (SCA) is a method of using the information found in the electromagnetic emissions of a cryptographic system to extract the secret key, compromising the security of such a system. Such attacks have been shown to be capable of actually extracting secret key information, as in [2] and [3]. These EM emissions originate from current consumption of an IC running cryptographic algorithms, which while flowing through the metal layers of an IC cause EM radiation as described in [4]. The EM emissions can either be caused by key-dependent operations or other operations. EM emissions caused by key-dependent operations contribute to the side-channel signal, while EM emissions caused by other operations contribute to algorithmic noise. EM SCA attacks have successfully been used in the real world on PCs, shown in [5] and [6], and also on Smart Cards, in [7] [8]. One powerful and commonly used side-channel analysis technique is correlational electromagnetic analysis (CEMA). In CEMA, EM measurements are taken while a cryptographic algorithm is

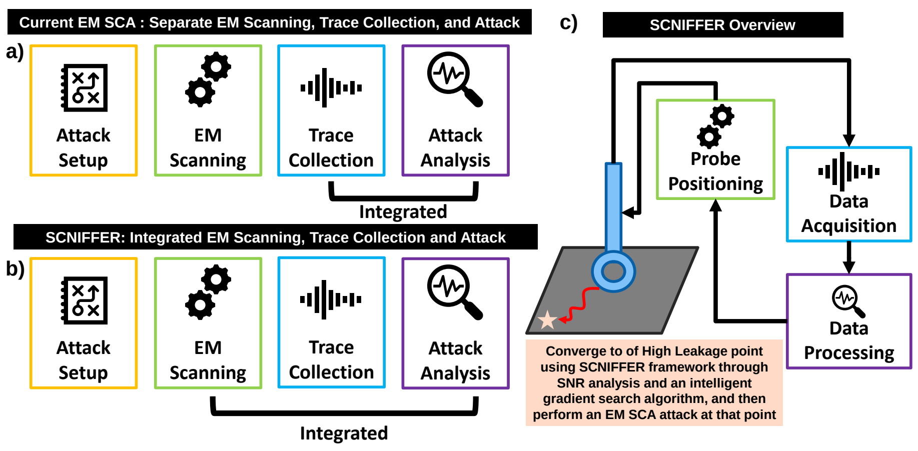

Current EM SCA : Separate EM Scanning, Trace Collection, and Attack
a)
Attack Setup
EM Scanning
Trace Collection
Attack Analysis
Integrated
SCNIFFER: Integrated EM Scanning, Trace Collection and Attack
b)
Attack Setup
EM Scanning
Trace Collection
Attack Analysis
Integrated
c)
SCNIFFER Overview
Probe Positioning
Data Acquisition
Data Processing
Converge to of High Leakage point using SCNIFFER framework through SNR analysis and an intelligent gradient search algorithm, and then perform an EM SCA attack at that point

Fig. 2.1. (a, b) Comparison between existing EM SCA systems and `SCNIFFER`. While current frameworks have integrated trace collection and attack and analysis, `SCNIFFER` integrates EM scanning as well. (c) High level overview of proposed `SCNIFFER` framework. `SCNIFFER` analyzes EM leakage and uses a gradient descent algorithm to locate points of high informative leakage at which the EM SCA attack should be performed.

executing on the target system (each measurement is known as a trace), and these traces are correlated with a leakage model, such as the Hamming Weight or Hamming Distance of data at a particular point in an algorithm [1], under a hypothesis of a subset of the secret key. In a successful attack, the hypothesis that results in maximum correlation corresponds to the secret key. Thanks to the divide and conquer nature of side-channel analysis, the cost of performing an SCA attack is linear in the key size, rather than exponential, as in brute force or other cryptanalysis methods.

2.1.1 Motivation

EM side-channel attacks, while powerful in that they are non-invasive and do not require any physical changes to the system being attacked, and benefit from allowing an attacker to choose the location with maximum information leakage (SNR), introduce a number of additional challenges compared to the power SCA attacks.

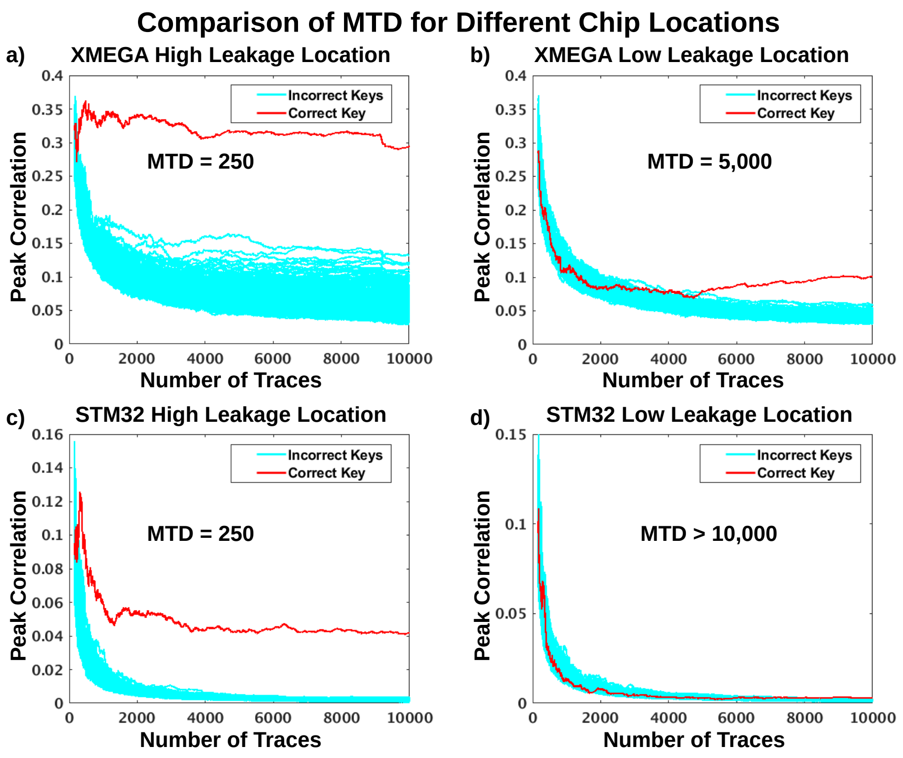


Fig. 2.2. The difference in MTD between a CEMA attack at a point of high leakage vs. at a point of low leakage for both an 8-bit XMEGA microcontroller (a, b) and a 32-bit STM32F3 microcontroller (c, d). At a location of high leakage, the correct key separates in 250 traces for both microcontrollers, while a low leakage location requires $> 20\times$ more traces on the XMEGA. At a low leakage location on the STM32F3, the key does not separate at all within 10,000 traces.

Firstly, as the EM signals go through a power to EM transformation that reduces amplitude compared to the measurement noise floor, meaning more traces, or more expensive measurement equipment may be needed to perform an attack. Secondly, unlike power attacks, EM attacks require attackers to choose the location of the attack in the system to capture the EM traces. However, scanning a device to determine this point is is not currently integrated into current frameworks (Figure 2.1(a)). This choice of location can have a drastic impact on the effectiveness and efficiency of an attack. As seen in Figure 2.2, depending on where the EM probe is placed on a chip, the MTD for a CEMA attack can vary by $> 20\times$, even for the small 9mm x 9mm Atmega and STM microcontrollers used as the target devices for this chapter. Current methods for determining the best location to perform CEMA are based on exhaustive search, simply performing a CEMA attack at most locations. Alternatively, it is also possible to choose an arbitrary location, and use as many traces as necessary to perform the CEMA. Practically, if the size of the system is larger, finding the correct location of the EM leakage becomes extremely challenging and requires scanning the entire chip/system.

Given the limitations of present attack systems, in this chapter, we propose a low-cost, fully automated, end-to-end platform for performing efficient EM side-channel attacks. `SCNIFFER` integrates EM scanning, trace collection, and attack/analysis into a single framework (Figure 2.1(b)). A high level overview of the `SCNIFFER` framework is shown in Figure 2.1(c). The core of this framework is a $\sim$ \$200 3-D printer, which we have modified to utilize as a low-cost EM scanner. `SCNIFFER` also uses a greedy gradient-search heuristic using a leakage measure, such as test vector leakage assessment (TVLA), or SNR to quickly and automatically locate a point of high data-dependant leakage (referred to as simply high leakage throughout this work). Finally, once the point is determined, the proposed `SCNIFFER` framework performs the correlational or differential EM analysis (CEMA/DEMA) at this point. While both CEMA and DEMA are possible attacks, in this chapter, we will demonstrate results with CEMA. Such an automated low-cost attack platform significantly increases the threat surface for IoT devices, however, it should be noted that `SCNIFFER` does not constitute a new attack; and existing countermeasures against EM SCA attack are effective against `SCNIFFER`. The `SCNIFFER` system presented in this chapter is published in [9].

2.1.2 Contribution

Specific contributions of this chapter are:

- **Low-cost Automated EM Side-channel Analysis Framework:** A fully-automated system for efficiently scanning a cryptographic chip and finding a location of high leakage to mount an end-to-end EM SCA attack is proposed. The entire attack set-up is extremely low-cost, owing to the custom-built EM scanner (adapting a $\sim$ \$200 3-D printer) used for mounting the attack, compared to the commercially available EM probe stations, which are very costly ($>$ \$$50,000$). The system achieves 100μm spatial resolution, and has a scan range of 220mm $\times$ 220mm, and is easily replicable (Section 3).

- **Integrated EM Scanning, Trace Collection, and Attack:** EM Scanning is brought in the loop of the attack framework through the proposed greedy gradient-descent heuristic algorithm, which analyzes leakage on-the-fly to efficiently scan the chip and locate a point of high leakage. This algorithm converges to a high leakage location on an $N \times N$ chip within $O(N)$ iterations. This algorithm is evaluated with both TVLA and SNR as the measures of leakage, and results for the complete system on a variety of cryptographic targets are shown (Sections 4, 5, 6).

2.1.3 Chapter Organization

The remainder of the chapter is organized as follows. Section 2 provides the background and summarizes the existing works on EM Scanning and side-channel attacks. In Section 3, the `SCNIFFER` framework is introduced and the low cost, custom-built EM scanning platform is presented. Section 4 describes two options for measuring leakage, TVLA and SNR, and provides motivation for finding a point of high leakage. In Section 5, the gradient-descent algorithm for efficiently determining a point of high information leakage is proposed. Next, Section 6 provides results of running the system on microcontrollers of varying architectures, cryptographic algorithms executed, and measures of leakage. Finally, Section 7 concludes the chapter.

2.2 Background and Related Work

IoT devices have been successfully attacked using side channel attacks, for example CPA was used to extract encryption keys from Philips Hue smart lamps in [10]. EM side-channel attacks were first proposed in [11], and share many properties with power side-channel attacks, however, can be performed at a distance, even up to one meter, as in [12]. One of the most powerful EM SCA attacks is CEMA, which is the straightforward application of Correlation Power analysis (CPA) [13] on EM traces.

However, to make these profiled and non-profiled EM SCA attacks more practical and real-time on any embedded platform/device, the trace capture and the attack needs to be automated and more efficient.

`SCNIFFER` can use several methods of assessing leakage, for instance, simple signal magnitude, Test Vector Leakage Assessment (TVLA) [14], or SNR [15]. In **TVLA**, two sets of traces are collected. In one set, both the key and plaintext used as input to the algorithm under test are kept fixed, and in the other the plaintext is varied randomly, while the key remains fixed. To assess the leakage, one then performs Welch's t-test for each time point of the trace. Welch's t-test is given by $t = \frac{\bar{X_1} - \bar{X_2}}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}}$, where $\bar{X_1}, \bar{X_2}$ are the sample means of the two sets, s_1, s_2 are sample standard deviations for the sets, and N_1, N_2 are the sizes of the sets. If the maximum t-value at a point is above 4.5, one can conclude leakage is present with 99.999% confidence. Meanwhile, we consider the signal to noise ratio as defined in [15], to be **SNR** $= \frac{VAR[Q]}{VAR[N]}$, where Q is the side channel leakage, and N is the noise. Unlike TVLA, which does not guarantee exploitable leakage, SNR defined in this way can be directly related to the success rate of a CEMA attack [15].

Once `SCNIFFER` has chosen a point to attack, CEMA is used to recover the secret key. CEMA revolves around making hypotheses on secret values, then predicting the EM leakage of an intermediate variable based on the key. Measurements (traces) are taken while the device performs encryption, then the measurements are correlated with the predicted leakage for all hypotheses. The hypothesis that results in the

Table 2.1.
Comparison with previous works. SCNIFFER is significantly lower cost compared to previous works, and additionally is the only system designed to maximize the effectiveness of an attack, as other systems seek only the location of most informative leakage.

	Search Technique	Search Metric	Attack Technique	Positioning Accuracy	Cost	System Focus
[18]	Exhaustive Search	SNR	CEMA	100μm	>$10,000*	Leakage Localization
[19]	Exhaustive Search	Difference of Means	Template Attack	50μm	>$10,000*	Leakage Localization
[20]	Greedy Search	DEMA	DEMA	2.5μm	>$50,000*	Leakage Localization
This work	Gradient Search	TVLA/SNR	CEMA	100μm	~$500	End-to-End Attack

* Estimated cost of system based on listed components and specifications

largest correlation is taken as the guess for the secret value. The number of traces needed to recover the key in this way is then the minimum traces to disclosure (MTD). In this chapter, the secret values are the bytes of the AES key, and the intermediate variable is the first round sbox output, and Hamming Weight, that is, the number of 1's in the binary representation of this variable, is used as the leakage model of data at this point.

Addressing the issue of finding where a chip leaks the most EM radiation has been investigated in [16], and [17]. EM scanning with a focus on side-channel attacks, that is, determining where the most cryptographic information leaks within a chip has been addressed in [18], [19], and [20]. However, such methods focus on observing the leakage over the entire chip, not efficiently finding the point or region of the maximum leakage. This causes these methods to take a long time and a majority of the time is spent collecting data that is unnecessary for an attacker. More recently in [21], an adaptive method to determine the location of greatest cryptographic leakage without resorting to exhaustive search is presented. However, this method performs a full SCA attack at each location analyzed, again making it unsuitable for an attacker, whose goal is only a single successful attack. By creating a framework that minimizes this

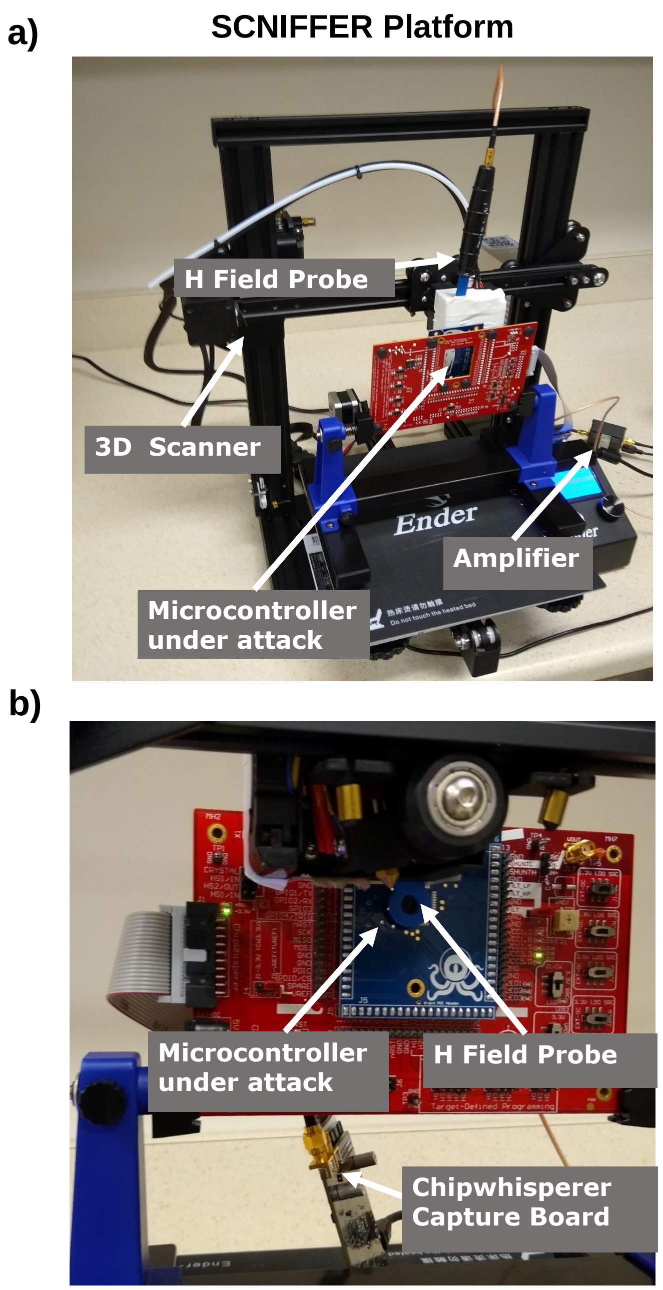

a)
SCNIFFER Platform
H Field Probe
3D Scanner
Amplifier
Microcontroller under attack
b)
Microcontroller under attack
H Field Probe
Chipwhisperer Capture Board

Fig. 2.3. (a) The complete EM Scanning and trace capture set-up system, including the 3-D printer, Chipwhisperer system, EM probe, amplifier, and victim. (b) Close-up of scanner, showing probe and victim board.

unnecessary data collection, EM side-channel attacks can be made more efficient, powerful, and practical, requiring far fewer traces to reveal the secret key of the cryptographic algorithm. Additionally, these platforms can be orders of magnitude more costly than the system proposed in this chapter, for instance the Riscure EM Probe station [22] itself can cost $\sim \$50,000$, while the entire `SCNIFFER` system costs $< \$500$. Table 2.1 compares the `SCNIFFER` system to previous works. Note that while all previous works as shown in the table aim to locate the point of greatest informative leakage, only `SCNIFFER` focuses on minimizing the total number of traces needed for a successful attack. `SCNIFFER` is the first fully-automated, efficient EM SCA attack framework and the system is described in the following section.

2.3 `SCNIFFER`: Low Cost Automated EM Scanning

The `SCNIFFER` system is designed for low cost and automation. In this section, we first describe the physical components that make up `SCNIFFER`, then discuss the automation aspect of the system.

2.3.1 Low Cost EM Scanning Setup

The scanning hardware consists of an Ender-3 3-D printer [23] with a 10mm loop diameter H-field probe attached to the extruder, the Chipwhisperer [24] platform for interfacing with the victim (The CW309T-XMEGA mounted on the 308 UFO Target board) and trace collection, an amplifier to amplify the EM probe output, and finally a PC to control both the 3-D printer and the Chipwhisperer Lite capture board. While such EM scanning systems do exist, for instance, Riscure's EM Scanning Station, we chose to create such a system from scratch for the following reasons: 1) Commercial scanning systems (like Riscure [22]) scanning station is orders of magnitude more expensive and 2) It is very straightforward to interface with the custom system to develop the scanning algorithm. As seen in Table 2.2, the cost of a commercial scanner is orders of magnitude higher than `SCNIFFER`, and while it is hard to know if this price

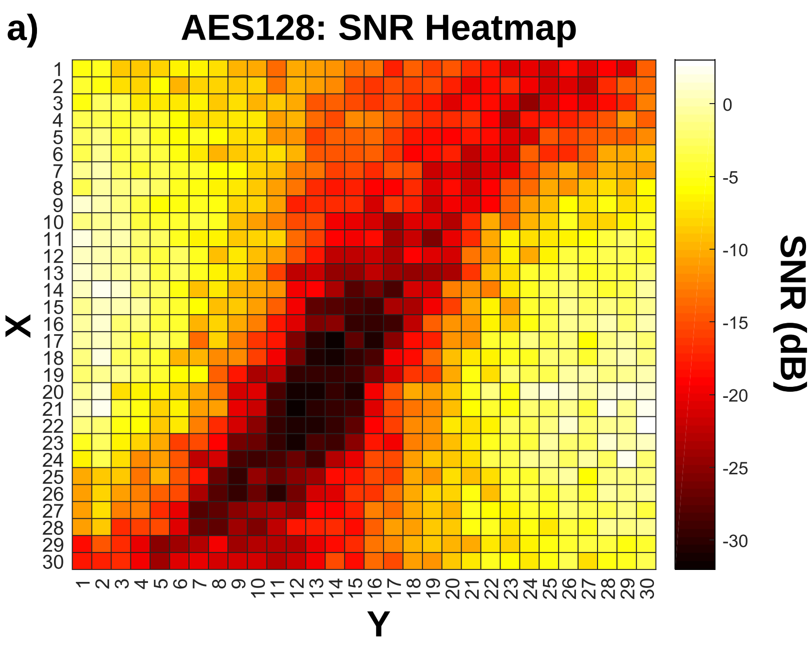
Table 2.2.
Summary of the main components of the SCNIFFER system, their costs, performance, and a comparison to Riscure's EM Probe Station.

	Scanner	Amplifier	Probe
Picture			
Cost	$200	$50	$10
SCNIFFER Specifications	100 μm	20dB	16mm^2
Riscure EM Probe Station Specifications	2.5 μm	-	1mm^2

has been inflated by the selling company, it is reasonable for prices to be higher, as there are not many EM scanners on the market.

To manipulate the probe, an Ender-3 3-D printer, running stock firmware was used. This model of printer has a minimum step size of 0.1mm, and can be controlled via a USB serial connection. It has a maximum movement speed of 180 mm/s, with a print area of $220mm \times 220mm \times 250mm$. The precision and speed offered by this 3-D printer are sufficient to complete a 50×50 scan of the $9mm \times 9mm$ IC used in testing in an acceptable time. Additional justification for the choice of printer, beyond the cost includes the ease of interfacing, the form factor, maintainability, and software support. The open source firmware used by this printer is well documented, and can be controlled through an exposed serial port, making interfacing very easy. The printer also has an open form factor that allows the probe and victim board to be mounted easily. While the durability and hardware support would not be as good as a commercial EM scanner, the simple construction and use of off-the-shelf components make maintenance straightforward. The software support is quite strong, being open source, and the printer is plug-and-play compatible with any device with a serial port. The system is capable of performing a 30×30 scan of the chip in ~ 15 minutes, and perform an amplitude scan in ~ 75 minutes. The probe used is a commercial H-field probe for performing EMC measurements, and the signal is amplified before being passed to the Chipwhisperer capture board. *While the probe used does not have extremely high spatial resolution, the probe resolution matches the scan resolution*, allowing heatmaps such as the one in Figure 2.4(a) to be created, and Chipwhisperer is able to capture enough information leakage for the target devices considered, leading to low MTDs when probed at appropriate locations, as seen in figure 2.2, while still being low cost. Even though this probe is on the larger side, the `SCNIFFER` platform is compatible with more sensitive probes and is expected to become more precise with such probes. The complete system is shown in Figure 2.3(a) showing the 3-D printer, the probe, Chipwhisperer system, and PC. The probe and victim IC are shown in detail in Figure 2.3(b). The probe position can be controlled manually, through the 3-D printer controls, or programmatically through the serial connection to a PC, as it is in the `SCNIFFER` system.

The major cost savings in the `SCNIFFER` system come from using a low cost 3-D printer to control the probe, instead of a high cost motorized table. The total cost of the 3-D printer, probe and amplifier used in `SCNIFFER` is $\sim \$500$, which is a few orders of magnitude less expensive than many motorized tables by themselves, and nearly two orders of magnitude less expensive than systems such as Riscure's EM probe station ($\sim \$50,000$). While more expensive scanners, probes and measurement systems could improve spatial and frequency resolution, such a system would only be available to very sophisticated attackers. As `SCNIFFER` aims to demonstrate that practical, low-cost attacks are possible using systems two orders of magnitude cheaper than existing scanners, high-cost, high resolution components are not used. Table 2.2 summarizes these components, including their costs and performance compared to the Riscure system.



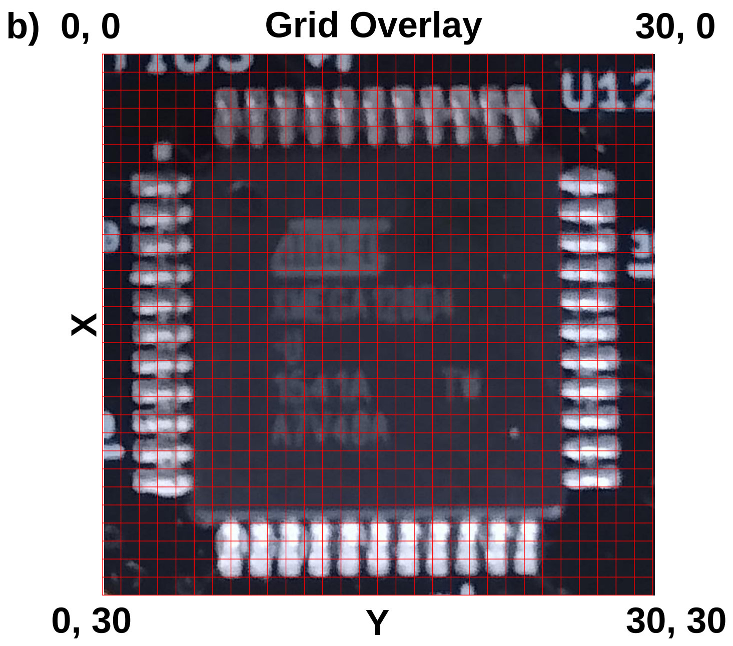


Fig. 2.4. (a) Heatmap of the SNR values obtained by performing a full 30×30 scan of the 8-bit target microcontroller. (b) This shows the grid divisions where leakage measurements were performed. 1000 traces were used to compute the SNR values at each point. The part of the target microcontroller board which leak the most information can be observed.

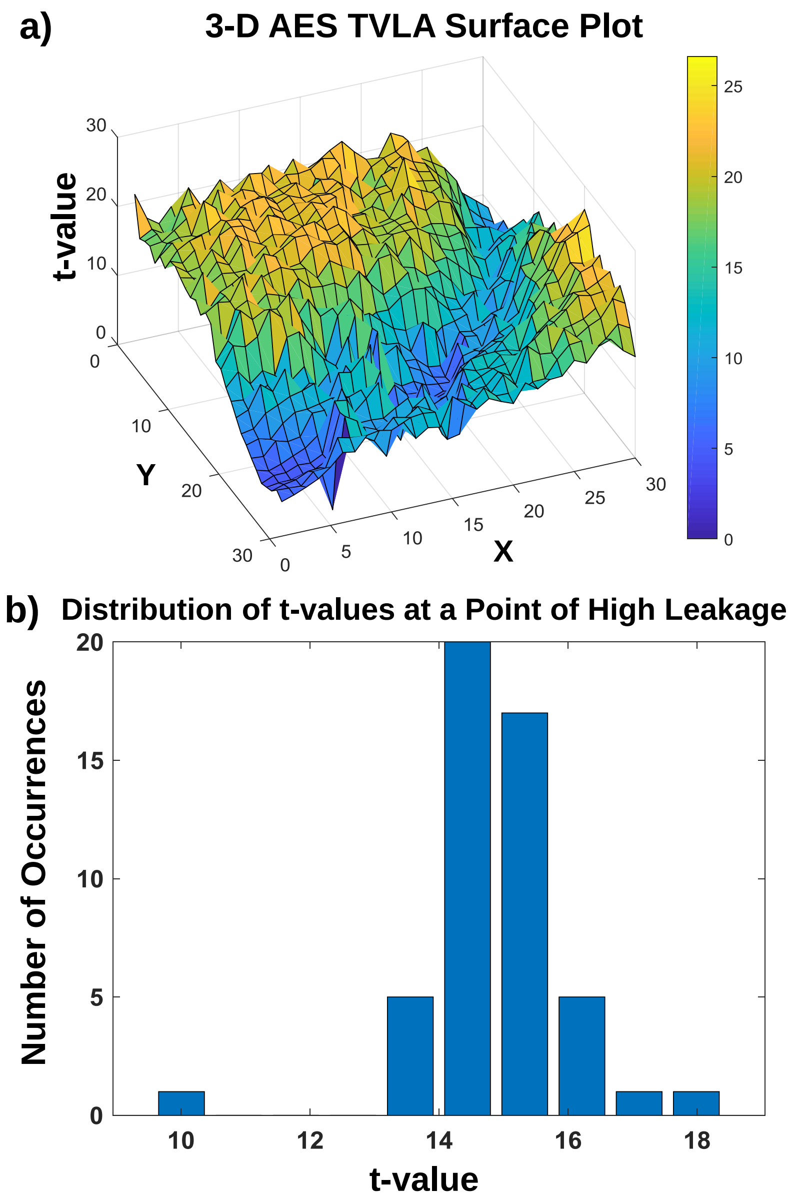


Fig. 2.5. (a) TVLA surface plot. Again, the surface is not smooth or monotonic, as there are many local minima and maxima, as in Figure 2.6(a). (b) Histogram of TVLA measurements at a single point. 50 TVLA measurements were made at a point of high leakage, each done as in (a), using 400 traces each. Given the distribution much wider seen in (b), the increased roughness of the surface in (a) can be explained.

2.3.2 Automated EM Scanning

Now that the `SCNIFFER` system's low cost hardware has been described, we move to the automated scanning and attack procedure. The basic premise of the automated system is to locate a point on the target device where the chosen leakage measure is high by using the scanning algorithm specified in Section 5, and then to automatically perform CEMA at this point. This removes the need for an expert to manually analyze example traces to choose a location for an attack.

During an attack, the probe is positioned at a location dictated by the intelligent scanning algorithm, then, the appropriate ADC phase for trace collection is determined by capturing traces at varying ADC phases, and the phase giving the largest average amplitude is chosen for further measurements at that particular point. The signal is sampled at 29.48MHz, 4$\times$ the clock frequency of 7.37MHz, so clock edges are aligned to samples. The signal is amplified by the external amplifier, as well as the Chipwhisperer internal amplifier (set to a gain of 34.5dB), but no other prepossessing is performed. Chipwhisperer is then used to capture traces for leakage measurement (through SNR, TVLA or other measures) and finally CEMA is performed at the location found by the algorithm to have the highest leakage. Example leakage measures tested with `SCNIFFER`, and the development of the intelligent scanning algorithm, along with detailed results are described in the following sections.

2.4 Signal Leakage Measurement using `SCNIFFER`

As the choice of probe location is a major factor in determining the number of traces needed to recover a key in CEMA as shown in Figure 2.2, this location must be chosen intelligently. Currently, this is done by either exhaustive search of the entire chip, or by an expert evaluating sample EM traces at several locations, and choosing a location based on visual inspection of the traces. While an exhaustive search will certainly produce the best location to attack, it requires a large amount of time, especially for systems with a large initial MTD. Choosing a location based on visual inspection of traces may result in a location that can be attacked, however not necessarily the best in terms of MTD. Additionally, this method requires an expert to perform the inspection of traces. In this chapter, we aim to fully automate the process of choosing a location as an expert might, by looking at measures of leakage, and finding a location with high leakage. As with a manual choice, this location may not be the location corresponding to the lowest MTD, but should leak enough information to be attacked in a reasonable amount of time, without the need for an expert.

`SCNIFFER` is designed such that any measure of leakage can be used. For example signal amplitude, Test Vector Leakage Assessment (TVLA) [14], or SNR could be used, and the `SCNIFFER` platform will be able to converge to a location where the leakage measure is high in $O(N)$ measurements. We provide results using both TVLA and SNR, both described, and then compared in the following subsections.

2.4.1 Signal Amplitude for Leakage Measurement

As motivation for why side-channel leakage measures must be used with `SCNIFFER` to locate low MTD locations, we measure the signal amplitude at each point of the victim chip, producing the heatmap seen in Figure 2.7(b). The amplitude was measured as the mean square amplitude of each trace, averaged across 10 traces. As can clearly be seen in that figure, the amplitude does not correlate to the MTD at all, as expected.

Hence, further results are shown using one of the two leakage measures explained in the following sections, TVLA and SNR. While these are the measures chosen for demonstrating `SCNIFFER`, they are by no means the best nor the only measures that can be used, as `SCNIFFER` does not rely on specific leakage type, only requires that the leakage correlate with the MTD. Determining the best measures of leakage in terms of the attack success rate and minimum number of traces required is a future research direction.

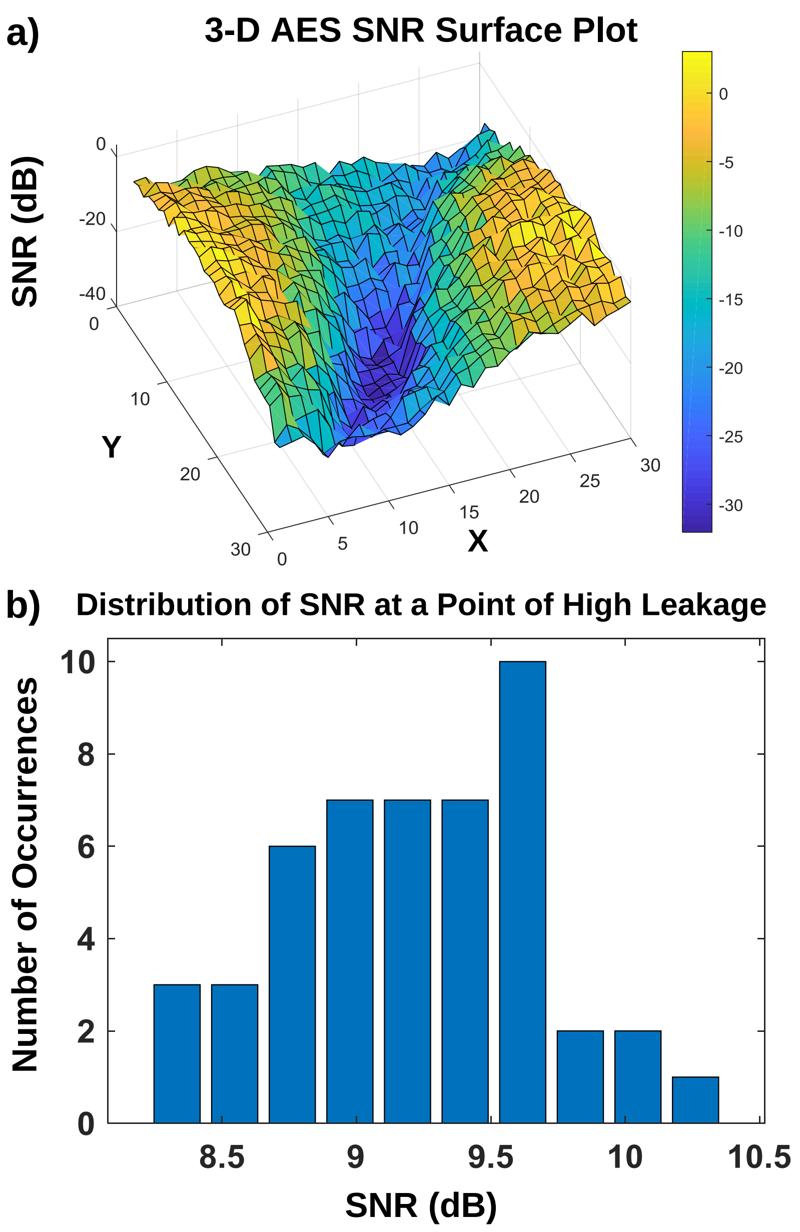


Fig. 2.6. (a) SNR surface plot of the same scan as Figure 2.4(a). Here it can be clearly seen that the surface is not smooth or monotonic, as there are many local minima and maxima. (b) Histogram of SNR measurements at a single point. 50 SNR measurements were made at 1 point. This distribution can explain some of the roughness of the surface seen in (a).

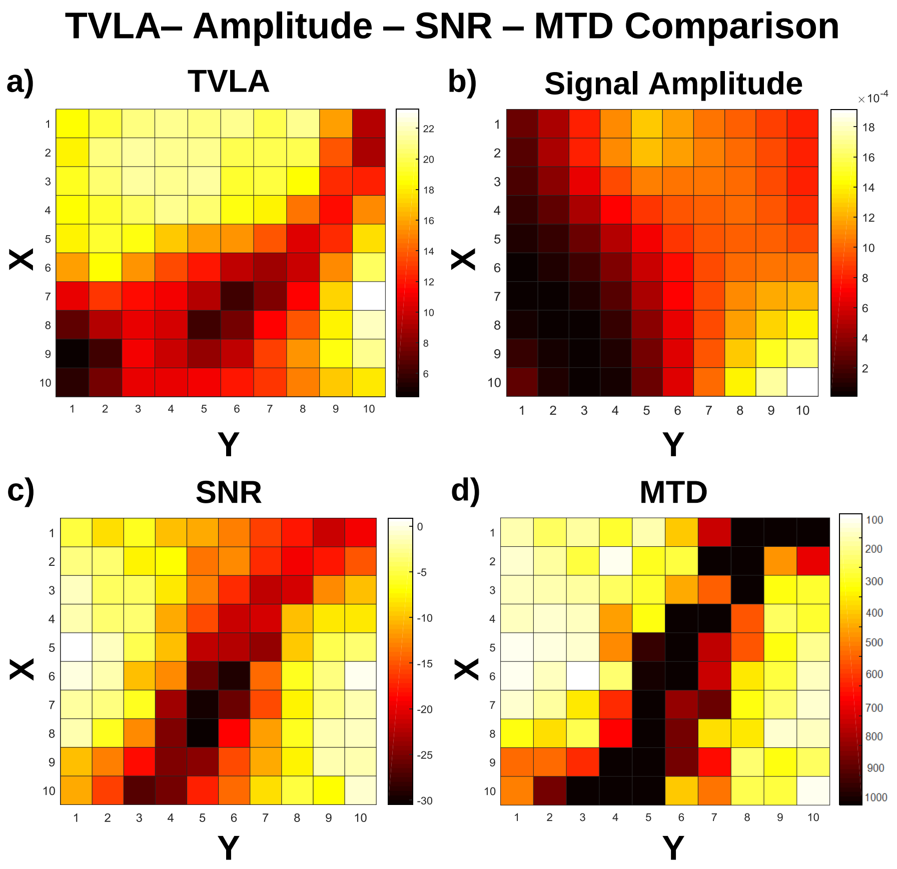


Fig. 2.7. 10 × 10 heatmap of (a) TVLA values (b) signal amplitudes (c) SNR values and (d) MTDs. From these plots TVLA and SNR appear to correlate to MTD much better than the signal amplitude. While amplitude is easy to measure, it is clear that high amplitude of leakage does not necessarily correspond to high information leakage.

2.4.2 TVLA for Leakage Measurement

While signal amplitude is quick to measure, it has no relationship to side channel leakage. As the goal of `SCNIFFER` is to locate a position with high side channel leakage, amplitude is therefore not a good measure. A measure that does consider side channel leakage, and may be a better fit for `SCNIFFER` is TVLA. While high t-values from TVLA may not necessarily imply a low MTD, it allows locations where leakage is detected with high confidence to be focused on. The TVLA performed is the non-specific, fixed versus random t-test. We choose $N = 200$ for the number of traces in each group, for a total of 400 traces per TVLA performed. This number of traces creates large separation between points of low leakage and ones of high leakage, as seen in Figure 2.5(a), where the high leakage location reaches a t-value of 22, while the low leakage location only reaches a t-value of 4. Note that the TVLA surface is rough, with many local minima and maxima. Even at a fixed location there is variance in the TVLA measurements, shown in Figure 2.5(b). However, it is infeasible to perform many TVLA measurements at each point to average out this noise.

2.4.3 SNR for Leakage Measurement

Compared to amplitude and TVLA, SNR, as defined in [15] requires more traces, however has a direct relationship to the MTD. Given this relationship, one can estimate the MTD, thus a location maximizing SNR will minimize MTD. 1000 traces were used to calculate the SNR, as for the 8-bit microcontroller used, this gave large separation between locations of high and low leakage, as seen in Figure 2.6, where the SNR varies from -30dB to 3dB. SNR is calculated using the same intermediate variable and leakage model as the CEMA used, that is, the first round sbox output and the the Hamming Weight model, respectively. Like with TVLA, the surface is somewhat rough, but again it is infeasible to take many SNR measurements to average out this noise.

2.4.4 Correlation among Amplitude, TVLA, SNR, MTD

While signal amplitude, TVLA, and SNR can all be used with `SCNIFFER` as measures for leakage, since the end goal of the `SCNIFFER` system is to perform an attack, we investigate how these measures compare to the MTD at each location. To compare the measures, a 10×10 scan of the chip was carried out, and CEMA was performed using 1,000 traces at each point. The resulting heatmap, along with heatmaps for SNR, TVLA, and amplitude, are shown in Figure 2.7. From these results, clearly TVLA and SNR both appear to correlate to the MTD strongly, however amplitude correlates very poorly. While signal amplitude is easy to measure, there is no guarantee that this measure correlates to the MTD, as high signal leakage does not imply high information leakage. Additionally, an uncorrelated EM source having high signal leakage could confuse an attacker into choosing a poor location to attack. While TVLA also does not guarantee high exploitable leakage, it can be used to identify and focus on regions where leakage is detected with confidence. Additionally, for the microcontroller considered in this chapter, TVLA does empirically correlate to the MTD quite well, even if it is not guaranteed to be the case in general. Finally, as SNR is directly related to the attack success rate, it unsurprisingly is highly correlated in practice. Further, due to this correlation, the location of highest SNR will theoretically be the location of lowest MTD, achieving `SCNIFFER`'s goal.

2.5 Greedy Gradient-Search Heuristic

A critical piece of the `SCNIFFER` system is the algorithm for locating the point of high leakage at which the attack should be performed. It is through this algorithm that the `SCNIFFER` attack framework gains benefits over an exhaustive search, as the high leakage location in an $N \times N$ grid can be found with N measurements as opposed to N^2. As an example, we use SNR as the leakage measure to demonstrate the performance of the `SCNIFFER` greedy gradient-search algorithm throughout this section. The remainder of this section describes the algorithm in detail, and provides results of running the algorithm on an Atmel XMEGA 8-bit processor running software AES.

2.5.1 Algorithm Description

To avoid taking measurements at all possible points, we propose a heuristic search algorithm for finding a point of high leakage in a minimum number of scans. The search algorithm works in two phases. In the first phase, the search space is divided into an $M \times M$ grid, where M is the initial grid size parameter, and the leakage is measured at the center of each grid cell. This initial grid must be more coarse than the measurement grid, which would be $N \times N$ Then in the second phase, a gradient search algorithm is started from the point of the highest leakage found in the first phase. The gradient is computed by measuring the leakage of the four grid cells adjacent to the current cell, then treating each measurement as the magnitude of a vector whose direction is the direction from the cell where the gradient is being estimated to the cell where the measurement was made. The sum of these vectors is treated as the estimate of the gradient. The next point to measure is determined by adding a vector in the direction of the gradient with a magnitude of stepSize to the current location. This location is then mapped to a grid cell, and the leakage is next measured in the center of this resulting grid cell. Given this method, movement is restricted to be between grid cells, and is not entirely arbitrary, however movement to diagonal cells or moving multiple cells at once are possible moves, depending on the stepSize parameter.

If the algorithm attempts to measure outside the search space, it will instead move only to the edge and then stop. A maximum number of iterations can also be specified, along with an “iterations without improvement” stopping criteria. The “iterations without improvement” parameter should be set to a sizeable fraction of the grid resolution N, for values too small, several iterations may pass without improvement, especially for noisy surfaces, and the algorithm may stop prematurely. This two phase process is described in Algorithm 1.

2.5.2 Algorithm Performance

Based on experimental results, the algorithm is able to locate a point of high leakage in a $N \times N$ grid of possible measurements in $\approx N$ SNR measurements. Figure 2.8 demonstrates that as the search grid size increases by N^2, the number of tests

```
$N$ = Grid Resolution;
maxLeakage = 0;
initLocs = getInitialLocations(initialGridSize, N);
for $loc \in initLocs$ do
    moveProbe(loc);
    leakage = getLeakage();
    if $leakage > maxLeakage$ then
        maxLeakage = leakage;
        startLoc = loc;
    end
end
moveProbe(startLoc);
bestLoc = startLoc;
$m$ = startLoc;
while Not Converged do
    delta = getDelta(get4Neighbors());
    $m = m -$stepSize$*$delta;
    moveProbe(m);
    leakage = getLeakage();
    if $leakage > maxLeakage$ then
        maxLeakage = leakage;
        bestLoc = loc;
    end
end
```
Algorithm 1: Gradient Search Heuristic to find the high leakage location

required only increases by N, showing the improvement over an exhaustive search is more drastic as the size of the scan increases, either due to increased resolution or larger scan area. We also see the effect of the parameters of the algorithm, and see how varying them affects performance. In Figure 2.9(a), where, by increasing the resolution of the initial search grid, the lowest MTD found for a given number of measurements changes. As expected, as more initial points are scanned, fewer gradient steps are required to converge to the high leakage location. In Figure 2.9(b), the step size is varied, and we see that for a small step size, the algorithm gets stuck in a local minimum, and does not converge to the point of high leakage the other step sizes do. It is worth noting that even though the algorithm gets stuck in a local minimum, the initial grid search, SCNIFFER still finds a relatively low MTD

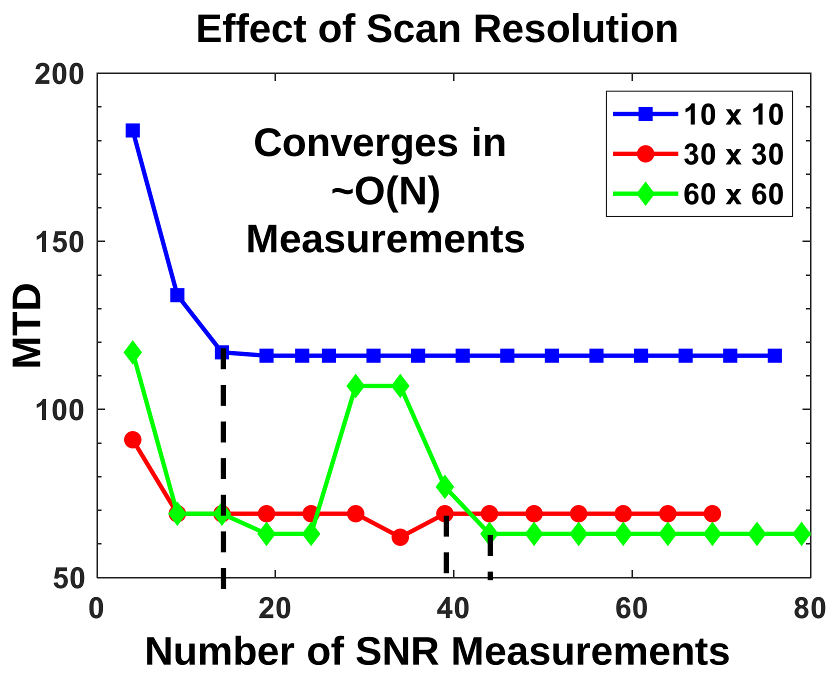


Fig. 2.8. Leakage vs. number of SNR measurements for varying grid scales. Each SNR measurement is computed using 1000 traces collected at the measured location. The data for the 30 $\times$ 30 grid was the same as in Figures 2.4 and 2.6(a). The full 60 $\times$ 60 and 10 $\times$ 10 grids were also collected, allowing the performance of the algorithm to be seen at various degrees of measurement resolution. Through these results, it can be seen that even as the size of the search space increases by N^2, the time to converge increases by only N.

location. A larger step size also converges, and if the step size is too large however, the convergence is slower, and less smooth, as it may step over the best point. Note that the effective step size is a function of both the resolution of the scan, N, and the step size parameter of the algorithm. This, along with the dimensions, L, of the chip allow calculating the effective step size as $\frac{1}{N} * L\ mm * StepSize$. Given these results, one can see that for reasonable choices of parameters, the algorithm is observed to converge to a point of high leakage in $O(N)$ steps for an $N \times N$ grid of measurements, providing `SCNIFFER` with a significant improvement over an exhaustive search.

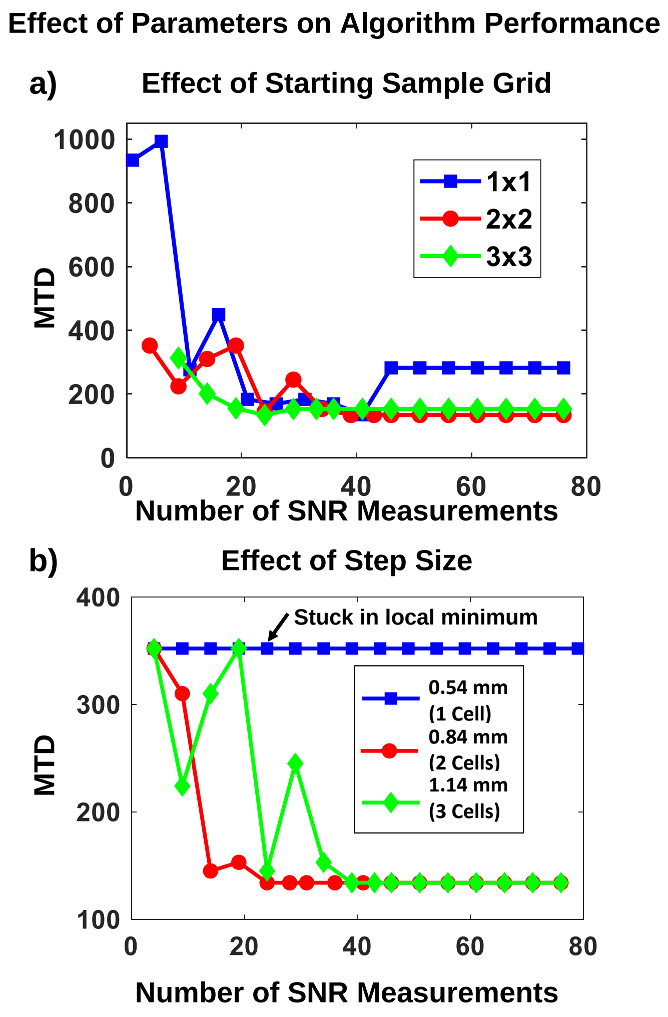
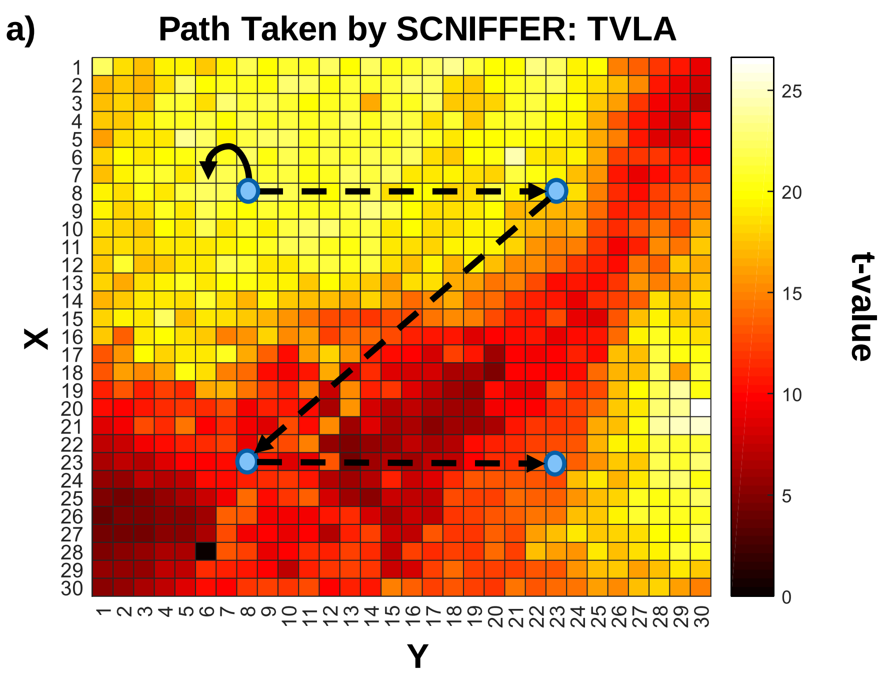


Fig. 2.9. (a) MTD vs number of SNR measurements performed for varying the initial sample grid size parameter. Note that the 2×2 and 3×3 grids locate the point of high leakage within 40 SNR measurements, while a single initial sample point results in a higher MTD, and after 45 such measurements. For all initial sample grid sizes, a step size of 1.14mm was used. (b) This demonstrates the effect of step size on performance. A step size too small can result in the algorithm getting stuck in a local maximum, and in this case as the step size increased, convergence sped up, however, for much larger step sizes, it is possible to overshoot the location of highest leakage, resulting in slower, less smooth convergence. For all step sizes, a 2×2 initial sample grid was used. Both (a) and (b) used a 30×30 scan resolution.



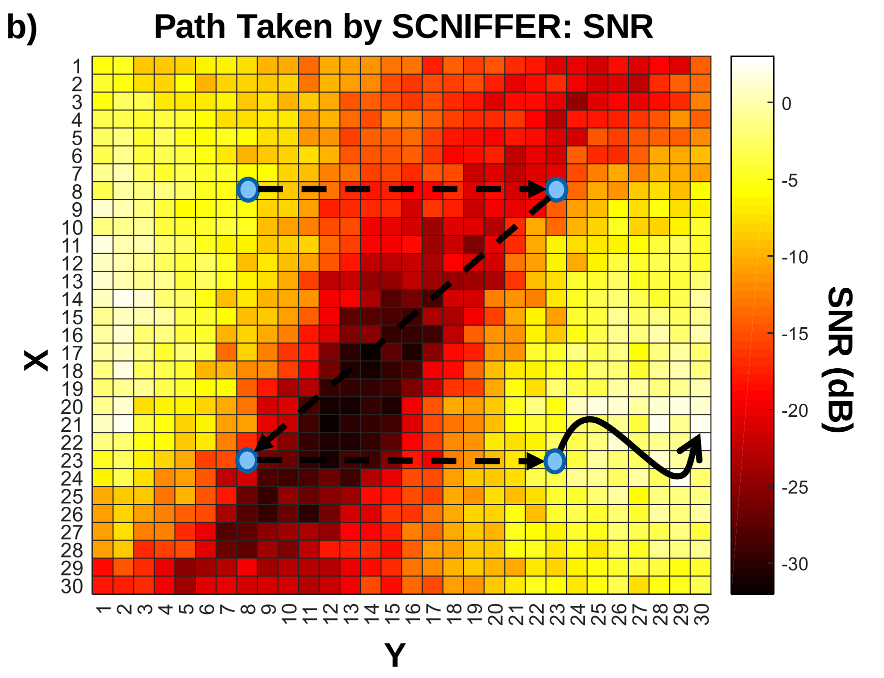


Fig. 2.10. Heatmaps for AES running on the 8-bit microcontroller, with the path taken by `SCNIFFER` shown for TVLA in (a), and SNR in (b). The same search algorithm parameters were used in all cases.

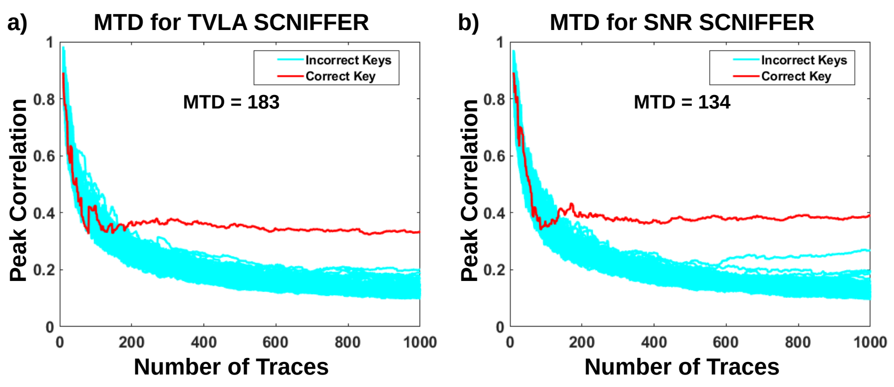


Fig. 2.11. MTD plots at locations found by `SCNIFFER` using TVLA as a leakage measure (a), and SNR as a leakage measure (b). While the MTD is not the minimum, it is fairly close to the minimum for both measures, with SNR having a slightly lower MTD than TVLA.

2.6 Results

In this section, we provide results of using the SCNIFFER framework in various scenarios. We start with the results of an attack using TVLA, then with SNR. Following this, we provide a short discussion of the number of traces needed in a `SCNIFFER` attack. We then show the performance of the TVLA and SNR based attacks for a variety of cryptographic algorithms. Next, results comparing the 8-bit architecture chip used so far to a 32-bit architecture chip are shown, again for both TVLA and SNR measures. Finally, we show results showing the effects of a masking countermeasure, using the SNR based attack.

2.6.1 TVLA Based `SCNIFFER`

While it is not guaranteed to correlate with MTD, TVLA can be used with the `SCNIFFER` algorithm. The path taken for this case is shown in Figure 2.10(a). This path remains in the zone of high TVLA values, and as TVLA correlates well with MTD in our experiments, this location has a very low MTD, seen in Figure 2.11(b),

Table 2.3.
Comparison of different leakage measures used with SCNIFFER, as well as results of a full exhaustive search. The total traces includes the traces needed for the initial search, gradient search, and CEMA. The exhaustive search total traces includes a 1000 trace CEMA at all 100 locations.

Leakage Measure	Convergence Location	MTD	Total Traces
TVLA	(2, 2)	183	5,807
SNR	(7, 10)	134	10,134
Exhaustive	(3, 6)	91	100,000

and is among the lowest on the chip. TVLA at each location requires a total of 400 traces to compute TVLA, and additional traces would be needed for systems with lower SNR, as we describe in section IV D. Additionally, as the TVLA surface is not smooth, convergence is slightly slowed, increasing the attack time.

2.6.2 SNR Based SCNIFFER

In contrast to TVLA, which does not guarantee leakage found is exploitable, SNR does, as it is related to the MTD. We see that SNR based SCNIFFER does take a different path than TVLA, and converges to a different location. The MTD at this location is slightly lower than the TVLA location, but still not the absolute lowest found on the chip. Furthermore, to accurately measure SNR, more traces than TVLA are needed for measurement, increasing the number of traces needed, and this number increases as the SNR reduces, as discussed in section IV D. Despite this, once the SNR reduces below a certain point, shown in Figure 2.12, a SNR-based SCNIFFER attack becomes as efficient as a TVLA-based attack, with the additional guarantee of exploitable leakage.

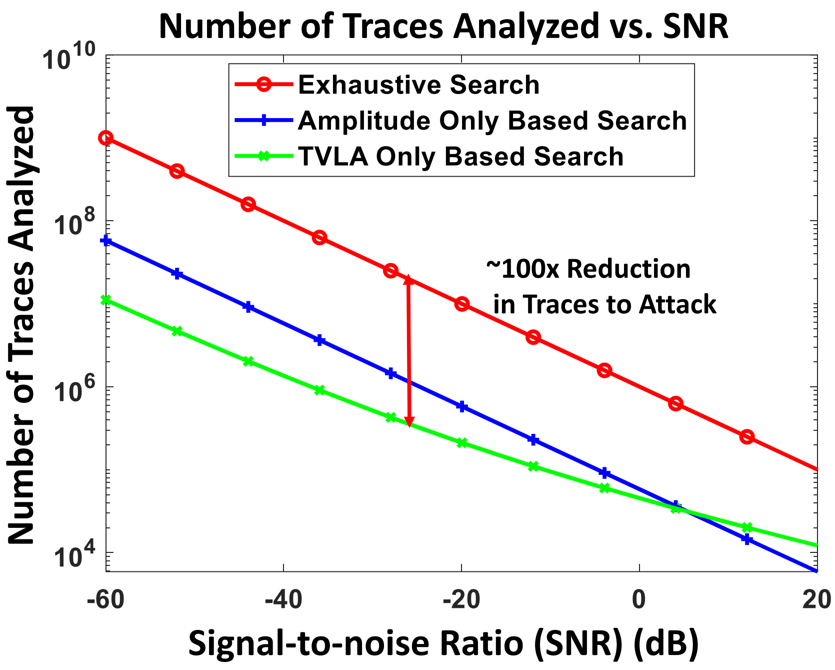


Fig. 2.12. Number of traces required for TVLA and SNR based SCNIFFER compared to exhaustive search vs. SNR for the case of a 10×10 scan. The $\sim 100\times$ reduction is due to the fact that an exhaustive search must perform a CEMA at each location, while SCNIFFER only visits N locations.

2.6.3 Number of Traces Needed For SCNIFFER Attacks

The performance of the SCNIFFER platform can be quantified and compared to other methods by investigating how the total number of traces needed to perform an attack changes as the SNR of the device under attack changes. Previous works have shown in [25] and [15] that the MTD for a CEMA attack is related to the SNR of the signal used in the attack by $MTD = k_0 * \frac{1}{SNR^2}$. Additionally, [26], [27] have shown that the number of traces needed to perform a TVLA (N_{TVLA}) or calculate SNR (N_{SNR}) is also related to SNR by $N_{TVLA} = c_0 * \frac{1}{SNR}$ and $N_{SNR} = c_1 * \frac{1}{SNR}$. From there, it is straightforward to quantify the performance of an exhaustive search and SCNIFFER using both TVLA and SNR as follows,

$$N_{SCN-TVLA} = N * c_0 * \frac{1}{SNR} + k_1 * \frac{1}{SNR^2} \tag{2.1}$$

$$N_{SCN-SNR} = N * c_1 * \frac{1}{SNR} + k_1 * \frac{1}{SNR^2} \tag{2.2}$$

$$N_{exh} = N^2 * k_1 * \frac{1}{SNR^2} \tag{2.3}$$

where $N \times N$ is the resolution of the grid scan, and k_0, k_1, and c_0 are arbitrary constants chosen such that the models match the results presented.

A `SCNIFFER` attack requires measurements to be made at approximately N points for an $N \times N$ grid, as the search algorithm requires $O(N)$ measurements, with each requiring N_{TVLA} in the TVLA case and N_{SNR} in the SNR case. Additionally a single CEMA attack requiring MTD traces is needed, resulting in equations (2.1) and (2.2). An exhaustive search on the other hand would require a CEMA to be performed at all N^2 locations, resulting in equation (2.3). These trends are pictured in Figure 2.12, which clearly shows the $100\times$ reduction in required traces in the case of a 10×10 scan for low values of SNR. This reduction can be explained by the fact that the number of traces needed to measure TVLA or SNR changes as $\frac{1}{SNR}$, compared to the MTD which changes as $\frac{1}{SNR^2}$. Additionally, the number of points traversed is only N, as opposed to N^2 for an exhaustive search. Also, we see TVLA slightly outperforms SNR in terms of number of traces needed to perform an attack when SNR is high. For low SNR, the performance of both measures is mostly equivalent, as the number of traces needed is dominated by the CEMA, and using SNR as the leakage measure gives guarantees on the success rate of the CEMA, which TVLA does not.

2.6.4 Effect of Cryptographic Algorithm on Convergence

Next, in Figure 2.13(a), the effect of different cryptographic algorithms running on the target microcontroller can be seen, when using TVLA. For AES, DES, and RSA,

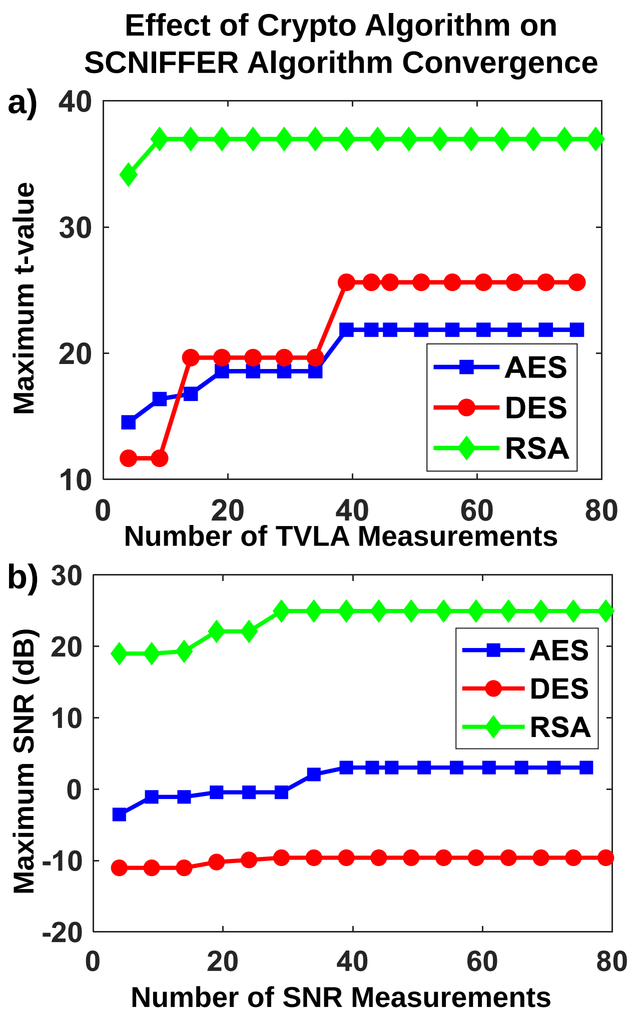


Fig. 2.13. (a) Max t-value vs. number of TVLA tests performed for all cryptographic algorithms (AES, DES, RSA), showing the scanning algorithm performs well, finding the point of max leakage within 40 TVLA tests in all cases, with a grid size of 30×30. The initial sampling grid was 2×2 and the step size was 0.84mm. Note that for RSA, one of the initial samples is already close to the maximum, and this maximum is found in just one step. For AES and DES, whose leakage patterns are less smooth, and have smaller areas of high leakage, the time to converge is higher. (b) Max SNR vs. number of SNR measurements for all algorithms (AES, DES, RSA). The search algorithm again performs well, converging in all cases in about $O(N)$ measurements ($N = 30$ in this case).

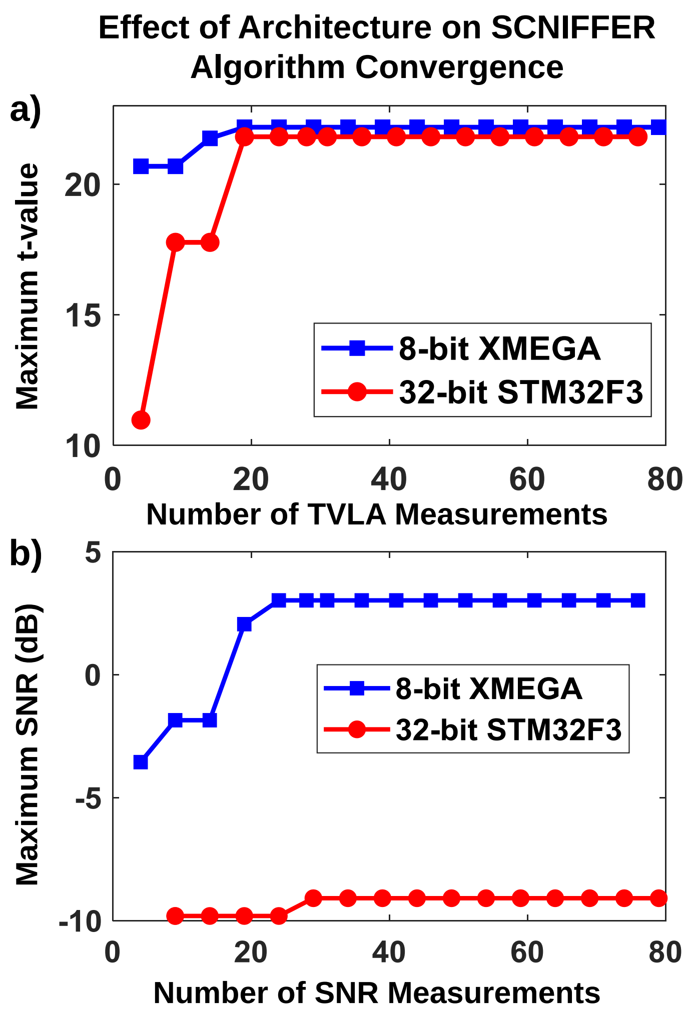


Fig. 2.14. (a) Max t-value vs. number of measurements for both the 8-bit XMEGA microcontroller and the 32-bit STM32F3 microcontroller. The algorithm converges within $O(N)$ measurements, where $N = 30$ in both cases. the algorithm parameters used are the same as in Figure 2.13. (b) Max SNR vs. number of measurements for both microcontroller architectures, again showing convergence in $O(N)$ measurements. The parameters used are the same as those in part (a).

the gradient search algorithm converges a point of high leakage in a similar number of traces. A 30×30 scan was performed for all algorithms, and the parameters were fixed at a 2×2 starting grid and step size of 0.54 mm for all cases. A similar plot, using

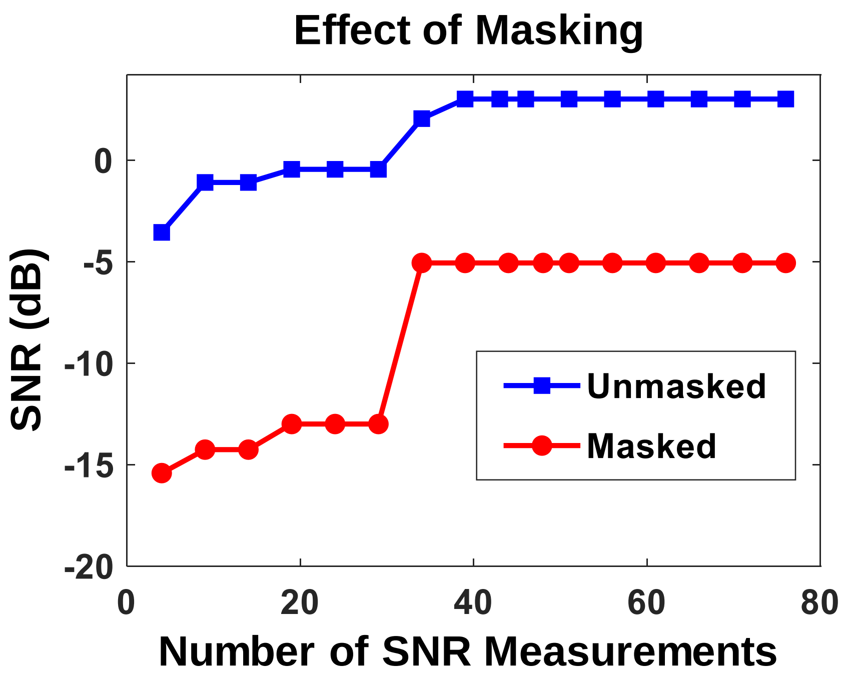


Fig. 2.15. Max SNR vs. number of SNR measurements for the unmasked and a masked implementation of AES on the 8-bit microcontroller. The algorithm converges within $O(N)$ measurements, where $N = 30$ in both cases. The algorithm parameters used are the same as in Figure 2.13.

the same parameters but SNR as opposed to TVLA can be seen in Figure 2.13(b). Again, the search converges in approximately the same number of measurements for all algorithms. Through this, we see that the greedy gradient search algorithm performs well regardless of the specific cryptographic algorithm, and regardless of the leakage measure chosen.

2.6.5 Effect of Architecture on Convergence

Additionally, we investigate the effect of different architectures (microcontrollers) on `SCNIFFER`. Up to now, the results shown have been obtained with an 8-bit XMEGA microcontroller. We now use a 32-bit STM32F3 microcontroller running software

AES as the target device. The STM32F3 uses the same clock frequency as the 8-bit XMEGA, 7.37MHz, and sampling is again done at 4$\times$ this frequency. Similarly the amplifier gain is the same as the 8-bit case. Given the same parameters for the greedy gradient search, the algorithm converges to a location of high leakage within N measurements, with $N = 30$ in this case. These results are shown in Figure 2.14(a) for TVLA, and Figure 2.14(b) for SNR. In both figures, the 8-bit and 32-bit architectures are compared, given the same measurement and search algorithm parameters. In this context, it is worth mentioning that as the size of the chip under attack increases, finding the location of the cryptographic engine could be a difficult task. In scenarios such as attacking large systems, the `SCNIFFER` framework would be extremely useful in efficiently determining the position of high leakage and then performing the attack at that point.

2.6.6 Effect of Masking on Convergence

Lastly, the effects of a masking countermeasure with a fixed mask on the performance of `SCNIFFER` have been investigated. The same 8-bit XMEGA microcontroller was used as the target device, now running the masked implementation of AES-128 from [28]. We again use the same measurement and search parameters, and for both cases, the `SCNIFFER` algorithm converges in approximately $O(N)$ measurements. These results are shown in figure 2.15, where we see the algorithm converges after 35-40 measurements for both masked and unmasked implementations. As one would expect, the SNR for the masked implementation is significantly lower than the unmasked implementation, but the `SCNIFFER` search algorithm is still able to locate a higher SNR region through gradient search. While the measurement parameters used here were the same as elsewhere, an important note is that for countermeasures that reduce the SNR more drastically, would require more traces to be used to calculate the SNR.

2.7 Conclusions

This chapter has introduced `SCNIFFER`, a fully automated integrated system for conducting end-to-end EM side-channel attacks against cryptographic systems. `SCNIFFER` combines an EM leakage scanning platform, and correlation EM analysis into a single system, which can perform all steps of an attack automatically. The system is comprised of a low-cost custom scanning hardware and gradient search heuristic based scanning algorithm. We also plan to make our code for implementing the efficient `SCNIFFER` framework and controlling the low-cost 3-D printer for scanning publicly available.

`SCNIFFER` is capable of using a variety of measures of leakage, and the search algorithm was shown to find a location of high leakage in an $N \times N$ chip search space with $O(N)$ measurements, providing a significant improvement over exhaustive search, and performing all stages of the search and attack completely automatically, removing the need for expert analysis.

Using this fully automated attack, it is possible to efficiently find a point of high leakage and launch a CEMA attack at this location at the press of a button. The attack uses a minimal number of traces, for a variety of microcontroller architectures and cryptographic algorithms. Even as the size of the chip increases, or as protections lowering the SNR, such as masking, are put in place, `SCNIFFER` retains efficiency. Finally, we show that as the SNR of the system under attack decreases, `SCNIFFER` attacks maintain their advantage over existing methods, reducing the number of traces needed by a factor of N compared to an exhaustive search, for an $N \times N$ scan of a chip.

3. EM-X-DL: CROSS-DEVICE DEEP LEARNING SIDE-CHANNEL ATTACK USING ELECTROMAGNETIC SIGNATURES

3.1 INTRODUCTION

With the ever-increasing prevalence of embedded devices and the growth of the Internet of Things (IoT), the security of these devices has become a major concern. Some of the most serious threats to the security of these devices are side-channel analysis (SCA) attacks. By analyzing physical leakage information regarding the power [1], timing [29], or electromagnetic (EM) signatures [11], cryptographic secrets can be extracted. The threat of EM attacks is particularly dangerous, as secrets can be extracted from a distance, and work has been done to minimize EM emissions by using the human body as a communication medium [30], [31]. Among the most powerful of these side-channel attacks are profiled attacks [32], and recently machine learning (ML) models have been shown to be very effective in this profiled SCA attack scenario using both power and EM measurements [33, 34].

3.1.1 Motivation

The main limitation of ML models for profiling SCA attacks is their portability to other target devices. Specifically, these models have been shown to work when the same device is used for both profiling and testing, however, in a real attack, the attacker would use a device to profile, then attack a separate, identical device. The device-to-device variation can be useful, such as with physical Unclonable Functions (PUF's) [35] but cause problems for profiled attacks. This issue of portability has recently been addressed for power ML SCA models on AES-128 in [36], [37], and also with a 3-class DNN attacking RSA implementations for EM SCA [38]. However,

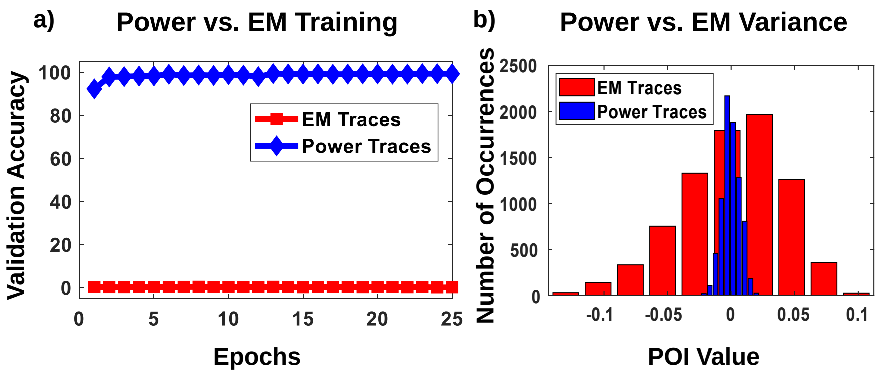


Fig. 3.1. (a) DNN training with raw power and EM traces. The model learns quickly from power, but is unable to learn from raw EM traces. (b) The variance of a point of interest (time sample 103) for both power and EM traces, demonstrating significantly lower SNR of the EM traces.

these works only consider high SNR scenarios, and considering SNR reducing countermeasures such as in [39], [40], [41], and [42] or low cost, low sensitivity EM probes, practical attacks must address the reality of low SNR trace measurements. In this work, we show a deep-learning based cross-device SCA attack in the case of low SNR measurements. In this chapter, we show the first deep-learning based cross-device EM (EM-X-DL) SCA attack on a symmetric key encryption algorithm (AES-128). The EM-X-DL attack is published in [43].

A 256-class DNN model that can be trained successfully ($>$ 99% validation accuracy) [36] using raw time-domain AES-128 power traces (available at `https://github.com/SparcLab/X-DeepSCA`) for a particular microcontroller is rendered futile for EM SCA training even with traces collected from the same device (Figure 3.1(a)). Figure 3.1(b) shows the variance of a point of interest (POI, determined using the difference of means approach [32, 44]) across 10K EM and power traces. It clearly shows that the variation in the EM traces is much higher than the power traces, revealing significantly lower SNR for the EM signatures. On top of this, to solve the problem of portability, we need to take into account the inter-device variations [45]. To resolve all these issues, we utilize averaging to enhance the SNR, analyze different pre-processing techniques to reduce the dimensionality of the data, and develop an intelligent algorithm to choose the set of training devices for efficient profiling. Finally, we also propose an end-to-end EM-X-DL attack framework to perform EM scanning and find the best point of leakage on an unseen target device. A combination of these techniques allows us to achieve $> 90\%$ cross-device accuracy.

3.1.2 Contribution

The specific contributions in this chapter are:

- This chapter presents the first cross-device deep-learning based EM SCA (EM-X-DL) on an AES-128 encryption engine using a 256-class DNN with ten devices for training and tested on a different set of ten test devices (Sec. 3).

- Effect of different pre-processing techniques including principal component analysis (PCA), linear discriminant analysis (LDA), fast fourier transform (FFT), spectrogram, on handling the portability issue is analyzed and compared, showing that the LDA is the most efficient approach to achieve maximum average cross-device key prediction accuracy of $\sim 91.5\%$ with minimum training time (Sec. 3.3).

- An algorithm for the optimal selection of the training devices is proposed, so that the number of training devices and thus the effective training time is minimized (Sec. 4, Algo. 1).

- Finally, an end-to-end EM-X-DL attack using the trained DNN model is demonstrated, starting from the EM scanning to finding the point of maximum leakage on the chip, leading to a successful attack at the best location (Sec. 5).

Table 3.1.
Literature Review for Profiled-Attack Scenario

Profiled Attack Scenario	**Measure-ment Type**	**Profiling Method**	**Corresponding Article**
Same-Device	Power	TA	[32]
		SVM, RF	[33], [46]
		DNN	[47]
	EM	TA	[32]
		DNN	[34]
Cross-Device	Power	TA	[44]
		DNN	[36], [37]
	EM	TA	[48]
		3-Class DNN	[38] (RSA)
		256-Class DNN	**This Work* (AES-128)**

**First EM Cross-Device Deep-Learning Attack on a Symmetric Key Algorithm*

3.2 BACKGROUND & RELATED WORK

3.2.1 EM Side Channel Attacks

Since the inception of power SCA [1], a wide variety of attacks have been demonstrated, which can be broadly classified into non-profiled attacks like differential/correlational power/EM analysis (DPA, CPA, DEMA, CEMA) [13], [1], and profiled attacks, such as the statistical template attacks [32] and ML SCA attacks. While non-profiled attacks perform an attack in a single phase on a target device, profiled attacks consist of two phases, a profiling phase, to learn a leakage pattern and an attack phase, to attack with only a few traces, which practically operate on different devices. During the profiling stage, the attacker will collect traces from a "profiling" device identical to the victim device to build a model. During the attack, this model is then used to recover cryptographic secrets from the victim device.

3.2.2 ML-SCA Attacks

Template attacks have been shown [32] to be capable of recovering secret keys with a small number of traces, making them among the most powerful side channel attacks. More recently, supervised ML techniques have been used for profiling SCA [33]. Among these techniques, DNNs have been one of the most successful, defeating many common countermeasures, such as masking [49] and clock jitter [50]. Table 3.1 provides the summary of related works on profiling attacks. Till date, only one prior work [38] has focused on cross-device EM ML SCA attack using only one test device running RSA. Note that this attack required a 3-class DNN [38], whereas the proposed single-trace (averaged) EM-X-DL attack on AES-128 requires a 256-class DNN, and thus the effects of portability across devices is significantly more prominent. Hence EM-X-DL is the first cross-device EM ML SCA attack on a symmetric key encryption algorithm (AES-128) and has been evaluated against ten different test devices (8-bit Atmega microcontroller).

3.3 EM-X-DL SCA ATTACK

This section evaluates the single-trace (averaged) EM-X-DL attack on AES-128 using a 256-class DNN. For profiling the DNN, EM traces are collected from a set of ten training devices (8-bit Atmega microcontrollers) using the Chipwhisperer [24] platform, specifically the CW-Lite capture board, along with an off-the-shelf H-field sensor (10mm loop diameter) and a 40dB wideband amplifier. The efficient selection of the training devices is discussed in the subsequent section. For evaluating the attack, ten different devices are reserved separately and the cross-device (EM-X-DL) accuracy is reported as an average of these ten test devices.

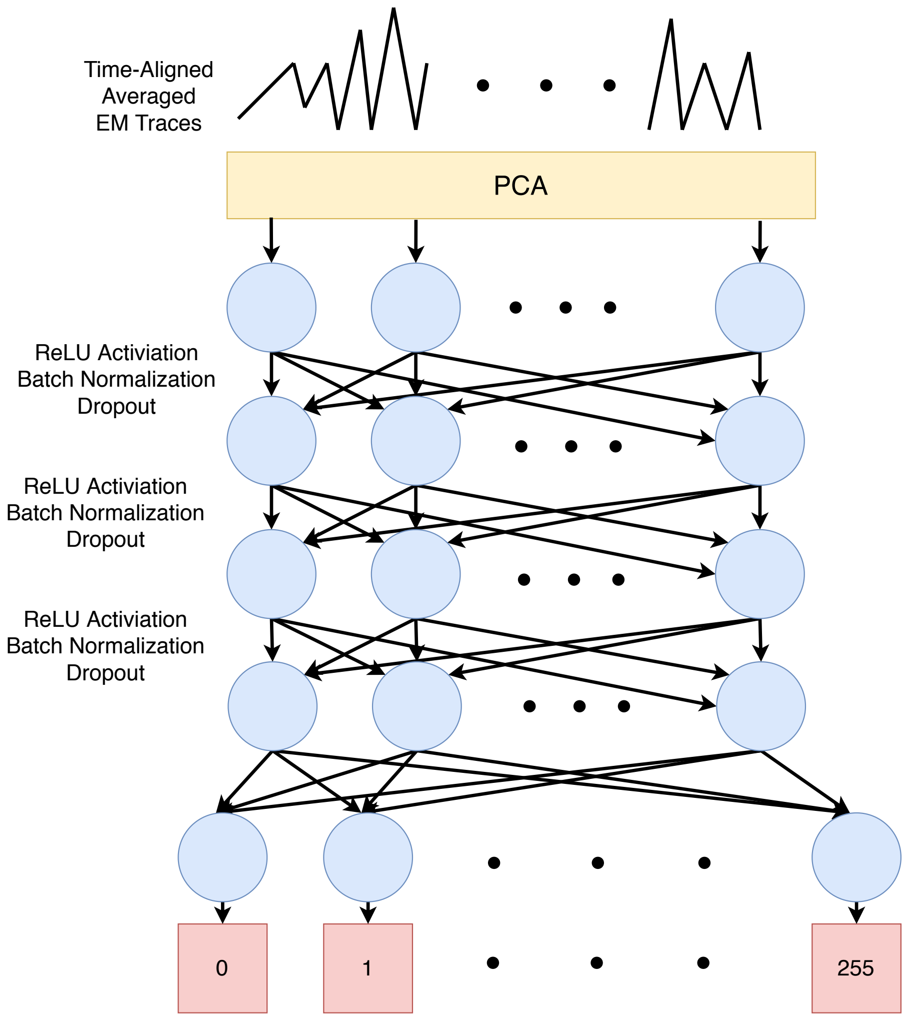


Fig. 3.2. Architecture of the proposed DNN. The network contains 3 dense layers, following each dense layer is a ReLU activation function, batch normalization, and finally a dropout layer. The final output layer provides the output class predictions - the key byte, and thus is size 256, and uses a softmax activation function.

3.3.1 DNN Architecture & Training

Figure 3.2 shows the architecture of the proposed 256-class fully-connected (FC) DNN for the EM-X-DL attack. It should be noted that the EM traces captured using Chipwhisperer are time-synchronized and hence use of a convolutional layer is

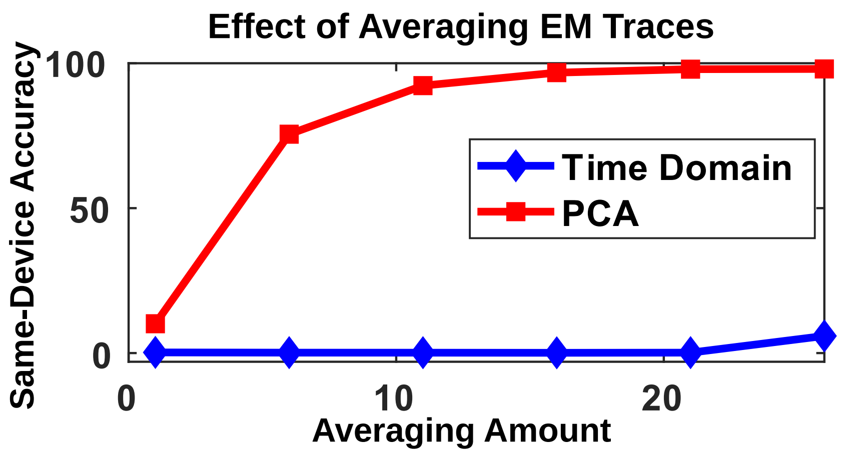


Fig. 3.3. Effect of averaging on the test accuracy of the 256-class DNN when using raw traces and PCA-transformed traces. Increasing averaging hardly allows the DNN to learn from the time-domain EM traces. With PCA used as a pre-processing step, averaging upto 20× smoothly increases the test accuracy to $> 99\%$ for the same device.

not necessary [51]. 3000 time samples for each trace were collected from the 8-bit microcontrollers running AES-128 clocked at 7.37MHz.

The DNN, implemented using Tensorflow [52], has a 3000-neuron input layer, followed by three hidden layers with 100, 1024, 512 neurons respectively, and finally the 256-neuron output layer. Rectified Linear Unit (ReLU) activation functions along with batch normalization and dropout used to achieve generalization are utilized for training the DNN. The Adam optimizer, with an initial learning rate of 0.005, which is halved whenever five consecutive training epochs pass without any validation accuracy improvement, is used for training. The effect of different hyperparameters is shown in Figure 3.4. A dropout of 0.45 is the most optimum for the first hidden layer (Figure 3.4(a)), while 1024 hidden neurons for the second hidden layer (Figure 3.4(b)) provides the maximum cross-device accuracy without overfitting to the training devices. For all the results that follow, unless otherwise mentioned, the DNN is trained with ten devices for 100 epochs with a batch size of 64.

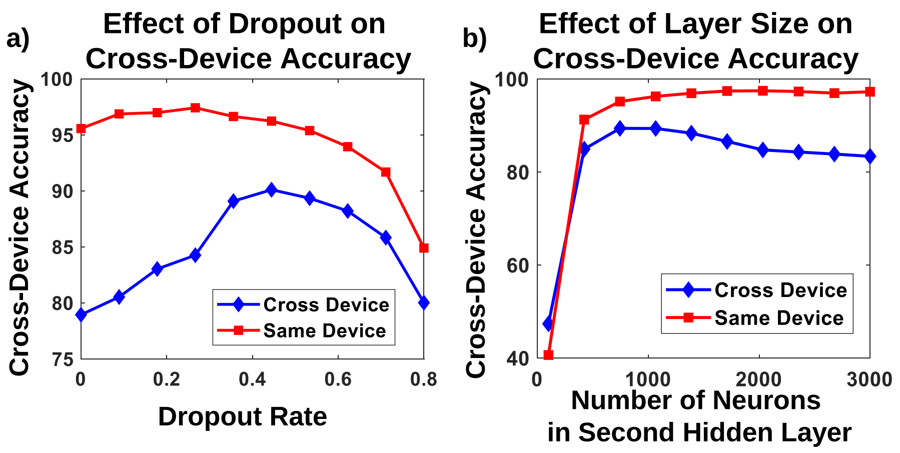


Fig. 3.4. Effect of hyperparameters on both same- and cross-device test accuracy for the PCA-DNN model. (a) Dropout between the first and second hidden layers helps prevent overfitting, maximizing cross-device accuracy at a dropout rate of 0.45. (b) Layer size also demonstrates a similar trend, and reaches maximum cross-device accuracy at ~ 1000 for the second hidden layer.

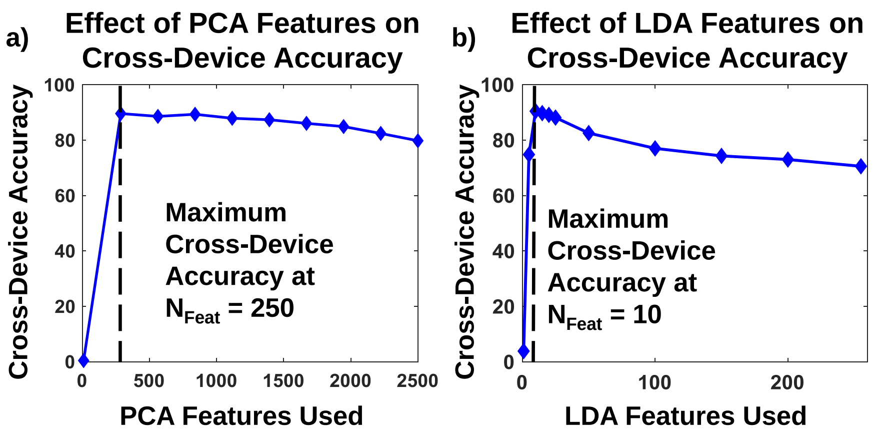


Fig. 3.5. PCA and LDA reach their respective peaks (250 and 10) with relatively few features compared to the size of the original traces (3000). As LDA features are chosen to maximize the class separation, while PCA maximizes variance, LDA is a more efficient technique for this higher dimensional data as it can train the DNN significantly faster.

Now, as the raw EM traces collected from the ten training devices (100K traces each) are fed to the DNN classifier, the validation accuracy remains low ($< 1\%$) although training accuracy increases, even after 100 epochs. Figure 3.3 (blue curve) shows the effect of averaging on the same-device (test) accuracy. Even with $20\times$ averaging, the time-domain traces shows a test accuracy of $< 1\%$, while a dimensionality reduction using PCA achieves $> 99\%$ test accuracy for the same device. Next, we will look into the effect of augmenting traces from ten training devices along with $20\times$ averaging and different pre-processing strategies on the cross-device accuracy. Note that, unless otherwise specified, cross-device accuracy refers to the average key prediction accuracy of the EM-X-DL attack across all the ten test devices.

3.3.2 Single-Trace Attack with Pre-Processing

In the previous sub-section, it was shown that the averaged time-domain EM traces (100K $\times$ 10 devices) do not train the DNN efficiently, while dimensionality reduction techniques like PCA have a significant impact in training the DNN. Here, we study the effects of PCA [51], LDA [45] on the time-domain EM traces, as well as the effects of frequency domain based processing (FFT, spectrogram [53, 54]) on the cross-device accuracy.

Dimensionality Reduction using PCA & LDA

PCA transforms the input EM trace samples to their principal sub-space where individual features maximize the variance, while LDA achieves the same effect by maximizing the inter-class separation. As seen in Figure 3.5(a, b), the optimal number of features to use in these techniques is much lower than the dimensionality of the raw trace, around 250 in the case of PCA, and a mere 10 in the case of LDA. As shown in Figure 3.6(a, b), both of these techniques lead to roughly similar cross-device accuracy, 91%. However, LDA is more efficient as it requires significantly lower training time ($< 10\times$) than PCA to achieve the same level of accuracy.

```
Input: Trace Samples from all Devices: TraceData, Number of Devices to
  select: nDev
Output: Subset of size nDev
    for $dev = 1$ : length(TraceData) do
        $\mu_1$ = mean(TraceData[dev][:,POI[1]])
        $\mu_2$ = mean(TraceData[dev][:,POI[2]])
        meanMap.append(dev, $(\mu_1, \mu_2)$)
    end for
    subset = [1]
    for $i = 1$ : nDev$-1$ do
        $\mu_{train}$ = mean(meanMap[subset])
        nextDev = argmax$_j$ $||\mu_{train}-$meanMap[j][2]$||$
        subset.add(meanMap[nextDev][1])
        meanMap.remove(nextDev)
    end for
    return subset
```

Algorithm 2: Algorithm for Device Selection

Frequency Domain Analysis using FFT & Spectrogram

Using FFT on the time-domain averaged (20×) EM traces produces an EM-X-DL attack accuracy of $\sim$ 91% (Figure 3.6(b)), which is similar to PCA/LDA. However, it requires higher training time than both PCA and LDA, and hence is not the most efficient approach. Spectrogram combines both time- and frequency-domain information and is naturally two-dimensional. Hence a 2-D CNN [55] is used for the spectrogram, which achieves a cross-device accuracy of 74.6% (Figure 3.6(b)).

3.4 EM-X-DL SCA: EFFICIENT SELECTION OF TRAINING DEVICES

As shown in the previous works [36], [37], the challenge of a ML SCA model being able to accurately classify traces collected from devices it has not been trained with, can be addressed by training with a variety of devices, so that the model does not overfit to the particular leakage pattern of one device. This remains true when using EM traces, however, many more devices are required to gain a high level of cross-device accuracy. Moreover, averaging clearly plays a key role, again increasing the number of traces required. Thus, it is of interest to be able to train using the smallest possible set of devices, reducing both the number of traces needed as well as the training time for the DNN. For this, two things must hold true: First, the choice of devices must affect the cross-device accuracy for a given number of devices, and second, there must be a way of determining whether or not to include a device for training from a small sample of traces.

3.4.1 Cross-Device Accuracy Variance

To address the first point, the effect of the subset, the EM-X-DL model is trained with a random subset of six devices, then tested against all the remaining fourteen devices. As shown in Figure 3.7, the average cross-device accuracy can vary greatly even for a set of only six devices, with accuracy ranging from 10% to 75% for different six-device combinations. This shows that there are subsets of training devices that can improve accuracy rather than simply adding more devices. However, as there are a large number of possible subsets for a given size, an algorithm is necessary to choose one such subset which results in high cross-device accuracy. Such an algorithm would then enable an attacker to gather quick measurements from a large set of devices, and determine a small subset of devices to collect a large number of traces from, for training the DNN model.

3.4.2 Bivariate POI Based Device Selection

The proposed algorithm begins by identifying two points of interest (POIs) in the traces. This can be done through any POI identification technique, here POIs are chosen as time samples which have the highest difference of means (DOM). Once the top two POIs are found, the mean $\boldsymbol{\mu_i} = (\mu_{POI1}, \mu_{POI2})$ of this POI pair is calculated across all traces for each device. Then, to construct the subset of devices for training, one device is initially chosen arbitrarily, and additional devices are added as follows: The mean POI pair of all devices currently included in the training subset, $\boldsymbol{\mu_{train}}$ is calculated. Then, the next device is chosen such that $||\boldsymbol{\mu_i} - \boldsymbol{\mu_{train}}||_2$ is maximized, where i varies over all devices not already included in the training subset. In this way, at each step, the device whose top two average POIs are furthest from the average POIs of the currently selected devices is added to the training set. This method is detailed in Algorithm 2.

Figure 3.8 shows the 2-D bivariate normal distribution of the first three devices chosen using this algorithm, along with the total distribution of all devices. With these three devices, a large portion of the distribution spanned by all the devices is covered, revealing the successful operation of the algorithm. Importantly, this algorithm also provides the desired results during training, shown in Figure 3.9, as using this algorithm to choose the training devices gives higher cross-device (EM-X-DL) accuracy for any number of devices. Additionally, training with the devices closest to the current training set, as opposed to the furthest away, results in cross-device accuracy significantly lower than the maximally different devices, and generally lower accuracy than randomly selected devices as well. These results were obtained with the proposed 256-class DNN, using $20\times$ averaging and PCA-based pre-processing. From Figure 3.9, we also see that to attain a certain cross-device accuracy, this algorithm requires between $20\% - 40\%$ fewer training devices compared to random device selection.

3.5 EM LEAKAGE ASSESSMENT & ATTACK

Once the DNN model for the EM-X-DL SCA is trained, the main goal of an attacker is to break the secret key with minimum number of traces from an identical but unseen target device. This section demonstrates an end-to-end attack strategy using the EM-X-DL model on a new device. By scanning the surface of the victim microcontroller and collecting traces at each point (seen in Figure 3.10(a)), the heatmap in Figure 3.10(b) was created by classifying the traces and determining the test accuracy for each point. As all training traces were collected from the same location

Table 3.2.
Cross-Device attack Performance of Deep Learning-based Methods for different Pre-Processing Techniques

Preprocessing	*Cross − Device Accuracy* (%)		
Technique	Minimum	Average	Maximum
Time Domain	0.28	0.37	0.45
PCA	81.27	90.72	96.77
LDA	81.21	91.52	96.42
FFT	82.40	91.07	95.50
Spectrogram	30.53	74.58	94.02

(with maximum leakage on the chip evaluated using test vector leakage assessment (TVLA)), as expected the accuracy is highest in this region, then drops off sharply further from the measurement point. Figure 3.10(c) shows the minimum traces to disclosure (MTD) from a CEMA attack over the same chip. Comparing this to the accuracy heatmap shows that the ML model can correctly classify traces that are collected from a location which has an MTD less than 250.

Now, in this virtual grid, to converge to the best location for the EM-X-DL attack on the new device, the attacker can query the EM-X-DL model with multiple averaged traces collected from the test device and **observe if the frequency of the highest predicted key byte is distinguishable from the next.** Should leakage be present, the correct key byte would be predicted more often than others. If leakage is not present, predictions would be split between several key values. Thus, the ratio between the first and second most commonly predicted value provides a measure of the attacker's confidence in the prediction. This effect is shown in Figure 3.10(d), which shows the five most common predictions for both a location of high leakage,(1,2) (left) and low leakage, (2,9) (right). Note that, with this prior knowledge of the heatmap, the attacker can also divide the chip into 4 quadrants (for this particular chip) and get the correct key from the left most quadrant with a very high confidence.

3.6 REMARKS & CONCLUSION

This chapter showed a Cross-device Deep Learning based EM (EM-X-DL) SCA attack on a symmetric key encryption engine (AES-128). Utilizing a 256-class DNN, averaged EM traces from 10 training devices with dimensionality-reduction based pre-processing (like LDA) achieves $\sim 91.5\%$ EM-X-DL single-trace (averaged) attack accuracy against another set of ten test devices. Table 3.2 summarizes the EM-X-DL attack accuracies for the different techniques studied in this chapter. An algorithm for efficient selection of training devices is proposed to speed up the profiling phase. Finally, an end-to-end attack using EM scanning is demonstrated showing that the attacker can detect the position of highest leakage on the chip using the proposed EM-X-DL model along with the secret key with high confidence.

For the future scope of this work, the end-to-end EM-X-DL attack can be more generalized by capturing traces from multiple locations across the chip, rather than a single location, for training the DNN. This would make the EM-X-DL attack much more efficient and faster as the attacker would be able to extract the key without having to detect one of the highest leakage locations on the chip.

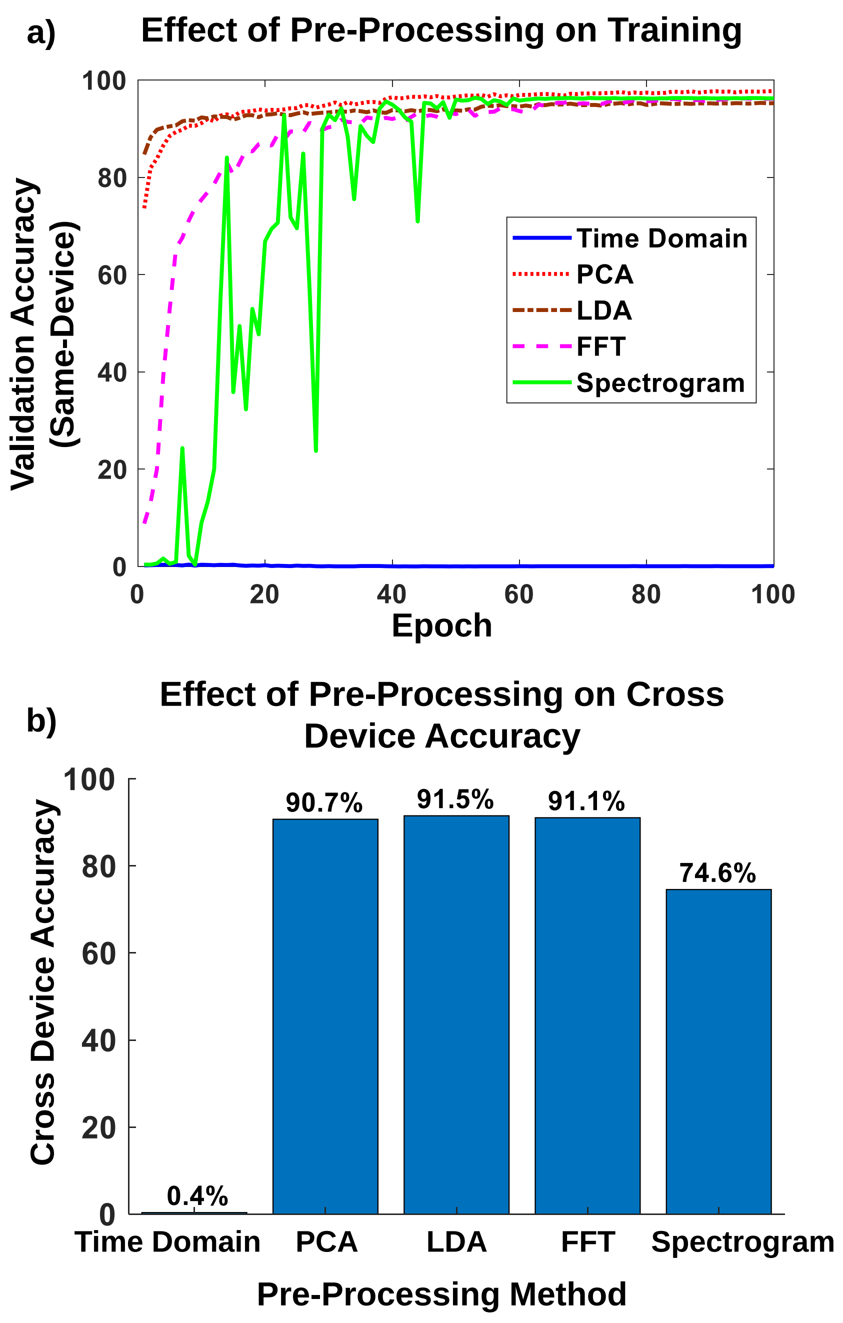


Fig. 3.6. Effect of the different pre-processing techniques on (a) the DNN training accuracy, (b) the cross-device attack (EM-X-DL) accuracy. While all the pre-processing techniques result in high validation (same-device) accuracy, PCA, LDA, FFT result in $> 90\%$ cross-device accuracy, while spectrogram yields 74.6% cross-device accuracy.

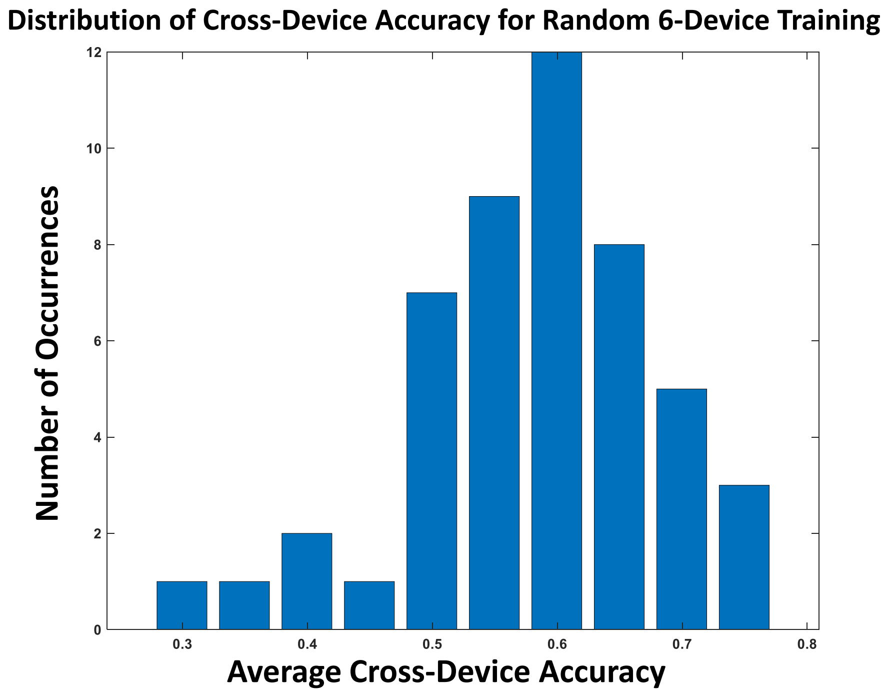


Fig. 3.7. Distribution of cross-device accuracy of the 256-class PCA-DNN trained on random subsets of 6 devices. The mean accuracy is 60%, however, the depending on the subset, it can vary significantly between $30-75\%$, highlighting the need for an intelligent selection of the training devices.

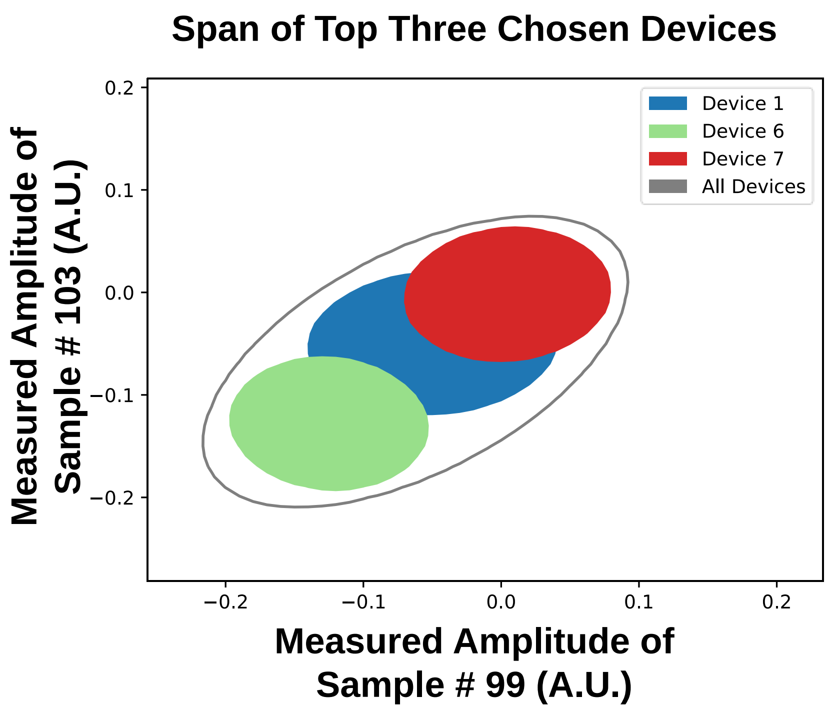


Fig. 3.8. Bivariate analysis of the first 3 devices chosen by Algorithm 2. The top three chosen devices already span a large portion of the distribution containing all devices.

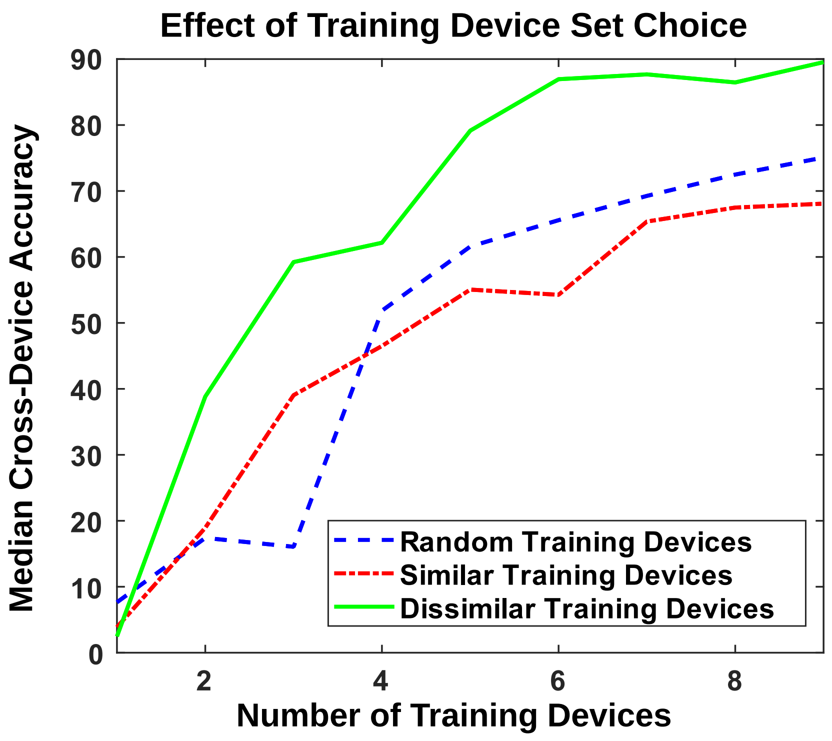


Fig. 3.9. Depending on the choice of devices used for training, cross-device accuracy varies significantly. Choosing ”dissimilar” devices by algorithm 2 gives high accuracy, while choosing ”similar” training devices yields a low cross-device accuracy. Randomly selecting devices shows slightly higher test accuracies than choosing ”similar” devices.

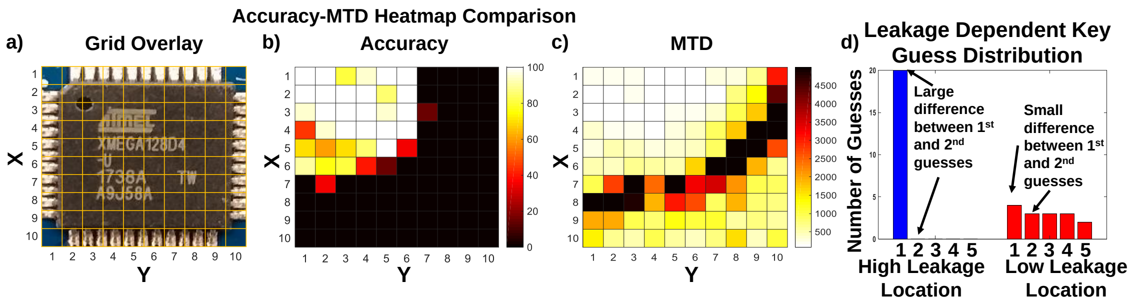


Fig. 3.10. (a) 10×10 virtual grid overlay of the chip. (b, c) Comparison of EM-X-DL model accuracy to CEMA-MTD. The ML model is able to predict with high accuracy in the region of the chip with low MTD values, however, when the MTD rises above 250, the model is unable to correctly predict the key values. (d) EM-X-DL model predictions on 20 samples from a high leakage location (1,1), and a low leakage location (9, 4) on a test device. At a location with high leakage, the frequency of the highest predicted key byte value is distinguishable from the next, demonstrating the high confidence of the attacker.

4. ADDITIONAL SIDE-CHANNEL ATTACKS

4.1 Efficient EM Attack: Power to EM Mapping

4.1.1 Introduction

In the previous chapter, a variety of methods to improve EM-ML SCA attacks were presented, particularly when dealing with low SNR measurements. This chapter presents a new method for efficiently training an EM-ML SCA model by creating a power to EM mapping function. Mapping power traces to EM traces would allow attackers to collect high SNR power traces, convert them to high SNR EM traces, and train an ML model with the generated EM traces, needing fewer measurements than collecting only EM traces. This improves the efficiency of EM-ML SCA attacks, increasing the power of an attack. Furthermore, determining the mapping between power and EM for side-channel measurements may allow for the development of more effective EM-SCA countermeasures, by designing around the transfer function to minimize EM emissions. As electromagnetic emissions fundamentally originate from changes in current/power, a mapping between the two must exist. However, finding an appropriate mapping is nontrivial and evaluating a mapping can also be challenging. In this section, two methods are investigated to find the power to EM mapping, and a method of evaluating mappings is also presented.

4.1.2 Background

4.1.3 Transfer Function Approximation Methods

Two methods have been used to generate an approximate power to EM transfer function. First is adaptive filtering, which produces the optimal finite impulse response (FIR) filter coefficients given an input trace and a corresponding target trace using least squares optimization. The second method is polynomial fitting. While adaptive filtering is limited to linear operations, polynomial curve fitting can model higher-order effects, but does not take into account dependencies between time points as a filter could. Examples of traces generated using both methods are shown in Figure 4.1, where at first glance both methods produce decent traces. One challenge in performing this mapping is deciding on a metric for evaluating different transfer functions. A straightforward approach would be to look at the MSE between the generated traces and true measured traces. While MSE can give a sense of which mapping performs better than another, it does not guarantee that the chosen mapping is useful for side channel analysis in practice. To measure the performance of a learned mapping, a separate set of EM traces are used to train an ML-SCA model, the EM traces generated by the mapping are then tested on this ML-SCA model, and the resulting accuracy of the model tested on these generated traces is used as a measure of the power to EM mapping performance. If the mapping is correct, the accuracy on the generated traces should be as good as the accuracy of true measured traces. If this accuracy is lower, then the mapping does not transform relevant side-channel information from power to EM correctly.

4.1.4 Results

Adaptive Filtering

The first method chosen was adaptive filtering. Given we expect a power to EM transformation to be mostly linear, this approach seems reasonable. Through this method, a set of power traces are used as the input traces, and 10x averaged EM traces were used as the target traces. The adaptive filtering was done in MATLAB, and produces FIR filter coefficients for a filter of a specified size. The performance of the filer is then estimated by the ML model accuracy, testing on the filter's generated traces. Figure 4.2 shows the effect of the filter size on the ML accuracy. Somewhat

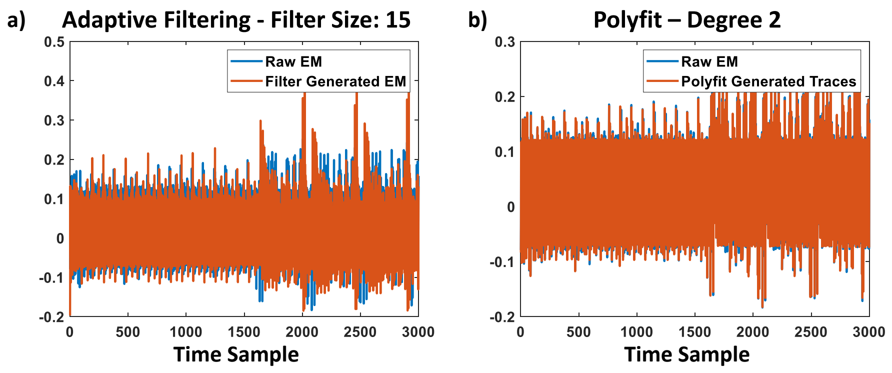


Fig. 4.1. (a) True EM trace compared to a trace generated with adaptive filtering, with a filter size of 15. (b) True EM trace compared to a trace generated with polynomial fitting and a degree of 2.

unsurprisingly, as the filter size increases, the accuracy increases, as more filter coefficients allow for more accurate filtering. Despite this, even for large sized filters, accuracy never rose above 1.1%. This is above the accuracy of a random guess (0.43%) but is drastically lower than the true EM test accuracy of 90%. So, while the adaptive filtering method does convert power traces to EM to some degree, it does not completely preserve the leakage patterns the ML model expects. One possible reason is that nonlinear effects play a more important role in side-channel leakage than initially expected, so the next method chosen was polynomial fitting, to attempt to capture these non-linear effects and preserve the true EM leakage pattern.

Polynomial Fitting

The polynomial fitting method estimates a polynomial mapping from power to EM for each time point. This results in a polynomial like the one in Figure 4.3. Unlike the adaptive filtering method, the polynomial method allows for non-linear effects to be captured. However, the polynomial method does not capture effects where one time point in EM relies on a combination of time points in the power domain. Evaluating the polynomial fitting method in the same way as the adaptive

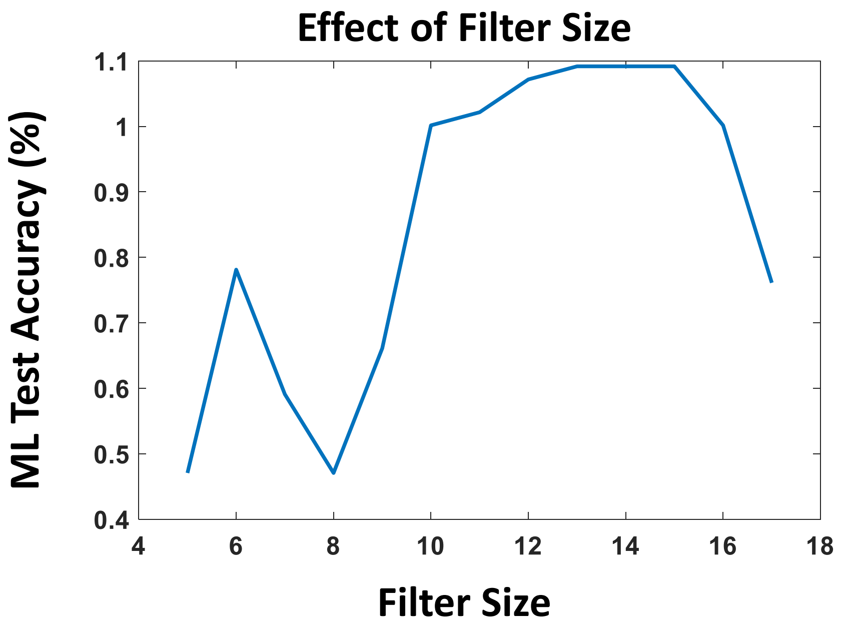


Fig. 4.2. At small filter sizes, the generated traces are unable to be classified by the ML model. As filter size increases, the accuracy begins to rise slightly above the level of a random guess, to a maximum of 1.1%.

filtering model, we see somewhat improved performance. With the polynomial fitting method, the maximum accuracy achieved is 2.9%, nearly triple the accuracy of the adaptive filtering method, but still far lower than the model's accuracy on real traces. So, while the polynomial method does perform better than adaptive filtering, it still does not completely capture the leakage patterns in the generated EM traces.

4.1.5 Conclusion

Neither adaptive filtering nor polynomial fitting fully captured the power to EM function, but both resulted in generated EM traces which could be classified with above random accuracy by a trained ML-SCA model. Given this, it seems finding a function that can map power to EM and preserve side-channel leakage is feasible. By analyzing the MSE of the two methods, there appears to be a relationship between

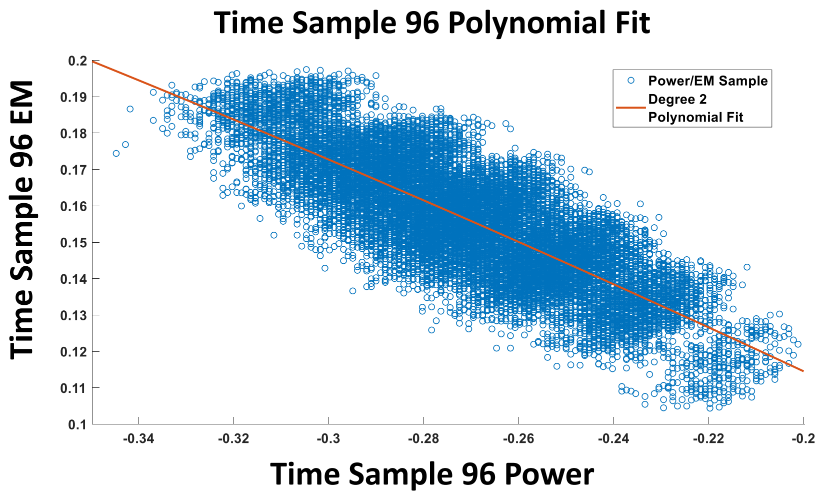


Fig. 4.3. Power vs. EM for time sample 96 and the polynomial fit for this time sample. While not perfect, polyfit results in lower MSE and better SCA-ML performance.

the MSE and ML accuracy, as we lower the error in the generated traces compared to true traces, we also raise the ML accuracy. So, by reducing this error, a usable power to EM mapping function may be found. Future efforts in this area might consider using more powerful methods to estimate the power to EM mapping, or collecting more and higher SNR power/EM traces to improve the model.

4.2 Side-Channel Resilience with High Level Synthesis

4.2.1 Introduction

In the face of side-channel threats such as `SCNIFFER` and X-EM-DL SCA, many countermeasures have been proposed to reduce the efficacy of side-channel attacks.

One such countermeasure is to use high level synthesis (HLS) to produce SCA resistant implementations. The directives one can provide to an HLS system can drastically impact the side-channel leakage patterns, even if the underlying high level functional description is unchanged. However, even if the leakage changes, this does not imply the system is any more secure against side-channel analysis. Thus, a detailed white-box analysis of the digital logic generated by the HLS system is needed to determine the security of the system.

4.2.2 Background

Side-Channel Countermeasures

Since the introduction of side-channel analysis, there have been numerous countermeasures proposed to reduce the efficiency of attacks using side channel information. These countermeasures can be broadly classified into three categories: Algorithm level countermeasures, architecture level countermeasures, and circuit level countermeasures. Broadly, algorithm level countermeasures change a specific algorithm in a way that side channel leakage is lower. Architecture level countermeasures change the implementation of an algorithm to reduce side channel leakage. Finally, circuit level countermeasures change the physical hardware on which a cryptographic algorithm operates. Algorithm level countermeasures are specific to an algorithm, such as masking for AES or blinding for RSA. Countermeasures at the architecture level change the implementation of an algorithm, without changing the logic of the algorithm or the physical hardware. Examples of architecture level countermeasures are the insertion of dummy operations or clock frequency shifting, both of which cause misalignment from one trace to another. Circuit level countermeasures are the most general countermeasures, and are not specific to any algorithm or implementation. Circuit level countermeasures change physical hardware - either by changing the implementation of logic cells, or by adding circuitry - to reduce side channel leakage. One method of changing logic cells is dual-rail logic, which has a variety of implementations, such as those in [56] and [57], where logic gates are designed to consume constant power regardless of the data processed. A second method is to reduce the leakage of the whole cryptographic block by forcing the block to consume constant power. This approach is demonstrated in [58], and is generally lower overhead than changing logic cells, and also lower overhead than many other countermeasure types.

High Level Synthesis Systems

High level synthesis (HLS) in digital system design is a process to take a high level description of a system, for example an algorithm written in C++, and generating synthesizeable hardware description language(HDL) code. Using high level synthesis, designers can create hardware implementations without needing to write HDL code, and can use algorithms as written in high level languages. This process does have some drawbacks, as generated code is often less efficient in both hardware resources used and execution time compared to handwritten HDL. In order to help improve this, HLS systems allow the high level code to be annotated with "hints" to the synthesis system to effect how the generated code is constructed. Using these options, designers can control things like loop unrolling, pipelining, and register use. The options provided by HLS can cause effects similar to architecture level countermeasures. For instance, changing loop unrolling has an effect similar to a shifting countermeasure.

4.2.3 Initial Leakage Assessment

To understand how HLS options effect side channel leakage, a high level AES implementation [59] was used as the source for the HLS, and the generated Verilog code was then further mapped to run on an Atrix-7 FPGA. Once again, the Chipwhisperer [24] platform was used to collect power traces from the target FPGA while it performed AES encryption.

To begin the analysis, TVLA [14] was used to gain insight to which HLS options have the largest effect on side channel leakage. First, to establish a baseline, HLS was used with default settings. The default settings resulted in a fully serial implementation of AES, with each byte being operated on one at a time. Using 1000 traces for TVLA (500 fixed plaintext traces, 500 random plaintext traces) this fully serial implementation had a maximum t-value of 7.2. This is a somewhat low leakage level, particularly since in a serial implementation, there is very little algorithmic noise. With this as a baseline, four variants of the AES implementation were created through varying HLS options. These initial variants unroll different loops in the AES source code. From a hardware perspective, this has the effect of changing the number of lookup tables created for the SBox operation in AES. For instance, the fully serial variant uses 2 SBox blocks, one for the SubBytes step in a round of AES, and one for the key expansion step of AES. Meanwhile, in the fully unrolled variant, there are 20 SBox units, one for each of the 16 bytes during the SubBytes step, and four for the key expansion step. The remaining variants fall between these two extremes.

Now, looking at the TVLA analysis of the variants, all unrolled variants had a much higher t-value, ranging from 18.7 to 24.4, all much higher than the serial implementation which contained no unrolling. Figure 4.4 shows the TVLA results for all variants. Table 4.1 summarizes the results, along with the performance (cycle count per encryption) and overhead (FPGA resources used).

While this seems to imply the unrolled variants are far less secure than the original variant, TVLA giving a high t-value does not imply the leakage is exploitable. So, to determine the true security of the variants, CPA [13] was used to recover the key, and the MTD was recorded for each variant. CPA requires a leakage model to be used, and the model chosen is a common model for hardware implementations of AES, the last-round state difference model. Under this model, leakage is proportional to the number of bits in the AES state that change from the second to last round to the last round (which is the ciphertext). Performing CPA with a total of 100,000 traces for each variant, the key was only recovered for two of the five variants. For the default, serial variant the MTD was 31,000 traces, and for variant 2 the MTD was $\sim$ 400 traces. For the other variants, even though the t-value was $>$ 20, the key was not recovered even after 100,000 traces. Given that the t-value and CPA result do not align with expectations, this suggests the other variants are leaking, but under a different model. To determine the appropriate leakage model, one must now

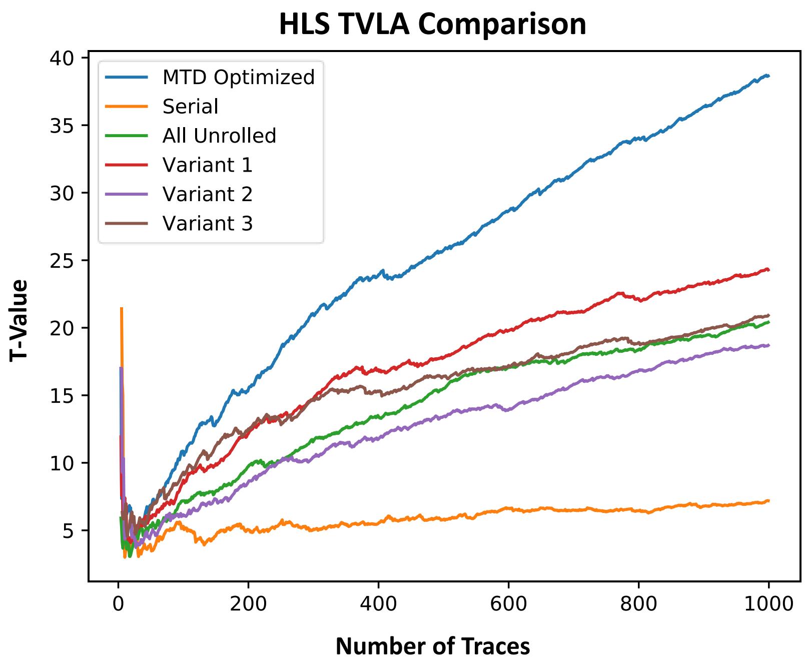


Fig. 4.4. Incremental TVLA for all variants investigated. The initial serial implementation shows low leakage under TVLA, and the dynamic implementation shows high leakage, however TVLA does not necessarily correlate to the true security of an implementation.

perform a white-box analysis of the variants to determine a leakage location that is exploitable, along with an appropriate leakage model.

4.2.4 White-Box Analysis

To determine at what point in the AES algorithm a particular implementation may cause leakage, it is necessary to look into the HDL code defining the system. By looking into the HDL code, it is easy to see that the two variants which CPA was able to break leak in the expected last round model. The remaining variants do not leak under the same model, as data flows through registers in a different way. Indeed, the

Table 4.1.
Summary of AES-128 implementations produced through HLS. HLS options have a large effect on both the size and speed of an implementation as well as the side channel leakage. The dynamic implementation combines several variants, leading to the much higher resource use.

	Serial	All Unrolled	Variant 1	Variant 2	Variant 3	Dynamic
TVLA	7.2	20.4	24.4	18.7	20.9	8.0
LUT	1026	2784	1794	1430	1650	3667
FF	1797	2622	2041	1746	2426	4903
Cycle Count	707	32	185	25	512	~250

other variants, while they do have a high t-value, do not leak in an exploitable way in the last round.

Given the last round leakage model was not appropriate for many of the variants, the main alternative is a first round model. In this model, under an assumption of a zero register initial condition, CPA succeeded after 94,000 traces (Figure 4.5a). Seeing a success, further analysis was done on the HDL code, and it was determined the initial state of the relevant register was not zero, but a fixed constant 0x63, which is the SBox output for a zero input. Further, this value was stored in the relevant register every encryption, making the leakage easy to predict. Using this corrected model, CPA then succeeded with only 2,000 traces(Figure 4.5b). This variant demonstrates the importance of using the correct leakage model, as the attack needs 47× fewer traces to recover the secret key. Additionally, we see that the loop unrolling HLS options appear to be unable to meaningfully reduce exploitable leakage.

4.2.5 Dynamic Implementation

In an effort to reduce leakage effectively with HLS, options were changed such that the loop unrolling changes from one encryption to another. While this does incur large overhead, the changing leakage pattern makes attacks like CPA more

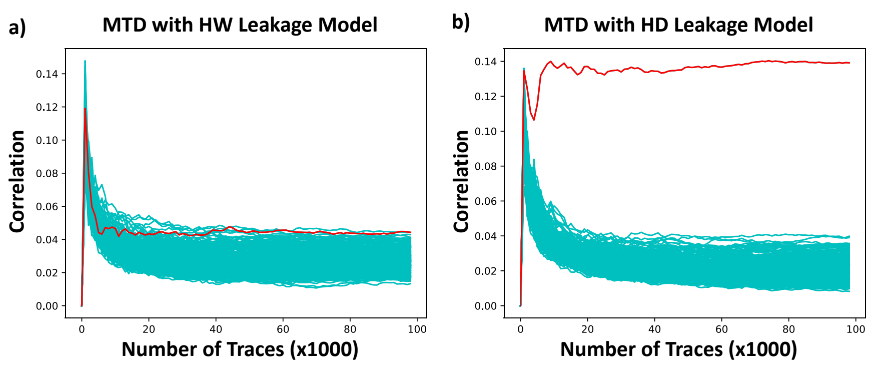


Fig. 4.5. (a) CPA using a HW leakage model. The key is recoverd after 94,000 traces. (b) CPA using a HD model. The key is now recovered in only 2,000 traces.

difficult. This is due to the fact that CPA operates on each time sample individually, so if the leakage does not occur at the same point in each trace, CPA will not succeed. However if there are a small, finite number of leakage patterns, then MTD is only increased by a factor equal to the number of leakage patters, as a particular pattern can be easily identified automatically, and a subset of traces which all have the same leakage pattern can be created. Without separating traces, other leakage patterns act as noise - increasing the MTD by a factor more than the number of patterns. As a demonstration, an AES implementation was created that switches between four variants. Initially CPA was performed without separating the traces by variant, and the CPA attack succeeded after 81,000 traces, using the correct leakage model, as shown in Figure 4.6 a). Next, the traces were separated by the leakage variant. This separation was done by correlating a new trace with a template, and a correlation of > 0.35 correctly identified all traces coming from the same variant as the template. By creating this subset, CPA now succeeded in 5,000 traces, as seen in Figure 4.6 b). In total, to get the required number of traces of a particular variant 20,000-25,000 total traces were needed. So, while to a naive attacker, using a variety of implementations greatly increases the MTD, to an intelligent attacker, the MTD increase is not nearly

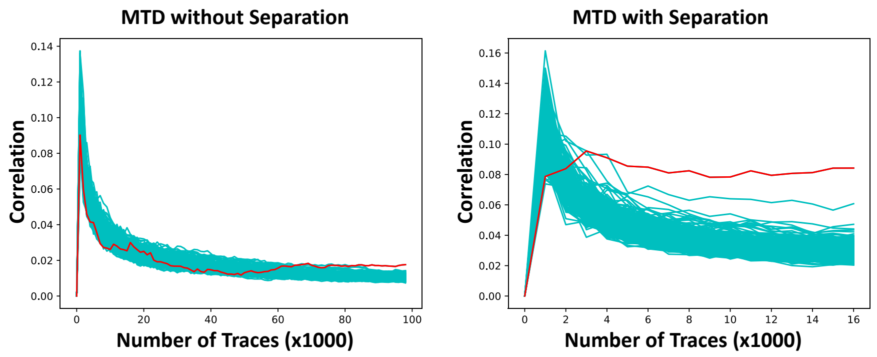


Fig. 4.6. (a) CPA performed on the dynamic implementation of AES, with the attack succeeding at 81,000 traces. (b) After separating traces by implementation, CPA succeeds after 5,000 traces.

as great. Further, the overhead cost of implementing multiple variants of the same algorithm is very high, as seen in Table 4.1, and the security improvements to not justify this level of overhead.

4.2.6 Conclusion

In the process of analysing methods of using HLS to improve side-channel security for hardware AES implementations, a number of variants were produced and compared. Starting with TVLA, and further investigating with different leakage models for CPA it is clear that HLS options can have an effect on side channel security. However this security comes with a large overhead relative to the added security.

While HLS is a powerful tool for digital circuit designers, and provides a large variety of options to control HDL code generation, these built in options do not give the capability to efficiently increase the side-channel security of a design. While they can be used to improve side-channel attack resistance, the cost to do so is far higher than existing countermeasures.

4.3 Side-Channel Analysis on Future ML Systems: Neural Cryptography with Generative Adversarial Networks

4.3.1 Introduction

Neural cryptography uses artificial neural networks to learn a cryptographic operation, that is two functions, one that takes a plaintext and encrypts it, and another function which takes the encrypted data and recovers the original plaintext. Generally, the functions also have access to a shared secret key to aid in keeping the encrypted data secure.

4.3.2 Background and Related Works

The concept of neural encryption was introduced in [60], where adversarial training was used to create the symmetric encryption and decryption functions (Alice and Bob, respectively), while an adversary network (Eve) was trained to recover the plaintext with only access to the encrypted ciphertext, not the secret key. Training all three networks simultaneously, Alice and Bob are attempt to communicate (minimize Bob's reconstruction error) while maintaining secrecy (maximizing eve's reconstruction error). This forces Alice and Bob to learn a mapping that relies on the secret key. While [60] showed this training was possible, [61] brought up a number of issues with the initial implementation that prevent this method from producing a practical cryptographic algorithm.

4.3.3 GAN Cryptography Security Analysis

The security of a neural cryptography system must be considered from both a traditional cryptography viewpoint and the machine learning viewpoint. In the traditional cryptography viewpoint, the trained encryption and decryption models should at least demonstrate good diffusion and confusion, while from an ML standpoint, an attacker should not be able to train a model like Eve to any meaningful level of accuracy. This analysis has only been briefly touched on by previous works.

ML attack resistance

Neural cryptography obviously has some inherent resistance to machine learning attacks, as it is trained via adversarial learning, but is not necessarily totally resistant. During training, the "Alice" and "Bob" networks can continuously update their communication to make it difficult for "Eve" to decode. However, in a real attack scenario, the encryption and decryption networks ("Alice" and "Bob") would be fixed, and the attacker could train for as long as needed on these fixed models. To determine if a trained model is truly resistant against ML attacks, this real-world attack was performed after training the neural encryption system. Figure 4.7 shows the bits incorrect per encryption for both the encryption system and the attacker. Once the encryption error stabilized (at about 800 iterations), the encryption and decryption models were locked, and the attacker was allowed to continue training. This shows that the attacker's error while training the encryption/decryption models is an overly optimistic estimate of the system's security. At the end of training, the attacker had 12 out of 32 bit incorrect, while after the attacker could train on the fixed encryption model, the attacker now only had 10 out of 32 bits incorrect per encryption. Optimistic or not, an attacker still is able to correctly predict 4-6 bits of the 32 bit plaintext with access to only the ciphertext. Ideally, the attacker would get 16 out of 32 bit incorrect, showing accuracy no better than guessing.

Diffusion and Confusion Analysis

Beyond resistance to machine learning based attacks, for neural encryption to be usable, it must also be resistant to classical cryptanalyis. For an initial analysis, the confusion and diffusion of the learned encryption/decryption models were analyzed. During the initial analysis, the issue of non-binary ciphertexts came up quickly, as no

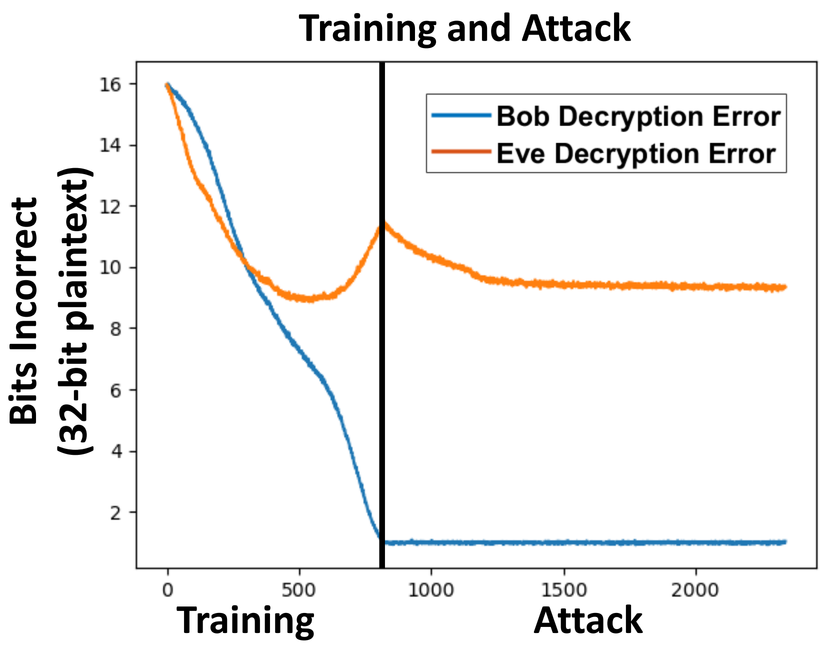


Fig. 4.7. During the training phase, both Bob and Eve are able to reduce decryption error at first, then Alice and Bob begin to use the secret key, and Eve's decryption error increases. Once training ends, the attack phase begins. Eve is able to further reduce the decryption error during the attack, but not to the level of Bob.

matter how many key or plaintext bits changed, the number of bit flips (that is, sign changes) in the ciphertext never grew above four. This again shows the issue that information is being stored in the ciphertext by small variations, not sign changes as intended. Additionally, the trend of very few bit flips was not the only issue. For some training runs, the number of sign changes in the ciphertext was equal to the number of bit flips in the plaintext. This presents a different problem, as now the encryption is extremely insecure, and performing an encryption with the same key twice would likely reveal most if not all of the key.

4.3.4 Real-World Limitations

One of the biggest challenges in this variety of neural cryptography is the encoding of the ciphertext. Under the formulation of neural cryptography above, the ciphertext is not required to be binary. While this does not mean the encryption/decryption engines learned by the network are unusable, they would be extremely inefficient, as each bit of ciphertext would need to be represented by a 32/64 bit floating point representation of the ciphertext bit (depending on the datatype used in training). This non-binary nature is seen in Figure 4.8 a). To counter this, two methods were proposed in [61]. First, to add a term to the loss function which penalizes ciphertext values that are far from the limits (-1, +1). The second method proposed a "hard tanh" activation function placed before the ciphertext, to push values towards the limits. In practice, it was found that the first of these methods had the intended effect on ciphertext values, as seen in Figure 4.8b, as we see the distribution of ciphertext values is split between +1 and -1. However, the "hard tanh" method does not achieve the desired result, and the distribution of ciphertext values was constrained to the linear part of the "hard tanh" function, centered at 0. Even though the addition to the loss function seems to achieve the desired results at first glace, the issue of non-binary ciphertext is not actually resolved. While the ciphertext values are grouped around the extremes, by mapping values < 0 to -1 and values > 0 to +1, the trained decryption engine fails to recover the plaintext. From this, we can conclude that information is being stored in the small variations around +/-1, not the difference between the two extremes. So, while there is no method of converting the ciphertext to be truly binary, neural encryption is still possible, but a much larger ciphertext must be used to transfer a given amount of data.

In addition to the issue of the ciphertext being non-binary, an additional issue with neural encryption is the inherent error rate of the system. Even without any constraints on security or ciphertext being binary, recovery of plaintexts is not guaranteed, as it would be in a typical encryption/decryption scheme. The baseline error

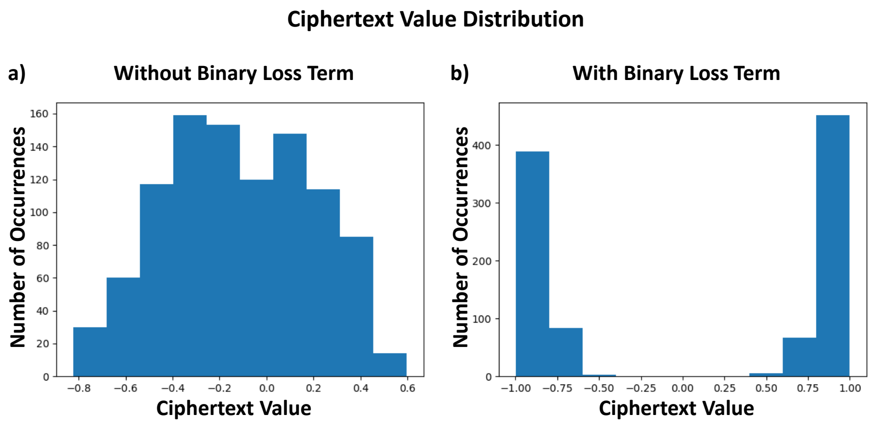


Fig. 4.8. (a) Distribution of ciphertext values for 1000 encryptions when using the standard loss function and hard tanh activation functions. (b) Distribution of ciphertext values for 1000 encryptions with binary loss term and normal tanh activation functions. While ciphertexts appear more binary, information is stored in small variations - not -1 to +1 changes.

rate is 0.1 out of 32 bits incorrect per encryption, but adding additional constraints causes the error rate to increase. For example, by requiring the ciphertext values to have binary distributions, the error increases to about 1 bit out of 32 incorrect per encryption. Like the issue of the binary ciphertext, this does not make neural cryptography unusable, but adds overhead, as error correction mechanisms will now be required.

4.3.5 Conclusion

Neural encryption could provide devices with a method of performing encryption with the same hardware as is used for neural network prediction. With the original goal of analyzing the effects of side-channels on the security of neural encryption, it was quickly found that a detailed analysis of the fundamental security of neural encryption was lacking. In the process of developing a neural encryption system to evaluate security, some problems arose. First, the issue that ciphertexts are not truly binary, which makes communication of ciphertexts difficult. Second, the error rate of neural encryption which again makes neural encryption less appealing. Setting aside these issues, neural encryption was found to have poor confusion and diffusion, making it vulnerable to simple cryptanalysis. Essentially, neural cryptography performs steganography rather than encryption, not making good use of the shared secret key. Thus, there is not a clear target for attacks against neural cryptography systems - traditional or side-channel. Until these issues are addressed, neural encryption is unlikely to be a viable alternative to current encryption schemes, and side-channel analysis of these methods would be premature. However, homomorphic encryption - a different fast-growing field in cybersecurity would perhaps be a better target for side channel analysis.

5. CONCLUSION

This thesis has presented several advanced EM side-channel attacks, enabling designers to easily evaluate cryptographic implementations against the most powerful side channel attacks. These attacks work to address the SNR of measurements, which directly effects the efficiency of an attack.

First, `SCNIFFER` addressed the algorithmic noise component of the SNR by providing a platform to quickly locate a position of high leakage. Such points have low algorithmic noise and are easily attacked through methods like CEMA. `SCNIFFER` is able to find such points efficiently through a gradient-ascent based search algorithm. Searching intelligently in this way reduces the number of measurements in an attack by a factor of N for and $N \times N$ search space.

Next, a practical machine learning based EM attack, EM-X-DL is presented. This attack works to address issues caused by high measurement noise by a combination of pre-processing and efficient choice of training devices. the EM-X-DL attack achieved 91.5% single (averaged) trace accuracy on traces from unseen test devices, despite lower SNR than equivalent power measurements.

Several topics are covered in the last chapter, starting with power to EM mapping to improve EM-ML-SCA attacks, then moving to an investigation into the effect of high-level synthesis on side channel security, then ending with an security analysis of neural cryptography.

By addressing issues relating to the SNR of EM measurements, advanced attack methods were developed. These attacks allow designers to evaluate the security of cryptographic implementations as well as the effectiveness of SCA countermeasures. Additionally, the low cost of the systems and tools used make the attacks accessible to researchers developing new implementations and countermeasures.

REFERENCES

REFERENCES

[1] P. Kocher, J. Jaffe, and B. Jun, “Differential power analysis,” in *Advances in Cryptology — CRYPTO’ 99*, ser. Lecture Notes in Computer Science, M. Wiener, Ed. Springer Berlin Heidelberg, 1999, pp. 388–397.

[2] K. Gandolfi, C. Mourtel, and F. Olivier, “Electromagnetic analysis: Concrete results,” in *Cryptographic Hardware and Embedded Systems — CHES 2001*, ser. Lecture Notes in Computer Science, C. K. Koc, D. Naccache, and C. Paar, Eds. Springer Berlin Heidelberg, 2001, pp. 251–261.

[3] J.-J. Quisquater and D. Samyde, “ElectroMagnetic analysis (EMA): Measures and counter-measures for smart cards,” in *Smart Card Programming and Security*, ser. Lecture Notes in Computer Science, I. Attali and T. Jensen, Eds. Springer Berlin Heidelberg, 2001, pp. 200–210.

[4] D. Das, M. Nath, B. Chatterjee, S. Ghosh, and S. Sen, “Stellar: A generic em side-channel attack protection through ground-up root-cause analysis,” in *2019 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. Los Alamitos, CA, USA: IEEE Computer Society, 2019, pp. 11–20. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/HST.2019.8740839

[5] D. Genkin, L. Pachmanov, I. Pipman, and E. Tromer, “Stealing keys from pcs using a radio: Cheap electromagnetic attacks on windowed exponentiation,” in *Cryptographic Hardware and Embedded Systems – CHES 2015*, T. Güneysu and H. Handschuh, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 207–228.

[6] ——, “ECDH key-extraction via low-bandwidth electromagnetic attacks on PCs,” in *Proceedings of the RSA Conference on Topics in Cryptology - CT-RSA 2016 - Volume 9610*. Springer-Verlag New York, Inc., 2016, pp. 219–235. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-29485-8_13

[7] A. Matthews, “Low cost attacks on smart cards,” Next Generation Security Software Ltd., Tech. Rep., 2006. [Online]. Available: https://pdfs.semanticscholar.org/1b5a/48426397d3e5d7d56b35ffe2d8456e29834e.pdf

[8] T. Kasper, D. Oswald, and C. Paar, “EM side-channel attacks on commercial contactless smartcards using low-cost equipment,” in *Information Security Applications*, ser. Lecture Notes in Computer Science, H. Y. Youm and M. Yung, Eds. Springer Berlin Heidelberg, 2009, pp. 79–93.

[9] J. Danial, D. Das, S. Ghosh, A. Raychowdhury, and S. Sen, “Scniffer: Low-cost, automated, efficient electromagnetic side-channel sniffing,” *IEEE Access*, vol. 8, pp. 173 414–173 427, 2020.

[10] E. Ronen, A. Shamir, A.-O. Weingarten, and C. OFlynn, “IoT goes nuclear: Creating a ZigBee chain reaction,” in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 195–212. [Online]. Available: http://ieeexplore.ieee.org/document/7958578/

[11] D. Agrawal et al., “The EM side—channel(s),” in *CHES 2002*, 2003, pp. 29–45.

[12] C. Ramsay and J. Lohuis, “Tempest attacks against AES,” Fox-IT, Tech. Rep., 2017. [Online]. Available: https://hardwear.io/document/slides-craig-ramsay.pdf

[13] E. Brier, C. Clavier, and F. Olivier, “Correlation power analysis with a leakage model,” in *CHES 2004*, 2004, pp. 16–29.

[14] G. Becker, J. Cooper, E. DeMulder, G. Goodwill, J. Jaffe, G. Kenworthy, T. Kouzminov, A. Leiserson, M. Marson, P. Rohatgi *et al.*, “Test vector leakage assessment (tvla) methodology in practice,” in *International Cryptographic Module Conference*, vol. 1001, 2013, p. 13.

[15] S. Mangard, “Hardware countermeasures against DPA – a statistical analysis of their effectiveness,” in *Topics in Cryptology – CT-RSA 2004*, ser. Lecture Notes in Computer Science, T. Okamoto, Ed. Springer Berlin Heidelberg, 2004, pp. 222–235.

[16] Y. Liu and B. Ravelo, “Fully time-domain scanning of EM near-field radiated by RF circuits,” *Progress In Electromagnetics Research*, vol. 57, pp. 21–46, 2014.

[17] V. Lomné, P. Maurine, L. Torres, T. Ordas, M. Lisart, and J. Toublanc, “Modeling time domain magnetic emissions of ICs,” in *Integrated Circuit and System Design. Power and Timing Modeling, Optimization, and Simulation*, ser. Lecture Notes in Computer Science, R. van Leuken and G. Sicard, Eds. Springer Berlin Heidelberg, 2011, pp. 238–249.

[18] L. Sauvage, S. Guilley, and Y. Mathieu, “Electromagnetic radiations of fpgas: High spatial resolution cartography and attack on a cryptographic module,” *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 2, no. 1, pp. 1–24, 2009.

[19] J. Heyszl, D. Merli, B. Heinz, F. De Santis, and G. Sigl, “Strengths and limitations of high-resolution electromagnetic field measurements for side-channel analysis,” in *International Conference on Smart Card Research and Advanced Applications*. Springer, 2012, pp. 248–262.

[20] J. Heyszl, S. Mangard, B. Heinz, F. Stumpf, and G. Sigl, “Localized electromagnetic analysis of cryptographic implementations,” in *Topics in Cryptology – CT-RSA 2012*, ser. Lecture Notes in Computer Science, O. Dunkelman, Ed. Springer Berlin Heidelberg, 2012, pp. 231–244.

[21] V. V. Iyer and A. E. Yilmaz, “An adaptive acquisition approach to localize electromagnetic information leakage from cryptographic modules,” in *2019 IEEE Texas Symposium on Wireless and Microwave Circuits and Systems (WMCS)*. IEEE, 2019, pp. 1–6.

[22] EM probe station: Electromagnetic analysis solution. [Online]. Available: https://www.riscure.com/product/em-probe-station/

[23] Creality3d ender-3 3d printer economic ender DIY KITS. [Online]. Available: https://www.creality3d.shop/products/creality3d-ender-3-pro-high-precision-3d-printer

[24] C. O'Flynn and Z. Chen, "ChipWhisperer: An open-source platform for hardware embedded security research," in *COSADE 2014*, 2014, pp. 243–260.

[25] O. Standaert, E. Peeters, G. Rouvroy, and J. Quisquater, "An overview of power analysis attacks against field programmable gate arrays," *Proceedings of the IEEE*, vol. 94, no. 2, pp. 383–394, 2006. [Online]. Available: https://ieeexplore.ieee.org/document/1580507

[26] D. B. Roy, S. Bhasin, S. Guilley, A. Heuser, S. Patranabis, and D. Mukhopadhyay, "CC meets fips: A hybrid test methodology for first order side channel analysis," *IEEE Transactions on Computers*, vol. 68, no. 3, pp. 347–361, 2018.

[27] ——, "Leak me if you can: Does tvla reveal success rate," IACR Cryptology ePrint Archive, Tech. Rep., 2016. [Online]. Available: https://eprint.iacr.org/2016/1152

[28] R. Benadjila, V. Lomné, E. Prouff, and T. Roche, "Secure AES128 on ATMEGA8515," Laboratory of Embedded Security, ANSSI, Tech. Rep., 2017. [Online]. Available: https://github.com/ANSSI-FR/secAES-ATmega8515

[29] P. C. Kocher, "Timing attacks on implementations of diffie-hellman, RSA, DSS, and other systems," in *CRYPTO '96*, 1996.

[30] D. Das, S. Maity, B. Chatterjee, and S. Sen, "Enabling covert body area network using electro-quasistatic human body communication," *Scientific reports*, vol. 9, no. 1, pp. 1–14, 2019.

[31] S. Maity, N. Modak, D. Yang, S. Avlani, M. Nath, J. Danial, D. Das, P. Mehrotra, and S. Sen, "A 415 nw physically and mathematically secure electro-quasistatic hbc node in 65nm cmos for authentication and medical applications," in *2020 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 2020, pp. 1–4.

[32] S. Chari, J. R. Rao, and P. Rohatgi, "Template attacks," in *CHES 2002*, 2003, pp. 13–28.

[33] T. B. et al., "Efficient Template Attacks Based on Probabilistic Multi-class Support Vector Machines," in *Smart Card Research & Advanced Applications*, 2013.

[34] E. Prouff et al., "Study of deep learning techniques for side-channel analysis and introduction to ASCAD database," 2018. [Online]. Available: http://eprint.iacr.org/2018/053

[35] B. Chatterjee, D. Das, S. Maity, and S. Sen, "Rf-puf: Enhancing iot security through authentication of wireless nodes using in-situ machine learning," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 388–398, 2019.

[36] D. Das et al., "X-DeepSCA: Cross-device deep learning side channel attack," in *DAC 2019*. ACM, 2019, pp. 134:1–134:6. [Online]. Available: http://doi.acm.org/10.1145/3316781.3317934

[37] S. Bhasin et al., "Mind the portability: A warriors guide through realistic profiled side-channel analysis," 2019. [Online]. Available: http://eprint.iacr.org/2019/661

[38] M. Carbone et al., "Deep learning to evaluate secure RSA implementations," pp. 132–161, 2019.

[39] D. Das, S. Maity, S. B. Nasir, S. Ghosh, A. Raychowdhury, and S. Sen, "High efficiency power side-channel attack immunity using noise injection in attenuated signature domain," in *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2017, pp. 62–67.

[40] D. Das, M. Nath, S. Ghosh, and S. Sen, "Killing em side-channel leakage at its source," in *2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2020, pp. 1108–1111.

[41] A. Ghosh, D. Das, J. Danial, V. De, S. Ghosh, and S. Sen, "An em/power sca resilient aes-256 with synthesizable signature attenuation using digital-friendly current source and ro-bleed based integrated local feedback and global switched mode control," in *2021 IEEE International Solid-State Circuits Conference-(ISSCC)*, 2021.

[42] D. Das, J. Danial, A. Golder, S. Ghosh, A. R. Wdhury, and S. Sen, "Deep learning side-channel attack resilient aes-256 using current domain signature attenuation in 65nm cmos," in *2020 IEEE Custom Integrated Circuits Conference (CICC)*, 2020, pp. 1–4.

[43] J. Danial, D. Das, S. Ghosh, A. Raychowdhury, and S. Sen, "EM-X-DL: Efficient cross-device deep learning side-channel attack with noisy em signatures," *arXiv preprint*, 2020.

[44] M. O. Choudary and M. G. Kuhn, "Efficient, portable template attacks," vol. 13, no. 2, pp. 490–501, 2018.

[45] M. Renauld et al., "A formal study of power variability issues and side-channel attacks for nanoscale devices," in *EUROCRYPT 2011*, ser. Lecture Notes in Computer Science. Springer, 2011, pp. 109–128.

[46] L. Lerman et al., "Power analysis attack: An approach based on machine learning," vol. 3, 2014.

[47] H. Maghrebi et al., "Breaking cryptographic implementations using deep learning techniques," 2016. [Online]. Available: http://eprint.iacr.org/2016/921

[48] D. P. Montminy et al., "Improving cross-device attacks using zero-mean unit-variance normalization," 2013. [Online]. Available: https://doi.org/10.1007/s13389-012-0038-y

[49] R. Gilmore et al., "Neural network based attack on a masked implementation of AES," in *HOST 2015*, 2015, pp. 106–111.

[50] E. Cagli et al., "Convolutional neural networks with data augmentation against jitter-based countermeasures," in *CHES 2017*, 2017.

[51] A. Golder et al., *Practical Approaches Towards Deep-Learning Based Cross-Device Power Side Channel Attack*, 2019.

[52] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," p. 21, 2016.

[53] C. Rechberger and E. Oswald, "Practical template attacks," in *Information Security Applications*, 2005.

[54] G. Yang et al., "Convolutional neural network based side-channel attacks in time-frequency representations," in *Smart Card Research and Advanced Applications*, 2019, pp. 1–17.

[55] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[56] T. Popp and S. Mangard, "Masked dual-rail pre-charge logic: Dpa-resistance without routing constraints," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2005, pp. 172–186.

[57] K. Tiri and I. Verbauwhede, "A logic level design methodology for a secure dpa resistant asic or fpga implementation," in *Proceedings Design, Automation and Test in Europe Conference and Exhibition*, vol. 1. IEEE, 2004, pp. 246–251.

[58] D. Das, J. Danial, A. Golder, N. Modak, S. Maity, B. Chatterjee, D. Seo, M. Chang, A. Varna, H. Krishnamurthy *et al.*, "27.3 em and power sca-resilient aes-256 in 65nm cmos through¿ 350× current-domain signature attenuation," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 2020, pp. 424–426.

[59] M. Harvey, "Aes-128 ctr mode encryption/decryption system for vivado hls," 2019. [Online]. Available: http://www.markharvey.info/des/aesctr_128_hls/aesctr_128_hls.html

[60] M. Abadi and D. G. Andersen, "Learning to protect communications with adversarial neural cryptography," *arXiv preprint arXiv:1610.06918*, 2016.

[61] D. Modesitt, T. Henry, J. Coden, and R. Lathe, "Neural cryptography: From symmetric encryption to adversarial steganography," *Avilable at https://courses. csail. mit. edu/6.857/2018/project/Modesitt-Henry-Coden-Lathe-NeuralCryptography. pdf.*