

Luan Vo (30553600)
Professor Wayne Burleson
ECE 547 – Undergrad EE

[Table of Contents](#)

Title Page	1
Abstract & Keywords	1
Introduction	2
Motivation for Project	2
Report Organization	4
Related Works	5
Overview of SCA	5
DPA vs CPA	7
Methodology	8
Tools & Concepts	8
Implementation	10
Methodology	11
Partitioning Work and Schedule	12
Partitioning and Scheduling & Adaptation	12
Notes & Rationale	13
Experiments and Results	14
Summary & Procedure	14
Experiment	15
Results	23
Analysis and Reflection	24
Analysis	24
Personal Reflection & Project Reflection	27
Conclusion and Future Works	28
Conclusion	28
Future Works	29
Citation	30

I. Title Page

An Overview of Correlation Power Analysis and Noise, and Their Effects on AES-128

Abstract – Side-Channel Analysis (SCA) utilizes the additional information that leaks through the intermediate between the cryptographic algorithm and its hardware implementation. Ever since the inception of this method, over the years, SCA has steadily proven itself to be a strong contender and a force to be reckoned with amongst the wide ranges of attacks aimed at compromising cryptographic defenses – as shortly prefaced by [2]. Traditionally, cryptographic designs do not take into account the possible intermediate information leakage aside from specified inputs and outputs that are predetermined by cryptographic algorithms. These attacks are highly effective, easy-to-perform, cost-effective, and non-invasive when compared to software-based attacks that are much more complex, and resource-intensive. While Correlation Power Analysis (CPA) is a variant of Differential Power Analysis (DPA), however, they are very different when it comes to the method of execution – which will be further discussed below. CPA utilizes the linear relationship between the theoretical and actual power consumption of the device during the processes of cryptographic algorithms.

In this paper, I will explain the method of SCA – especially that of CPA – and its implementation against the pre-recorded AES trace file. At the same time, it provided a general overview of SCA and the effect that noises impose on the system. This paper will also discuss the in-depth and necessary complexities that are needed for the successful extraction of the secret key hidden in the AES system. The experimental results showed that CPA could successfully extract the secret key from the trace file, even under a simple noisy environment. In addition, I will also posit possible approaches and theoretical synthesis that will aid in the effort of combating CPA attacks in the future.

Keywords – Side-Channel Analysis (SCA), Correlation Power Analysis (CPA), Differential Power Analysis (DPA), Power analysis, Advanced Encryption Standard (AES), Hamming Distance Model (HD), Cryptanalysis, CMOS, Intermediate.

Preface – I apologize for the lateness in submission, the more detailed reason could be found in section IV (Methodology), and V (Partitioning Work and Schedule). The main reason for my lateness is due to my shortcoming in writing a long essay, as I want to make sure that everything keeps up with qualities from my previous works. And that while I have the idea, writing all of them out and organize a 25-page essay is a daunting and difficult task. I have been trying the best of my ability and I hope that you will be able to enjoy this work. Even if I do work 11 days straight with multiple all-nighters (the moment I am writing this is 4:05 am July 1st), so I hope that you could overlook my lateness for this one time. I do not like to turn in late work, as I have been submitting everything early or on time for this class, thank you for your understanding.

II. Introduction

Motivation – Amongst the numerous topics that we have gone through in this class, nothing has piqued my interest in the way that Side-Channel Analysis does. In the field of Electrical Engineering, nothing is quite as important as the concept of power. Power is an overarching unit that encompassed both fundamental concepts of voltage and current, thus, making power consumption of a circuit the most important design specification. With the introduction of SCA, I have been able to understand another use of power consumption that I would have never thought of with just the experiences gained inside a normal classroom. Besides its relation to my interest and major, I also chose this topic due to its practicality and diversity, as there exists an unimaginable amount of attack vectors (as well as solutions) that could be realized. My goal is to experiment – to the best of my ability and knowledge – that I have gained throughout this journey. In addition, I hope that this project will shed more light on my current interests in Electrical Engineering, as well as bolstering my understanding of Security Engineering.

Background – Ever since the inception of Side-Channel Analysis¹ – announced by Paul Kocher et al – in 1996, SCA has been extensively studied due to its emergence as one of the most efficient methods to crack a cryptographic algorithm without leaving behind visible traces, remain cost-effective, and dangerous due to its utilization of leakages from endless sources of interactions between layers of a system during cryptographic algorithm processes.

SCA exploits the weakness that is the link between multiple layers of the system, and it has shown to be even more of a threat 20 years after its introduction. As we further progressed down the road of technological development, each layer of a system is becoming more and more intricate and precise, making it increasingly difficult for each layer to effectively communicate and fully understand the others' functionalities [1]. Furthermore, under conventional mathematical cryptanalysis which does not consider the aspects of intermediate leakages – that is to say, everything is discussed under a black-box model, where no other information is available to attackers aside from plaintext and ciphertext [4]. These facts, combined with endless sources of human errors, create innumerable vulnerabilities and loopholes that could be exploited by malicious attackers. While the methods of SCA prevention remain to be extensively studied, however, the attackable surface of a cryptographic circuit is simply far too great – and its potential is only being limited by the attacker's imagination.

Amongst the methods of SCA as described by [3,7], there are some of the more well-known, well-documented, and well-studied approaches, such as DPA, SPA, and CPA. Differential Power Analysis (DPA) is a statistical analysis that analyzes the power traces to identify data-dependent correlations [1]. Simple Power Analysis (SPA) – a much more primitive and simple method – relies on the examination of features that are distinguishable in a single power trace, or by comparing many pairs of power traces, however, this method does not have the capabilities of noise-filtering [1]. Correlation Power Analysis (CPA) – the focus of this paper – evaluates the correlations between the captured power traces and the theoretical power leakage that is a value (or a function of) some intermediates during cryptographic operations [1].

1: SCA represents any attacks that bypass the cryptographic algorithm that was meant to defend the system. It is NOT simply DPA, CPA or SPA.

Problem Statement – There have been many solutions and proposals over the years on how to prevent SCA in both hardware and software, nevertheless, according to Jacqueline Lagasse et al, “some of the more popular countermeasures incurring 3-4x overhead in area and power.” [12]. While most proposals for combatting SCA are valid, they are not practical due to the increase in size and/or power consumption. This means that a proposal does not only have to be valid, it must also increase the preexisting functionalities and properties, because companies only want to implement systems that are optimized and robust – due to cost-effectiveness¹. Therein lies the root of the problem: the difference between academics and companies – one seeks to patch vulnerabilities, the other seeks profits.

This difference is what tips the balance in the favor of the adversaries, as companies – more often than not – do not always implement the fixes that were proposed, instead, they remain adamant that there exist no such issues, or ignore the proposed solutions completely. For as long as companies continue to exercise negligence in products’ security, we will always be fighting an uphill battle. In addition, the problem also stems from the majority of our theoretical proposals were made with some unnecessary overheads, thus, preventing them from being a practical solution for production purposes. However, this subject is not within the scope of this paper, and I cannot do it enough justices with just only a few sentences.

This paper will seek to explore the effect of noise² on SCA, and why it remains to be the optimal solution for combatting SCA. Besides, this paper will focus on ways that adversaries can invade a system via CPA, and the devastating effect that it could bring; in hope that it could further our understanding of the current situation, whilst providing some insights of my own.

Approach – To establish an understanding of CPA and how it operates, I will perform various CPA attacks on a given AES trace file and explain in-depth all the operations that lead to a successful key extraction via ChipWhisperer Toolkit. Afterward, noise and jitter will be introduced to the trace file. Since this is a pre-recorded file, not many alterations could be made, however, I will attempt to create a different noise function that will be able to combat CPA attack more efficiently than what has been provided. By learning the interactions between CPA and noise, it would be able to show why hiding and masking are currently amongst the most popular methods in preventing SCA attacks.

Expected Results – I expect that with the correct leakage model, CPA will be able to crack AES traces. However, if a source of sufficient noise is introduced, that should change. It is worth noting that, CPA key extraction is dependent on the leakage model – which is different when compared to DPA.

Connection with the rest of the course – At the fundamental level, SCA covers the fallacies and 6 C’s of security, it allows the reader to savor the richness combined with the wide variety of attacks and defense mechanisms that it has to offer. On a deeper level, SCA involves the aspects

2: no extra power consumption where it need not be, while retaining its size

3: Noise is an overarching term for any distortion presented/introduced to a system, be it mathematical, software or hardware.

of Confidentiality, Integrity, and Authenticity, as attackers could compromise the user data by breaking the AES system without leaving any visible traces. This nature of SCA makes it very similar to what we have learned during this course, such as Smart Grid's scalability, Automobile's connectivity, RFID's intermediate leakage, public-private key's encryption system. These are but a few connections that SCA has with other subjects, however, there has been one underlying subject that was hinted at but rarely explicitly told within the scope of this class – cost.

- Scalability: As technology becomes ingrained in our daily lives, security must also grow in proportion to this demand. Therefore, to accommodate this growth, security implementations must also evolve to meet this demand. This idea holds now than ever, as technologies are becoming readily available, so will the devices that have direct access to our personal information. Thus, aside from considering the possible attack vectors, we must also work to ensure that these devices will be appropriately secured even when further mass-produced.
- Intermediate Leakage: A reminder that there will always exist more than one way to compromise a system. Since the most effective attacks usually come from the most unexpected direction.
- Connectivity: One of the major reasons behind the diversity of SCA is due to many interactions between each layer. Therefore, the attacker can target any layers of the device, as displayed by SCA.
- Encryption: The lock that is AES or the cryptographic algorithm that was meant to defend the system could be secure, however, should the hardware implementation is not appropriate the whole system will be compromised.
- Cost: On the industrial level, the cost will be the determining factor of the resulting product. It represents the FIPS level, as each FIPS level is meant for each category of consumers and the intended system that it was meant to protect.
- Noise: could be modeled as RNG, which is related to the PUFs topic covered in the class.

Report Organization – This report will follow the standard format of most research papers:

- I. Title
- II. Introduction
- III. Related Works
- IV. Methodology and Tool
- V. Partitioning Work and Schedule
- VI. Experiments and Results
- VII. Analysis and Reflections (or Discussion)
- VIII. Conclusions
- IX. Citation and Related Works

III. Related Works

Overview of SCA – Electricity – one of the world most abundant source of energy – is the core foundation of our modern society, the lifeblood of civilization, and the forefront of technological development; it is ingrained in every electronic device, and soon will be integrated into our bodies with the introduction of RFIDs and IMDs. With the promise of convenience and a symbol of technological prowess, electronics have been extensively studied and their implementations into the world are up to our imagination. However, since 1996, a new form of security breach emerged and continued to pose a significant threat to technological growth – Side-Channel Analysis Attacks. To combat this rising threat, “cryptographic procedures of different embedded system are based on symmetric or asymmetric cryptographic algorithms such as DES, 3DES, AES or RSA” [8] were designed to protect the confidentiality, integrity, and authenticity of communication between devices.

SCA refers to any methods that do not directly aim at the device’s main defensive measures – cryptographic algorithms. One statement that well-captured the nature of SCA (and security engineering) was made by Noura Benhadjyoussef et al “even if the cryptographic algorithms and the protocols are protected, these devices are not trusted and confidential information may leak through side-channel analysis attacks” [10]. This statement brings the attention of engineers to the flaws that lie beyond cryptographic algorithms – the hardware. By manipulating the slight *unintentional information leakages* through intermediates that resulted from the *implementations* of cryptographic algorithms [1,4,7,10], adversaries could – potentially – gain insights into the device in ways that were thought impossible. However, it is worth noting that SCA does not simply attack a system indiscriminately, it only targets the *data-dependent power traces* – as these traces have direct correlations with the encryption processes. Beside from SCA’s effectiveness in cracking encryption standards, it also poses a threat to security due to its non-invasive capability [4]. Unlike software-based attacks – where one can trace the attackers via digital footprint (i.e: IP address, MAC address, etc) – hardware that are objected to side-channel attacks may not leave such obvious signs of being tampered, and worse, without a witness, it is nigh impossible to determine the culprit. In addition, “The attackers manipulate the devices with any available equipment to extract the information of the intermediate calculation results.” [1]. When compared to cracking software security, which is time-consuming, costly, and requires a large number of computational resources (i.e: public-private key algorithm). Instead of spending hundreds of trillions of years to solve a set of cryptographic algorithms, a well-executed side-channel attack would only require days or weeks. Also, one can easily manipulate a piece of hardware to act a certain way, which is worrying, since most hardware are made with consumers’ ease-of-use in mind, thus, making a large portion of devices being used today left dangerously undefended.

Over the year, numerous SCA attacks have been performed on devices and systems such as FPGA, SASEBO-GII, Microcontroller devices, and Cache. There are many forms of SCA, some of the most generally used are timing attacks, power analysis attacks, EM-attacks, fault attacks, acoustic attacks, visible light attacks, error message attacks, cache-based attacks, frequency-based attacks, and scan-based attacks [7]; where each attack is specifically tailored to aim at the system

most sensitive leakages, which meant to extract any possible additional information during the process. Refers to figure 1 for a general setup of SCA.

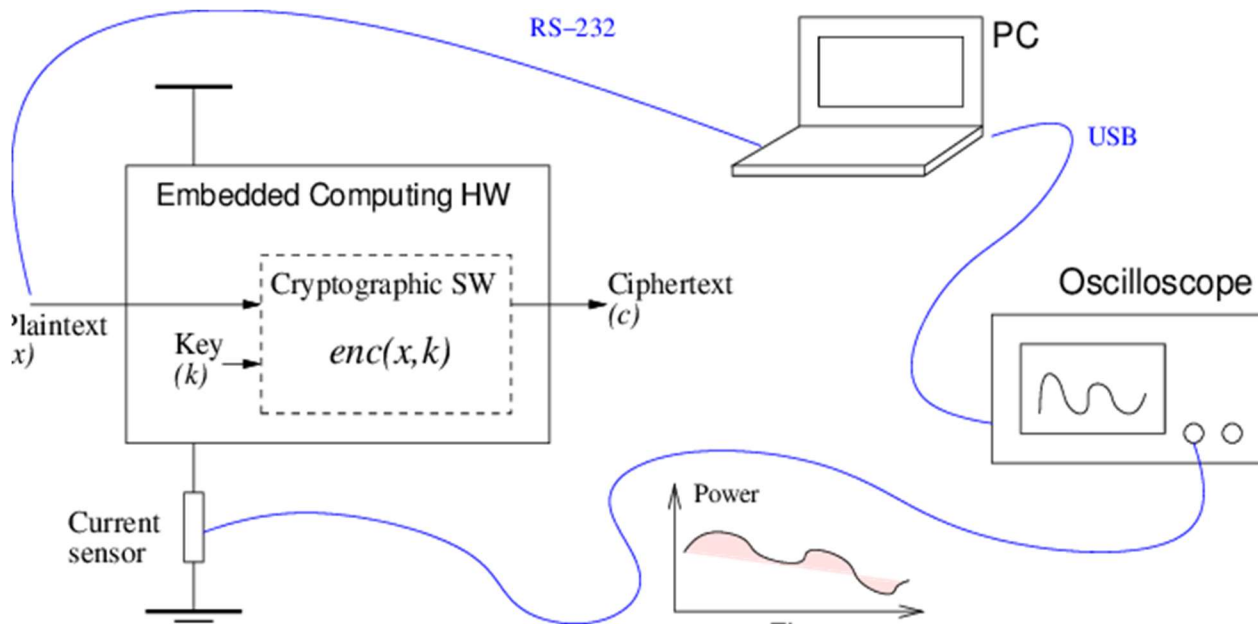


Figure 1. Simplified Setup for SCA

Firstly, SCA attacks can be used to extract the secret/encryption key, similar to what has been demonstrated by [8,9,10,11,14] where the main goal of the attacker is to extract any secrets hidden within the encryption module. According to Faisal Rahman Nuradha et al, the power consumption of any device could be modeled as follow: $P_{total} = P_{static} + P_{SC} + P_{dynamic}$ [9]. Where static power and P_{SC} are the general power consumptions of the circuit (due to current or voltage leakage) that tend to stay the same with time, thus can generally be ignored; whereas dynamic power is the output when transistors perform algorithmic operations. This means that the bulk of the data-dependent power consumption correlates to the output of $P_{dynamic}$, as the “data leakage through the power side-channel is related to the number of bits switching from one situation to the other” [10]. By aiming at the physical interactions of intermediates during algorithmic processes, attackers will be able to observe the *physical behaviors* of the defensive system during the cryptographic operations. With the help of these power traces, observers will be able to capture the system behaviors when a (guessed) encryption key is introduced to the system, thus, the correlation between the guessed key and secret key will become evident. However, SCA is not confined to only the use of power traces, with the rise in demand for online services, cache-based SCA has been taking an active role in extracting encryption key from email services (especially MUA⁴). For example, according to a recent study by Hodong Kim et al, “the attacker observes the victim’s execution of decryption algorithm provided by GnuPG, and acquires bit information of RSA private key of the victim.” [14]. Ultimately, these examples show that SCA may differ in forms (methods of attack) but similar in nature (make use of intermediate leakages). Therefore, it is well for us to remember – as security engineers – that the problem could materialize from many different directions, and the most devastating ones arise from the unexpected.

DPA vs CPA - As the name suggested, DPA relies on the differencing of the average power waveform between sets of data. This is to find the data-dependent correlations that exist within these waveforms. As with any statistical analysis, DPA is susceptible to noise if the trace samples are not sufficiently large enough when compared to the existing noise/jitter, as it will cause some disturbance and inconsistency in the DPA process. In addition, the part of power consumption that is being examined must only be the data-dependent part, that is to say, the power consumption of the system's cryptographic algorithm, not the entire circuit [5]. According to Alioto et al: "From the previous considerations, DPA (CPA) attacks are successful if the spike obtained with the correct guess can be distinguished from the others." [5]. This spike represents the correlation between the collected samples of power estimation and the measured power consumption. The magnitude of the spike is the waveform presentation of this correlation, the more correlation there is, the higher the spike's magnitude and vice versa. Due to this nature, if the noise level exceeds that of the spike's, no correlation could be made, as these spikes are drowned within the noisy environment.

While DPA and CPA are very similar, the method of execution is what sets them apart: differencing (of average) and correlation (of leakage model). According to Paul Kocher et al, "CPA involves evaluating the degree of correlation between variations within the set of measurements and a model of device leakage that depends on the value of (or a function of) one or more intermediates in the cryptographic calculation." [1]. Despite the similarity, unlike DPA, CPA does not necessarily compare the correlation of power consumption, instead, CPA seeks to compare the correlation between the leakage model of the theoretical and measured system. From figure 2, the leakage model is S-Box. However, it could be many other things (such as Last Round State leakage), but it usually falls into the calculation of Hamming weight/distance [5,10] mathematical model. Because of this dependency on the input leakage model, "CPA works best in the *white-box* analysis." [1,5]. During the mathematical process of combining K (secret key) and X (plaintext), there are some intermediate leakages pertaining to that specific system, and the combination of these elements then produces I (ciphertext). Therefore, by utilizing the leakage model, CPA can predict the key by observing the correlation between each byte⁵ of the guessed key and the actual one.

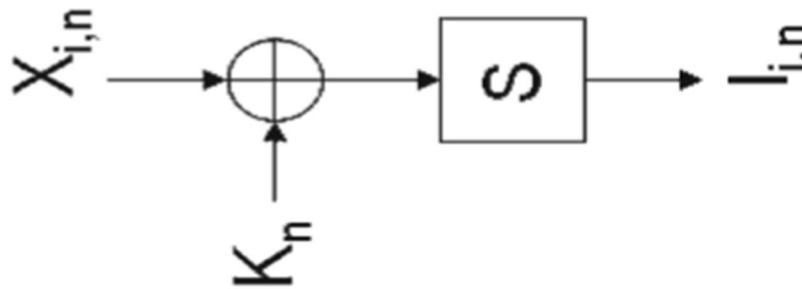


Figure 2. General CPA attack model

5: Though wasn't mentioned, CPA has the capability to analyze byte-by-byte, whereas DPA cannot.

IV. Methodology and Tools

Tools – For this project, I will be using the toolkit distributed by ChipWhisperer. There was an attempt to use the trace file given dpacontest.org, the trace file can be found via the download from contest's official web: (<http://www.dpacontest.org/v2/download.php>, a total of 3.5GB to 5.4GB, depending on the database that was used). However, time is limited, so I will be using the trace file given in DPA_assignment. Nonetheless, I will be performing exactly what I have planned with the contest's trace file. Below are the tools, and programs needed for the production of this project:

- Chipwhisperer 4.0.2
- WinPython-32bit-2.7.10.3
- Trace file: aes_unprotected

Concepts: Noise and Linearity – To understand the methods behind SCA attacks, it is imperative to learn how (unintentional) information leakages could be produced by hardware interactions during cryptographic operations. Three major properties tie with the foundation of CPA: CMOS transistors, Hamming Weight/Distance, and noise.

CMOS (or Complementary Metal–Oxide–Semiconductor), a type of transistor that utilizes the complementary pair of n-type and p-type MOSFETs, which are placed symmetrically inside a circuit (refer to figure 3). By placing these FETs symmetrically, it allows the circuit to have a switch-like⁶ property, as these FETs behave in contrast with each other. Thus, if an input to CMOS is high, then the output will be low, and vice versa. Therefore, CMOS is constantly transitioning between two outputs – high and low – create a switch-like behavior.

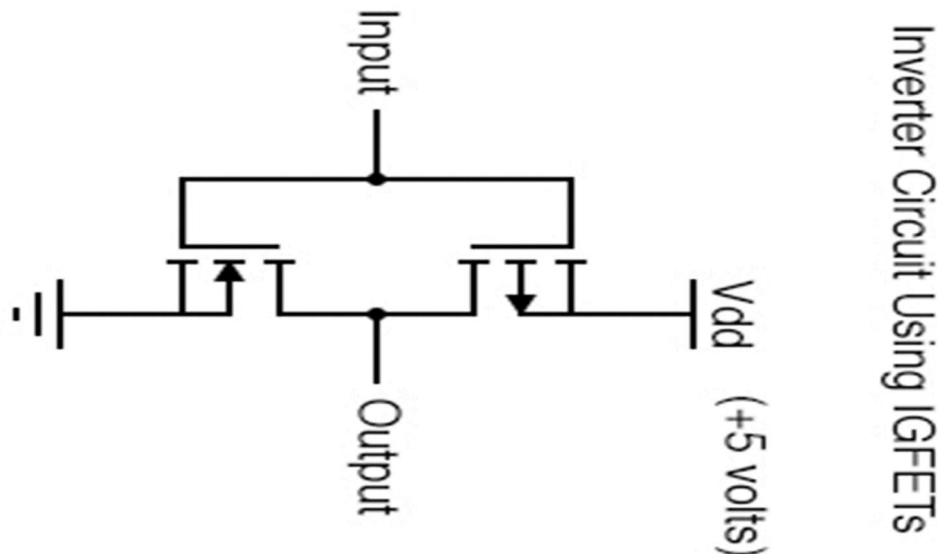


Figure 3. CMOS structure

6: they are not switching instantaneously, depending on how the frequency/voltage/model were chosen, the off-time for these FETs will change.

According to Eric Brier et al, “Classically, most power analyses found in literature are based upon the Hamming weight model, that is the number of bits set in a data word” [6]. The Hamming weight model is a method that is widely used within a communication system where it determines how far apart two (binary) data words are. In other words, we are trying to determine how many bits are getting turned on and off during the execution of cryptographic algorithms, since it is generally assumed that the power consumption correlates with the number of bits that need to be flipped from one state to the next. In addition, a microprocessor could be modeled as a state machine, where each state transition is triggered by events (such as clock signal) [6]. This is due to the behavior of a CMOS that can sometimes be very similar to that of a switch – by constantly alternating its output between high and low. Individually, each transistor may act in a non-linear fashion, but on a larger scale, the device’s hardware obeys this linear model quite well. In addition to the linearity, it is worth noting that we only want the energy consumption of the data-dependent part and not the entire circuit’s power consumption. If the power traces contain non-data-dependent parts, it could lead to the incorrect answer as there exist noises within the traces.

By introducing noise to the system, it will blur the linearity – that resulted from the hardware interaction during cryptographic operations. This is the reason why, most of the solutions, geared for combating SCA, fall in this category, as it is easy and efficient. As transistors are getting smaller and faster – per Moore’s law – it is now possible to fit more transistors on a piece of the device when compared to before. This is advantageous in our quest to prevent the growth of SCA for several reasons [13]:

- Space: Aside from being able to fit more transistors into a device, we could – in turn – reduce the size of the hardware to something smaller. This means that the device is portable, easy-to-use, and much more resistant to SCA.
- Low-power device: Inherently, CMOS are low-power devices. Thus, if a lot of CMOS are stacked together, it will become much more difficult to distinguish the data-dependent leakages from reading power traces. This is due to the existence of various parts of CMOS performing many different tasks in close-quarter (refer to figure 4). Imagine a phone call center, where the work area is small – there will be much more noise when someone calls the center. Whereas for a larger call-center, the caller will be able to fully distinguish the recipient’s voice and her surroundings. This tactic works for almost every SCA, be it EM-emission or visible light, etc (by using the logic and example above). In addition, by making CMOS smaller, it effectively introduces noise to the system, while incurs fewer overheads in area and power – which is a concern for Jacqueline et al in [14] and many others in their quest for preventing SCA. However, one cannot rely on Moore’s law forever, as it is predicted to end as soon as 2025.
- Noise: A general term for any distortion that blurs the linearity that inherently exists between a conventional CMOS design and cryptographic operations. Though introducing noise has its advantages and disadvantages, it remains to be the most effective method up to date.

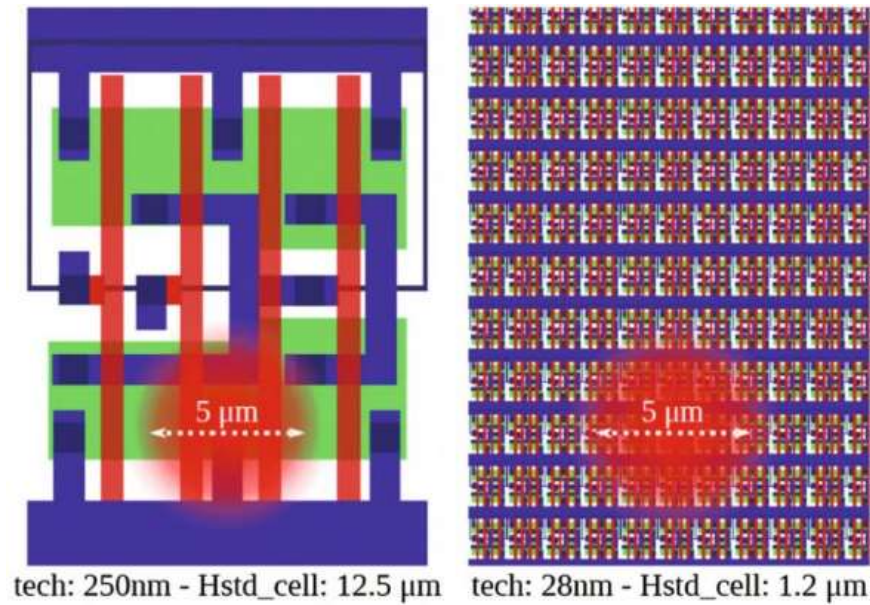


Figure 4. Effect of Moore's Law (left: before, right: after) [13]

When combatting SCA, there have been many proposals, some of which are “de-correlate the output traces”, “replace critical assembler instructions”, and “algorithmic changes to the cryptographic primitives so that attacks are provably inefficient on the obtained implementation” [7]. However, software-based countermeasures remain to be widely used, due to its flexibility, and in many contexts, cheapest to be put into practical uses. This is because hardware-based countermeasures often incur undesirable overheads such as an increase in area or power consumption. There are two primary ways (but not limited to) in which SCA can be combatted: masking and hiding. Both of these seek to introduce randomness and noise into the system to make the process of retrieving and analyzing much more difficult. This can be done via the introduction of a dummy variable, randomization process, etc. In other words, any process that can hide the intermediate step of the cryptographic algorithm (as this remains to be the biggest data leakage that SCA attacks make use of) [7]. However, to obtain this, one must be able to create a reliable source of RNG. For example, one of the countermeasures was proposed by Jacqueline et al, by using hardware generated RNG such as clock randomization and supply voltage noise, they have shown that “combining these countermeasures can nearly double protection from CPA” [14].

Implementation – With the above knowledge in mind, it is clear that the effect of noise runs much deeper than just simple systematic distortion, to be able to create a truly random and varied noise, we must be able to create it dynamically and without relying on a specific pattern. By making these points (noise, CMOS, and linearity) the focus of my project, I will modify the steps and my methodologies so that it will clearly show these effects during my simulation.

Methodology – Based on the above knowledge, my methodology will be as follow:

- Setup:
 - This step is done by using preexisting traces, however, theoretically, it can be done by following the figure 1 setup.
 - Installation follows this procedure:
 - First, installs WinPython version 2.7.10.3 for 32bit
 - Secondly, downloads Chipwhisperer version 4.0.2, then extracts to the desired location.
 - Navigates from winpython to chipwhisperer 4.0.2 directory via “cd” command.
 - Some more commands could be found in:
<https://www.tecmint.com/useful-linux-commands-for-newbies/>
 - Runs the command python CWAnalyzer.pyw
 - Loads given trace into CWAnalyzer program
- Attacks:
 - Performs a default CPA attack against the trace file. This is to serve as the controlled group or the benchmark of the project.
 - Performs CPA with different leakage model, such as LRS, inv_Sbox, Sbox successive, and Sbox difference, etc. This is to show the importance of choosing the correct leakage model, and why it is important in CPA.
 - Performs CPA with added noise to explore the effect that it has on SCA. This is one of the most important aspects, as it will show how much noise is needed to fool the CPA program. As discussed above in section III and IV.
 - This also holds another importance since the SNR (signal to noise ratio) must be sufficient when compared to the amount of traces. Which will be later discussed in the result section.
 - Explore the difference when noise is introduced individually and more than one source of noise.
 - Explains why noise is advantageous and disadvantageous at the same time.
 - In hope that this will explain the meaning of “linearity”
- Graphs:
 - Studies the graph to show where the program determines which is the correct byte. Must be able to fully extract the information that is presented within the graph.
 - Be able to fully explains the correlation of leakage model through graphical outputs.

V. Partitioning Work and Schedule

Partitioning work – The work for this project was done by me – Luan Vo. With professor Wayne Burleson’s extra suggestions.

Schedule – Below is the schedule that serves as the groundwork and backbone for this project’s progress.

Groundwork and Ideas:

- June 11 – 18: the project foundation started along with the submission of the presentation and the DPA_assignment.
 - o This is also the reason why I have a lot of extra information contained within that assignment, as I planned on reusing it should I don’t have time.
 - o The presentation served as the concept and methodology of this project.

Gathering relevant research:

- June 15 – June 20: Gathering extra research to help myself better understood the concept of SCA, especially that of CPA.

Report writing plan:

- June 21st: Abstract and formulate a table of contents
- June 22nd – 23rd: Introduction and citation
- June 23rd – 24th: Related Works
- June 25th – 26th: Experiment and data gathering
- June 27th: Methodology and tools & scheduling finalization
- June 28th – 29th: Experiment and Results
- June 29th – 30th: Analysis and Reflection
- June 30th – July 1st: Conclusion and Future Works

Adaptation – The progress (page-per-day), which I could write, was much slower than what I had hoped. Originally, I planned on writing 3-4 pages per day, but with my limited capability, I could only write as many as 2 (while maintaining coherency) – since English is not my first language.

Similar to my other papers in this course (SSD, voting, etc), it would take me 4-5 days to write a coherent and “well-written” 3-4 pages essay. I have tried to uphold the principle that my works will continue with good quality, thus, making the process of writing a 25-page essay a complete brain-racking for me. During the process, it is my top priority that the reader could enjoy the work I put forth – to show that I chose the subject because of passion, not just because I need to finish this assignment. I want to be able to fully understand the project and my essay should express the necessary knowledge to my readers as they read through.

Thus, the process of writing is slower due to 2 main reasons (because I want to maintain these):

- Coherent: Language is a strong tool, well-written essays will capture audiences' attention
- Knowledge: To further my own, as well as my reader's understanding of the subject.

Rationale – Realizing my own shortcoming as a slow writer, I have started this project as early as June 19th – the day after the submission of the presentation (June 18th). With each day progress is 1.5-2 pages, I realized that I cannot keep up this pace with the current deadline. As much as I wanted to keep up the quality of my work, however, the completion of the project should be prioritized. Instead of writing carefully, I have, instead, taken a stance to follow the style “flow-of-thought”. However, due to this method and my inability to write coherently in quick succession, there will be many grammar errors and clumsily worded sentences.

I have divided the work so that I will write the bulk of the introduction first, but even with spending days of working continuously, I cannot make it in time to experiment using the traces from DPAcontest, as that would likely take me at least another 3 days for experimenting. For the past week, I have been working from the moment of my wake till my sleep and repeat; still, I could not accomplish this. However rush the project might be, I still hope that the readers will enjoy the work I put forth, as I mentioned earlier, I chose this subject because it has been relatively glossed over during the course. Nonetheless, it is very intriguing to learn about how faults in implementations could result in information leakage (as this also coincides with Fallacy of secure lock: a lock might be secure but if the implementation of the lock isn't, then the system will soon be compromised).

In anticipation of my shortcomings, I have put my effort into ideas and sentencing, so that I could re-use my presentation as well as my DPA assignment for this project. As I believe that if I do not follow this plan, and instead of working on a completely new idea (such as PUFs or RFIDs), I would have to start from the ground up, and that would take me at least a month to finish the project. Thus, I have tackled this project with a considerable amount of consideration on what I should have done since the beginning of June – at the start of the DPA assignment. Though at the time I still not have a good understanding of the subject, so I was still a little hesitant about my goals. However, in anticipation of this, I have gone an extra step during the DPA assignment, so that I will have something to go off of.

Note – At the moment of 12:46 AM, June 28th, 2020, I am in the midst of writing the experiment and result section.

As an informal note, I sincerely apologize for not being able to finish this on time (which I have tried to do this for every assignment of this class – by not turning in late assignments). Since this is a long paper, even though you did say by July 6th – at the latest – I apologize for the drop in quality compared to my previous papers. I believe that I should be able to finish everything by July 1st, at the earliest.

VI. Experiments and Results

Summary – The objective of this assignment is to explore side-channel attacks using a chipwhisperer analyzer with various methods, such as CPA, DPA, and different leak-models with noise. This is to further our understanding of how malicious attackers can aim directly at hardware operations to unveil the secret key hidden within the system. I performed various attacks on a given trace file (aes_unprotected.zip), this trace file was made by capturing the activities of the AES system while it was running. It effectively acts as – what I would call – a static AES power consumption and activities. This means that the AES secret key will remain the same, and the various hardware operations will be the same – that is to say, unchanging.

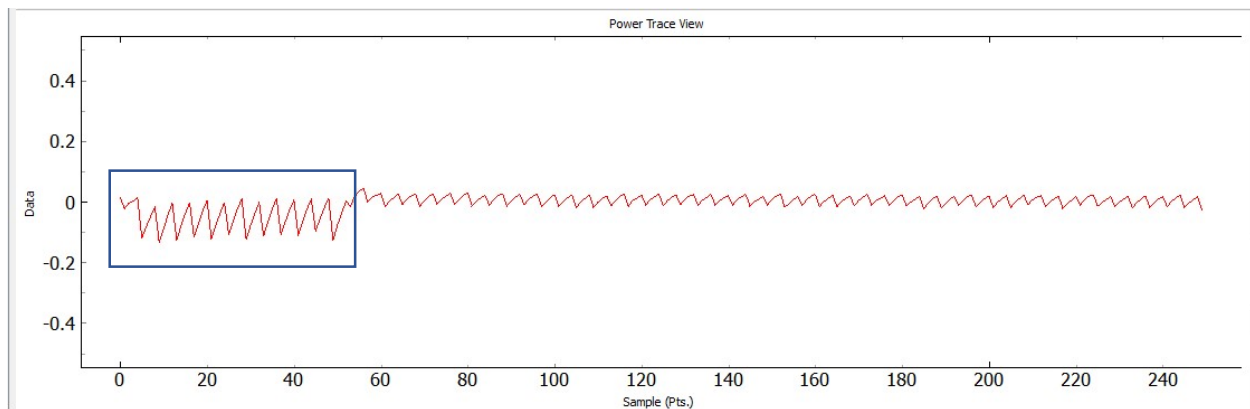
The interpreter being used for the project was: WinPython version 2.7.10.3 (32 bit). Besides, I also installed some dependencies, that will aid the interpreter is running CW Analyzer's interface: pyqtgraph, configobj, pyusb, PySide.

Original key (at the time of AES trace capture):

AckPattern: Basic (Key=Fixed:2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C,
Plaintext=Random); Aux:
set_prefixbefore_capturebefore_tracebefore_armafter_armafter_traceafter_capture

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Key	2B	7E	15	16	28	AE	D2	A6	AB	F7	15	88	09	CF	4F	3C

Lab & Procedures – The purpose of the experiment is to test the success of key extraction under different noise environments and leakage models. There exist 5000 traces, with the graphical output of 250 points, thus, $5000/250 = 20$ traces for each point. It is easy to visibly recognize the encryption rounds when AES-128 is unprotected:



This trace file is specifically for ChipWhisperer CW305 (Artix-7), and as shown in the boxed area above, the board encryption activity is shown within the vicinity of the box. This means that the encryption activities take place approximately between 0 – 1200 traces out of 5000 traces. With this in mind, we could look for the highest correlation within this spectrum.

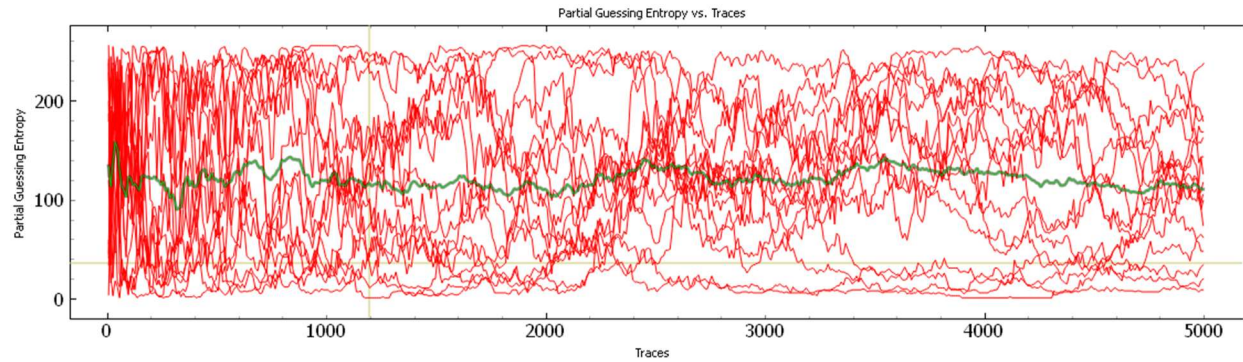
Experiments – Case 1: Default CPA attack with S-Box leakage model

It is worth noting that, the correct key number is the number that belongs to the PGE row. For example, the first correct key byte is at index 221, which equals to 2B. (boxed in red below). However, it does not indicate that the program has successfully extracted the key. For the analysis steps, one must examine the graphical output of correlation to understand if it is possible for the program extracts the correct key or not. (methods such as brute-forcing)

PGE means, as stated by [16]: The ‘guessing entropy’ is defined as the “average number of successive guesses required with an optimum strategy to determine the true value of a random variable X”. The ‘optimum strategy’ here is to rank the possible values of the subkey from most to least likely based on the value of the correlation attack (higher correlation output is more likely). This means that “A PGE of 0 indicates the subkey is perfectly known, a PGE of 10 indicates that 10 guesses were [incorrectly] ranked higher than the correct guess.” [16]

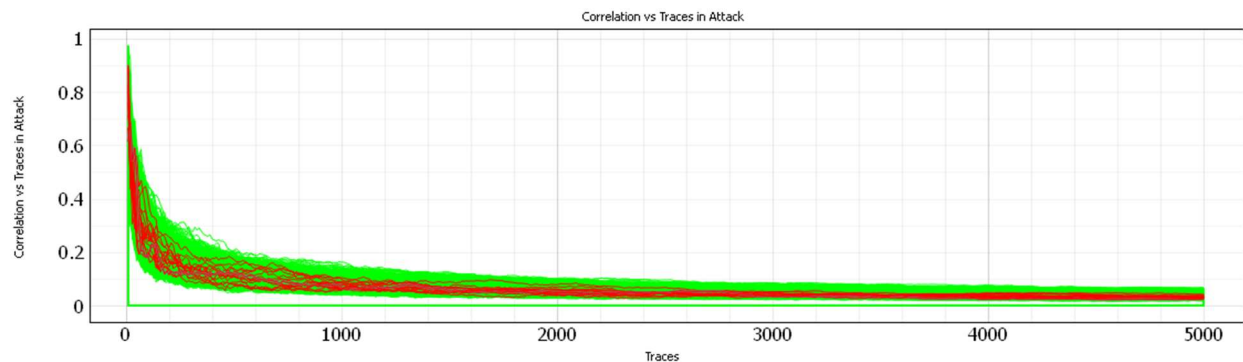
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	221	175	117	183	197	50	231	68	164	163	100	244	129	188	229	218
0	57 0.0665	1A 0.0573	FF 0.0597	AB 0.0549	92 0.0661	9E 0.0607	86 0.0594	36 0.0550	E7 0.0657	61 0.0628	E2 0.0578	FE 0.0618	1D 0.0553	FA 0.0632	8E 0.0613	D7 0.0595
1	B3 0.0584	21 0.0561	77 0.0583	55 0.0545	01 0.0557	B7 0.0556	5E 0.0583	43 0.0521	70 0.0563	B4 0.0516	9E 0.0493	04 0.0531	E2 0.0538	BB 0.0630	F0 0.0555	1B 0.0537
2	DB 0.0573	B9 0.0542	84 0.0562	AE 0.0512	FD 0.0549	0C 0.0551	4D 0.0547	2A 0.0502	5D 0.0563	16 0.0490	38 0.0490	F4 0.0519	F5 0.0514	F0 0.0594	EE 0.0535	E8 0.0519
3	F1 0.0572	8E 0.0539	34 0.0525	7C 0.0497	17 0.0542	AF 0.0539	1D 0.0535	39 0.0498	1C 0.0541	51 0.0489	84 0.0490	B4 0.0517	32 0.0512	CD 0.0573	63 0.0527	22 0.0504
4	82 0.0526	77 0.0531	02 0.0523	71 0.0490	67 0.0539	1B 0.0535	08 0.0534	B7 0.0497	10 0.0536	65 0.0480	2A 0.0487	4A 0.0515	C0 0.0502	85 0.0550	E1 0.0523	7E 0.0496
5	0F 0.0522	E1 0.0522	79 0.0518	8A 0.0488	0C 0.0530	DB 0.0527	8A 0.0530	EE 0.0493	75 0.0531	33 0.0480	EC 0.0487	01 0.0510	F9 0.0487	73 0.0543	CD 0.0516	A2 0.0489

- Approach:
 - This attack is a default attack that was included within the CW Analyzer’s sample attacks. The program analyzes the correlation between the input and the actual leak model. By observing the power consumption waveform, the program gets a slight understanding of AES activities, for example: if the power consumption is high (or non-zero), it means that AES is performing some processes, which means that there is a link between AES and its power consumption [1]. Correlation analysis works best in white-box analysis, that is: the leakage model of the target is known. [1]
 - Due to the above definition of PGE, in order to find the average guesses needed for all 16 subkeys, we need to average the PGE overall 16 subkey guesses. In this case, the average guesses needed to obtain the full encryption key is 167.3125.
 - Where $PGE_total: 221 + 175 + 117 + 183 + 197 + 50 + 231 + 68 + 164 + 163 + 100 + 244 + 129 + 188 + 229 + 218 = 2677$
 - $PGE_average = 2677/16 = 167.3125$. Refer to the picture below, where the green line is PGE_average.



- Interpretation of PGE vs Traces:

- The reason behind the fluctuation of PGE is due to the fact that, as the program goes through the traces, it guesses of PGE at that trace fluctuates, thus, creating the variation of PGE. However, at the PGE that produce the correct subkey, the average is 167.3125



- Interpretation of Correlation vs Trace in Attack:

- This graph shows the upper limit and lower limit of correlation between each subkey and trace in the attack. The limit is represented by the green area, where the max is the upper outline and min is the lowest outline. It is possible to see that the correlation between the guessed key and the actual key (from traces) lies within the green area, this is a valid attack and key extraction is successful. As mentioned above, this only means that this approach (S-Box leakage) has a chance at getting the key at an average of 167.3 guesses.
- Besides, we can see that the correlation is most active and unique between 0 to around 1000 traces. This coincides with our guestimate of 0 to 1200 from above. This means that the region between 0 and 1000 is where encryption activity happens.

- Obtained key:

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Key	2B	7E	15	16	28	AE	D2	A6	AB	F7	15	88	09	CF	4F	3C

- Result:

- The obtained key and original key matched.

- The attack was a success, this method was able to successfully extract the secret key from AES with S-Box leakage model.

- Reason:

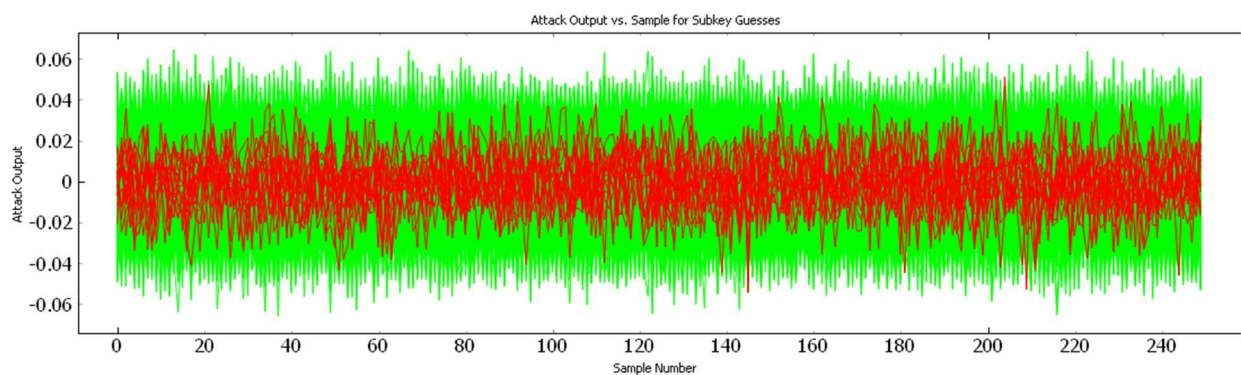
This works because there exists a correlation between the S-Box leakage model and the actual leakage of AES traces

Case 2: CPA attack with S-Box leakage model under noisy environment

In this case, we will examine the effect that noise imposes on the CPA process. Note that noise distribution is modeled after a Normal distribution (i.e $N(\mu, \sigma)$)

- With noise = 0.03

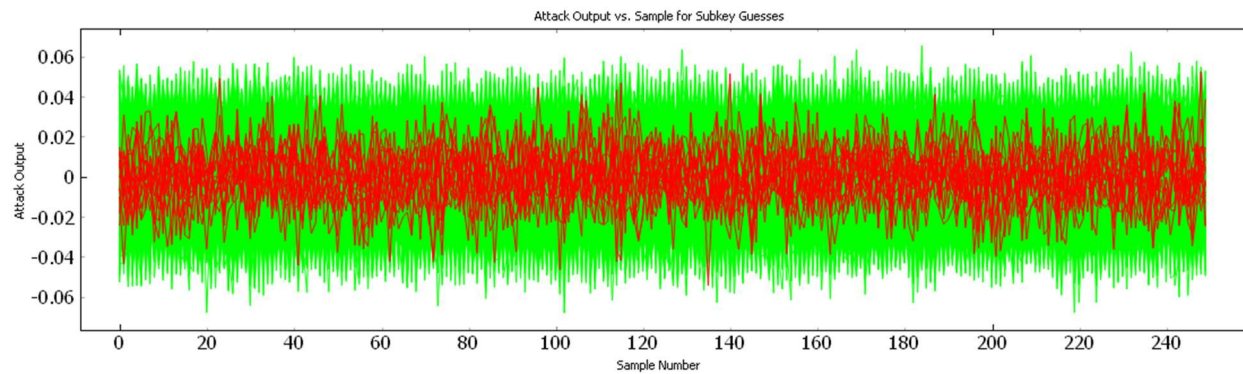
Results Table																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	22	197	9	242	208	139	82	86	53	153	136	107	238	126	189	201
0	A2	73	F4	D1	C4	42	5B	A9	17	D3	C5	B2	35	B7	6A	7C
	0.0622	0.0638	0.0607	0.0627	0.0596	0.0641	0.0559	0.0655	0.0616	0.0626	0.0661	0.0630	0.0639	0.0644	0.0647	0.0650
1	1C	9A	52	9E	49	84	B7	E7	84	69	35	66	07	64	7A	88
	0.0609	0.0614	0.0589	0.0604	0.0576	0.0573	0.0555	0.0632	0.0572	0.0625	0.0643	0.0620	0.0608	0.0560	0.0606	0.0637
2	0E	A8	CE	5B	A2	34	5F	32	82	2D	69	6C	EC	61	18	30
	0.0605	0.0550	0.0589	0.0573	0.0575	0.0572	0.0550	0.0620	0.0568	0.0596	0.0637	0.0597	0.0606	0.0545	0.0595	0.0621
3	0C	FD	7B	AE	97	DC	3D	5E	E5	0C	D5	A6	DB	BE	DA	60
	0.0583	0.0549	0.0581	0.0566	0.0569	0.0564	0.0550	0.0608	0.0563	0.0595	0.0575	0.0592	0.0590	0.0544	0.0584	0.0614
4	65	49	C4	EC	C5	10	40	B5	4E	36	9C	23	B8	BF	A4	37
	0.0567	0.0548	0.0569	0.0564	0.0559	0.0554	0.0536	0.0604	0.0560	0.0584	0.0563	0.0589	0.0585	0.0535	0.0568	0.0604
5	1E	0E	81	D7	F9	E7	0D	A4	E2	A5	B3	C9	D6	E0	6D	D3
	0.0565	0.0542	0.0564	0.0563	0.0552	0.0552	0.0532	0.0592	0.0557	0.0579	0.0557	0.0584	0.0582	0.0532	0.0548	0.0602



- With noise = 0.06

- With noise = 0.06:

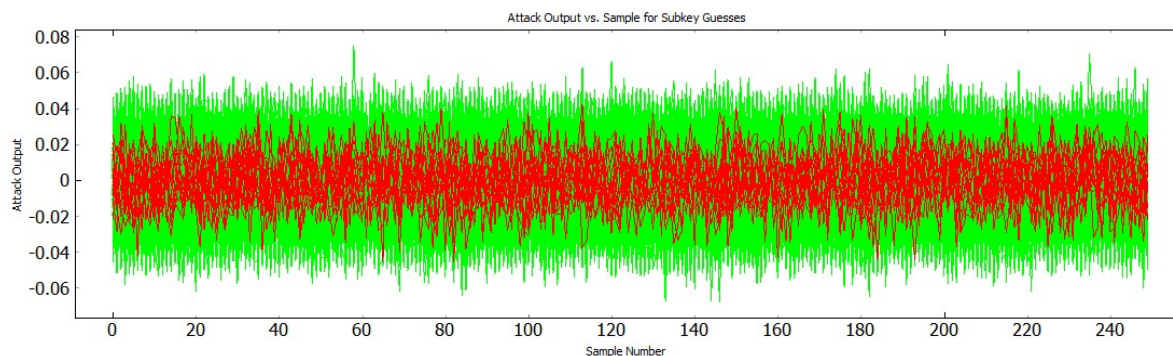
Results Table																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	217	164	91	158	125	45	118	63	150	19	10	251	16	149	113	241
0	69 0.0645	CE 0.0592	22 0.0634	75 0.0630	C8 0.0628	EA 0.0655	4A 0.0616	F6 0.0678	6B 0.0601	78 0.0593	26 0.0596	58 0.0679	52 0.0623	65 0.0594	35 0.0579	F5 0.0680
1	DB 0.0629	48 0.0568	6F 0.0605	FB 0.0609	84 0.0582	1E 0.0644	09 0.0579	8D 0.0600	D5 0.0567	5C 0.0576	9B 0.0584	3F 0.0571	24 0.0609	FB 0.0568	18 0.0572	C6 0.0593
2	87 0.0628	4B 0.0558	4B 0.0588	99 0.0602	4D 0.0567	5C 0.0634	DB 0.0578	36 0.0600	9F 0.0562	01 0.0564	D4 0.0577	59 0.0561	50 0.0584	D4 0.0568	52 0.0569	02 0.0580
3	C4 0.0591	59 0.0556	8B 0.0580	C1 0.0587	11 0.0565	AC 0.0618	CE 0.0559	CC 0.0575	3B 0.0547	AC 0.0560	18 0.0571	3C 0.0556	D4 0.0575	E6 0.0556	92 0.0561	E3 0.0567
4	6C 0.0586	46 0.0549	19 0.0579	95 0.0562	2F 0.0560	CB 0.0575	03 0.0558	45 0.0564	08 0.0544	A8 0.0558	E7 0.0561	FA 0.0555	4A 0.0566	CD 0.0556	AA 0.0545	21 0.0562
5	E5 0.0578	8A 0.0542	12 0.0578	9C 0.0555	AF 0.0559	DE 0.0573	01 0.0556	CA 0.0562	1A 0.0541	37 0.0554	73 0.0561	C0 0.0554	C2 0.0562	B6 0.0548	2B 0.0544	7B 0.0554



PGE_average when noise = 0.06: 120.625

- With noise = 2

Results Table																
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	203	213	196	75	253	84	100	254	236	241	143	126	130	99	210	212
0	64 0.0703	2E 0.0674	9C 0.0575	3F 0.0620	3E 0.0677	EC 0.0601	D3 0.0578	78 0.0584	A1 0.0645	D7 0.0561	2C 0.0598	14 0.0748	76 0.0655	F2 0.0622	FC 0.0555	D4 0.0567
1	49 0.0622	69 0.0638	7A 0.0558	E5 0.0594	EF 0.0674	A7 0.0578	55 0.0553	57 0.0577	D0 0.0645	53 0.0558	05 0.0581	70 0.0627	A1 0.0588	2C 0.0605	2C 0.0553	88 0.0564
2	A9 0.0574	B1 0.0605	62 0.0555	7E 0.0575	C9 0.0619	AA 0.0576	F8 0.0550	98 0.0570	D3 0.0612	6B 0.0549	AD 0.0578	4B 0.0615	E7 0.0581	4B 0.0589	BE 0.0546	B6 0.0550
3	F2 0.0570	DD 0.0590	5D 0.0548	1E 0.0567	9C 0.0618	3B 0.0568	20 0.0547	A1 0.0556	9F 0.0577	91 0.0548	57 0.0570	D1 0.0591	57 0.0572	A4 0.0585	24 0.0541	C3 0.0547
4	D6 0.0557	00 0.0582	5E 0.0544	72 0.0538	AA 0.0609	D1 0.0558	93 0.0542	85 0.0555	2F 0.0575	FB 0.0548	68 0.0558	DC 0.0584	12 0.0570	F7 0.0578	55 0.0538	2A 0.0540
5	91 0.0539	5D 0.0573	80 0.0541	7D 0.0536	9F 0.0594	EE 0.0543	05 0.0540	44 0.0548	08 0.0572	9E 0.0544	0A 0.0557	FA 0.0583	DD 0.0566	F0 0.0574	0B 0.0524	26 0.0539

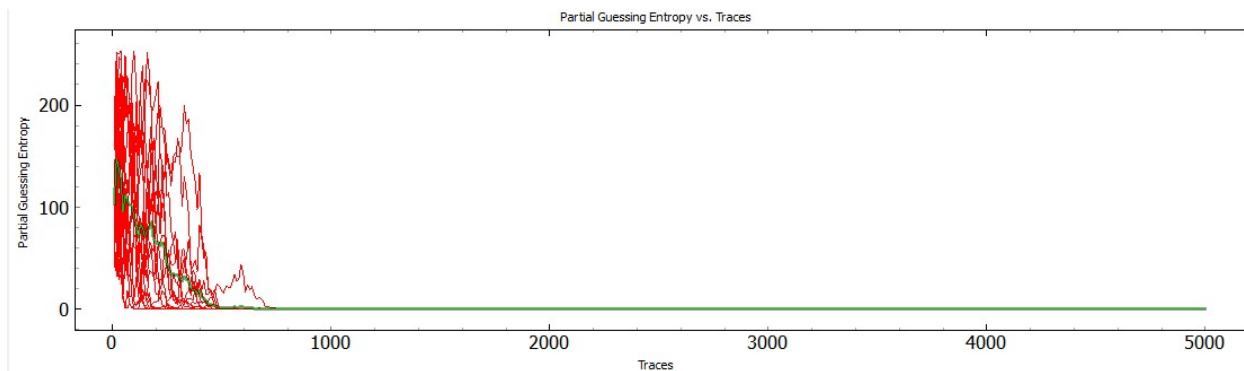


PGE_average for noise = 2: 173.4

Interpretation:

- Through trials, it seems that if the noise addition is larger than 1, in general, the key PGE average will also go up, though I have not been able to make the program unable to obtain the secret key through adding noise by using CW program.
- However, this is expected, as to be able to fool the CPA process, not just adding any noise will suffice, that is why it is important to learn the nature of noise to avoid unnecessary overheads. Therefore, to make the program unsuccessful in key extraction, noise must be added to the trace file, not during the processing.
- These are two important distinctions that must be made when discussing noise:
 - o The first type is noise that is added to the source file. Meaning that this noise is to prevent accurate reading and analysis of traces file. I noticed this because no matter how much noise I added via CW add noise function, the trace output does not change. This means that the trace file is not the target of modification when using this function.
 - o The second type is noise that is added preprocessing, or before/during the time of the attack, this noise might interfere with the key extraction. As we have seen with when noise addition is equal to 2, the PGE average goes up, it means that the average guess to get to the correct key slightly goes up. However, it is not a guarantee, as when noise is equal to 0.03 and 0.06, the PGE needed to get the key went down. While this is unexpected, however, I don't think that it is completely untrue. While noise is generally made to hamper the system, in this case, when observing the graph attack output vs samples for 0.03 and 0.06. I noticed that with the added noise, the red (the correct guesses) lines extend outward and getting closer to the upper/lower limit of the green area. This means that by adding noise using this method, I have inadvertently made the guess more accurate, as I am skewing the red guesses to come closer to the actual. (note that the green area is determined by the system).
- Therefore, a countermeasure based on noise and randomization that seeks to prevent SCA must be added to the trace file, not solely during the preprocessing, as these 2 noises are different by nature. We must understand this distinction to avoid overheads and unoptimized solutions.

Case 3: CPA with Last Round State leakage model (or LRS leakage model):



Results Table

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PGE	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	D0 0.2130	14 0.1947	F9 0.1803	A8 0.1837	C9 0.2160	EE 0.2286	25 0.2236	89 0.2098	E1 0.1718	3F 0.2212	0C 0.1818	C8 0.2066	B6 0.1789	63 0.2102	0C 0.1993	A6 0.1776
1	7C 0.0689	A8 0.0587	3E 0.0729	B9 0.0770	4C 0.0704	E7 0.0598	F7 0.0574	63 0.0593	B1 0.0730	13 0.0600	E0 0.0690	D5 0.0544	33 0.0648	BC 0.0612	DF 0.0635	89 0.0607
2	75 0.0625	A2 0.0536	A0 0.0580	09 0.0572	0D 0.0573	23 0.0579	7C 0.0524	03 0.0562	02 0.0715	F5 0.0594	FD 0.0561	CF 0.0541	AE 0.0604	7F 0.0567	E2 0.0569	E2 0.0596
3	57 0.0588	5A 0.0528	DD 0.0556	91 0.0538	07 0.0555	8F 0.0544	05 0.0515	DB 0.0556	33 0.0599	24 0.0547	9A 0.0560	0E 0.0535	43 0.0541	22 0.0540	E0 0.0567	D0 0.0582
4	85 0.0578	E0 0.0521	5E 0.0522	46 0.0517	FE 0.0532	8E 0.0513	8E 0.0508	5B 0.0503	18 0.0580	C3 0.0543	2D 0.0558	32 0.0528	CF 0.0531	F6 0.0525	8F 0.0543	9E 0.0539
5	01 0.0572	83 0.0499	52 0.0514	65 0.0509	6E 0.0529	55 0.0508	4D 0.0502	FE 0.0488	DD 0.0575	E7 0.0540	F0 0.0545	09 0.0504	A2 0.0526	E4 0.0519	81 0.0529	3E 0.0526

- Approach:

- To understand LRS, one must first understand the nature of AES. The ciphertext is created by combining (usually XOR) plaintext with the secret key. Each round of AES encryption has a round key from 1 to 10. These keys are calculated from the initial key 0. The last AES round encryption (round 10) has clear power signals. Thus, by aiming at the last round I will be able to extract the clearest power data. [8]
- While this leakage model produces a key extraction of PGE = 0, by definition, it means that when $PGE \leq 5$, then the key is known. However, it isn't the case in with this leakage model, since the extracted key is different from that of the actual key.

- Obtained key:

Index	0	1	2	3	5	6	7	8	9	10	11	12	13	14	15
Key	D0	14	F9	A8	EE	25	84	E1	3F	0C	C8	B6	63	0C	A6

- Result:

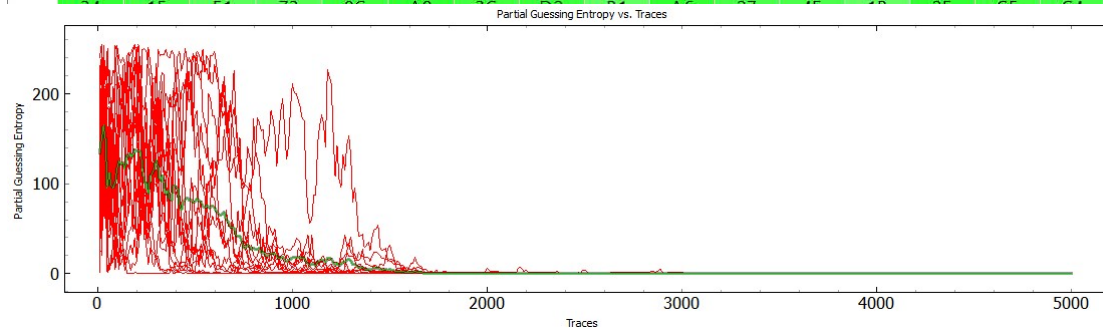
- The obtained key does not match with the original.
- The attack was unsuccessful, this method was not able to correctly extract the secret key from AES.

- Reason:

- While CPA can also be used for a black-box evaluation between the actual leakage of the device and the leakage model being used for CPA [1]. This means that the CPA approach will only work – if and only if – when the model is used for the CPA attack correlates with the actual target’s leakage model. Therefore, this does not work because there exists no correlation between our input model – LRS – to the provided AES’s leakage model.

Case 4: CPA LRS with noise = 0.01

Results Table																
PGE	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	D0	14	F9	A8	C9	EE	25	89	E1	3F	0C	C8	B6	63	0C	A6
1	0.1523	0.1226	0.0963	0.1001	0.1170	0.1596	0.1240	0.1323	0.0979	0.1437	0.1148	0.1147	0.0909	0.1397	0.1205	0.1120
2	E6	42	78	0A	4C	46	C4	4A	87	53	4F	3D	69	FA	CC	38
3	0.0690	0.0611	0.0586	0.0636	0.0644	0.0611	0.0643	0.0578	0.0641	0.0583	0.0585	0.0580	0.0587	0.0685	0.0628	0.0618
4	DD	25	21	8F	20	8C	3E	A2	44	AF	F0	8C	71	65	BF	A5
5	0.0686	0.0597	0.0581	0.0616	0.0587	0.0562	0.0597	0.0577	0.0610	0.0576	0.0571	0.0579	0.0585	0.0616	0.0612	0.0596
6	7C	F5	5E	2F	2F	D6	17	43	02	A1	6A	00	01	8B	2D	1E
7	0.0616	0.0596	0.0566	0.0577	0.0574	0.0558	0.0585	0.0559	0.0608	0.0575	0.0563	0.0576	0.0577	0.0563	0.0598	0.0593
8	1E	9F	7A	A5	C3	08	32	C9	E3	BC	16	E4	59	E0	67	75
9	0.0601	0.0590	0.0560	0.0564	0.0523	0.0555	0.0558	0.0542	0.0598	0.0573	0.0561	0.0559	0.0558	0.0558	0.0542	0.0591
10	C9	7A	1C	49	D3	EB	DB	9B	B8	DD	82	72	AE	13	0E	AD
11	0.0574	0.0569	0.0550	0.0558	0.0523	0.0554	0.0557	0.0539	0.0570	0.0568	0.0550	0.0559	0.0553	0.0550	0.0536	0.0544



- Obtained key:

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Key	D0	14	F9	A8	C9	EE	25	89	E1	3F	0C	C8	B6	63	0C	A6

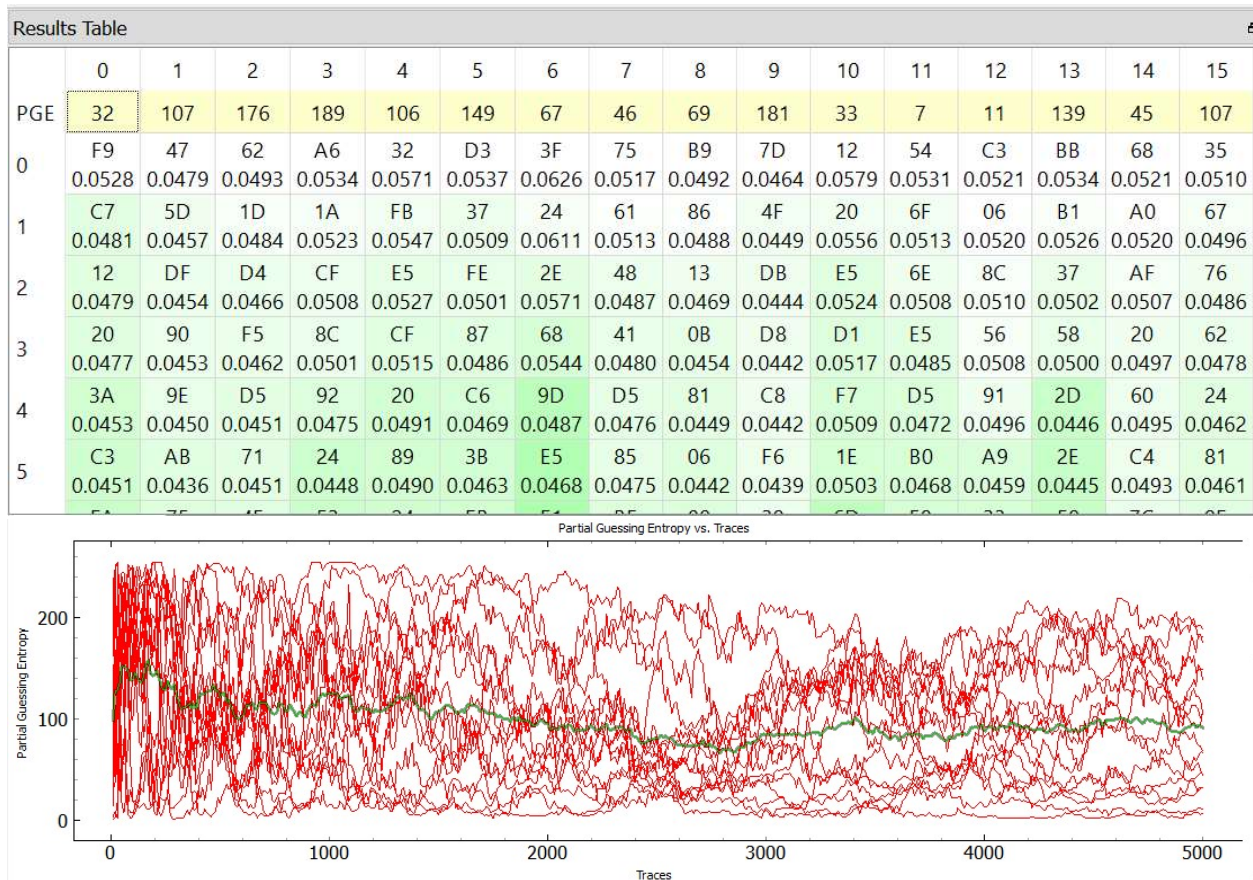
- Result:

- This does not work, given reason in Case 3, however, if we – theoretically – take Case 3 obtained key as the actual key. Then our code is working well despite the added noise since the obtained key in Case 4 is the same with that in Case 3.

- Reason:

- This is valuable because it allows us to test our code consistency under a noisy data set. After executing the attack numerous times, I noticed that the PGE in which the subkeys were found changed, but their values stayed the same. This means that the randomness retains, but the program keeps its consistency under these changes.

Case 5: CPA LRS leakage model with jitter = 3



- Obtained key:

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Key	D0	14	F9	A8	C9	EE	25	89	E1	3F	0C	C8	B6	63	0C	A6



- Result:











- This method does not work, because the obtained key is different from the original key. However, if we compare this to the one obtained from Case 3, the extracted key from Case 3 and 5 are the same.

- Reason:

- Similar to Case 4, after numerous executions, the subkeys' PGE changed but the subkeys' (hex) values stay the same. This means that the program is doing what it is supposed to do and with this, I am confident to say that the reason why the correct key was not obtained for Case 3,4 and 5 is due to the LRS leakage model does not correlate with the given AES's leakage model.

Result – Legends:

- Successful: Successfully obtained secret key or not
- Robustness: Program's consistency in obtaining key under various executions/conditions
- : Yes | : No

	CPAdefault	CPALRS	CPALRS-Noise	CPALRS- Jitter	CPASBOX- Noise and Jitter
Successful					
Robustness					

Note:

- PGE vs Trace datasheet:
 - o PGE stands for: Partial Guessing Entropy, that contains 0-255 elements, this is due to AES128_8 has 255 bits of information.
 - o PGE vs Trace graph allows us to observe the most likely guesses of the subkey byte. By measuring the correlation between the input leakage model to the target's leakage model.

By nature, CPA is a variant of DPA and as the name suggested, the only difference between the two is the last computational step: correlation vs differentiation.

- DPA operates by computing the difference between the average power consumption, then from power spikes, the program will be able to deduce whether or not there exist any correlations between the result of CPA. This observation of CPA is done by observing the resulting behavior of LSB average and spikes [1][5].
- CPA operates by computing the correlation between the input and actual leakage model, by observing the PGE vs trace plot, one will be able to grasp the understanding of power consumption and behavior of AES. This observation of CPA is done by evaluating the correlation coefficient between the calculated and collected power traces from AES [5][8]. This method is different from DPA because it is reliant on the leakage model, and as stated before, we can always guess the model, but the main job of CPA will be to calculate the correlation between the guessed model to the actual one. In this lab, we are looking at :
 - o S-Box: Combining (XORing) plaintext and secret key and through the help of an S-Box look-up table, a ciphertext is produced. [1][20]
 - o Last Round State: The ciphertext is created by combining (usually XOR) plaintext with the secret key. Each round of AES encryption has a round key from 1 to 10. These keys are calculated from the initial key 0. The last AES round encryption (round 10) has clear power signals. [8]

In addition, while PGE may report the success of key extraction when $PGE \leq 5$, such as in the case of LRS, however, it does not seem to be guaranteed that such will provide the correct extraction. This begs a question that, why does PGE for LRS reports 0, but was unable to extract the correct key? Because in theory, this should have worked, however, it isn't the case with LRS.

VII. Analysis and Reflection

Analysis – The experiment shows that its results align with what had been shown in section II (related works) and III (methodology). The results also confirm that my understanding regarding SCA’s operations and nature is correct, as there exist relations between CMOS, linearity, and noise – as stated in the above sections I, II, and III. SCA, specifically CPA, utilizes electricity, the fundamental and foundation of modern society – electrical power – to crack cryptographic operations. Aiming at the very source of technologies and devices, SCA bypasses any cryptographic/encryption processes by dissecting the very nature of these operations. No matter what it is, anything that pertains to the technology industry is subjected to the use of power and electricity. SCA is dangerous, and its potential is vast, due to its foundation and attack methods aimed directly at the lowest level of technological devices. To remedy this would result in 2 outcomes: overhauling the entirety of power concept in technological devices (i.e: a different source of power, or modify power source/circuit, etc), or by masking/hiding the data-dependent power consumption through various means (i.e: MUTE-AES, RNG, PUFs, etc).

It is worth noting that hiding and masking are two completely separate concepts, however, they achieve similar results. Hiding refers to the reduction of “data dependency of the power consumption measurable by the adversary” [16] by introducing noise or any randomization. Whereas masking – or secret sharing – seeks to “provide resistance by randomizing the intermediate values of the underlying algorithm” [16]. Thus, in this paper, these concepts fall under the category of increasing device’s resistance to SCA (could be software or hardware), so when masking or hiding is mentioned, they are being referred under this category.

The result shows a definite correlation between the hardware leakage model and the success of key extraction. This proves that each leakage model will be specifically tailored for each device, as leakage models are made possible by the linear property of transistors within the device (in the case of Power Analysis, it could be different for EM-emission, etc). Therefore, we could model the correlation because of “the idea that power consumption of the circuit is the function of data” – by Professor Wayne Burleson. This means that if data could be modeled as a function of power consumption, it also means that such a function could be distorted via the introduction of an erroneous variable – such as noise. Noise virtually exists in every system and is often the source of an uncontrollable and unavoidable variable within a system. Thus, the accuracy of a function or model will drop as noise exceeds the tolerable level. Since SCA follows a linear model, noise will prove to be fatal, as it will influence the data distribution, thus making it difficult to accurately analyze the trend line. If the trace samples are not sufficiently large enough when compared to the existing noise, it will cause disturbance and inconsistency in the SCA process. Due to this, many have considered using noise (or a source of randomness) to mask/hide the energy consumption of data-dependent parts – in order to fool adversaries. Adding noise will make the trace capturing process more difficult, as well as complicates the trace analyzing process. Though, since our traces were captured after-the-fact, there was no problem in obtaining the data-dependent traces due to noisy environment (as the traces themselves are already data-dependent, not the entirety of the consumption circuit). Ultimately, the reason behind using noise is to blur or prevent adversaries from being able to discern the relation

between data-dependent power consumption and static power consumption. With Moore's law, as mentioned in [13], devices will shrink in size over time, combined with the low power consumption of the circuit, less noise will be needed in the task of preventing SCA.

There are two primary ways (but not limited to) in which SCA can be combatted: masking and hiding. Both of these seek to introduce randomness and noise into the system to make the process of retrieving and analyzing much more difficult. This can be done via the introduction of a dummy variable, randomization process, etc. In other words, any process that could hide the hardware implementations of the underlying cryptographic algorithm (as this remains to be the biggest data leakage that SCA attacks make use of) [7]. In order to obtain this, one must be able to create a reliable source of RNG, or distortion. This has been demonstrated by numerous research, such as [12,15,16,17,18], each has its take on creating the desirable noise. The employed methods may differ, but they all seek to separate the linear relationship between the data-dependent power consumption and leakage model. These countermeasures could roughly be divided into two – general – category: hardware and software. However, software-based countermeasures remain to be widely used, due to its flexibility, and in many contexts, cheapest to be put into practical uses.

For hardware countermeasures, proposals have been along the line of increased emission, circuit power consumption, wave dynamic differential logic (WDDL), and MUTE-AES, etc [12]. However, aside from tweaking the power consumption, one novel method is a complete redesign of the circuitry through “decouples the main power supply from an internal power supply that is used to drive a single logic gate” [16]. This approach seeks to reevaluate and recreate the foundation of the device's circuitry, making the circuit itself resistant to SCA attacks. It is advantageous in that it will increase resistance to SCA while further the device's effectiveness in operation. However, this method is more difficult to implement, as it is a complete overhaul of existing blueprints and products. Some other interesting hardware countermeasures are WDDL, and MUTE-AES, nevertheless, as stated by Jacqueline et al, these complete overhaul countermeasures are “highly invasive and requires the entire encryption module to be redesigned, creating the potential for design flaws and removing the possibility of using standard IP blocks.” [12]. Therefore, hardware-based attacks are far and few due to their invasiveness, lack of scalability, and adaptivity to our current blueprints and designs. Though this does not mean that they are invalid solutions, speaking on a larger scale (of production, cost, and adaptivity) we would not prefer a complete overhaul, unless the new system is superior in every way and can be integrated without redesigning our current foundation. For example, the integration of electricity, such as lightbulbs, was possible because it was better than candles in every way and did not need a complete redesign of societal structure.

Thus, due to hardware invasiveness nature, its potential in overheads, and incompatibility with current designs, many have turned to software-based or less invasive methods. For example, Jacqueline et al showed that “by combining CR with either VN, the overall protection from CPA increases by about 71%” [12]. By utilizing the randomization existing within the hardware, it was possible to produce a source of randomization that could prevent SCA by distorting the environment around it. Another example proposed by Yusuke Nozaki et al to use

Lightweight PUFs to create a source of randomness [17]. This proposal hinged on the fact that each hardware has a slight variation when produced, thus each of them maintain their uniqueness due to slight production error/difference. However, this does not mean that such a method is unstable, while it is true that each piece of hardware is unique (in heat generation, etc), at the same time, they also possess reliability. Reliability is the idea that while devices are random and unpredictable, they always fall in some margin of errors that could be calculated/tolerated. Thus, by using PUFs as a unique randomizer, it was possible to add unique noises to each system, making it even more difficult for adversaries to determine the leakage model and similarity between each model.

It is worth noting that, while adding extra noise and randomization might help to prevent SCA, this method remains to be a double-edged sword in many solutions. As much as noise is a distortion to adversaries, it is also an extraneous, and unwanted element within the system. Therefore, it could potentially hamper the system's operations, thus decreasing the machine's overall productivity. This specification is also due to the cause of friction between industry vs academia, as the industry does not want to increase the power consumption of the device – as it will cause a rise in negative reviews. Since the bulk of users will remain oblivious to many security issues that impose on a piece of hardware, they will not be able to fathom the severity of SCA and other security loopholes. Thus, the top priority of users will be the products' ease-of-use, convenience, and power consumption. From this, it is painfully obvious that there exist opposing interests of production (company) vs consumption (users) vs security (academia). This triad has created an endless loop of problems for each other; however, such a relationship is also the drive for better products, and better solutions. In the simplest of terms, companies' products are made with users' interests in mind⁷ and to meet new demands/revolutionary ideas, academia will come up with potential solutions that will be picked and choose by companies. Thus, due to a combination of problems, countermeasures for SCA (and many other security proposals) have not been fully applied to the desired devices, leaving many dangerously under-protected.

Through this experiment, it is undeniable that there exists a link between the current CMOS transistors and leakage model, therefore, it is not out of the question that the current CMOS arrangements, models, and effectiveness are still upgradable/replaceable. Albeit changing something at the lowest level of technology is not always easy, however, we should not rule it out completely. This method might not be the most efficient short-term, but we need to remember as we further explore the capabilities of technologies, so will our foundations need to be evolved to meet the ever-growing demands. Similar to building a house, as the number of residents goes up, so will the size of the house – to accommodate new members. However, such an increase would only be possible if the foundation of the house is scalable and firm – as the foundation needs to be an immovable object that can withstand the test of time⁸.

7: if a company appeal to a larger audience, their income will increase

8: As scientific theory needs to be capable of withstanding the test of time while facing opposing ideas and peer-review

Personal Reflection – I felt that after this project, I learned a lot as both an Electrical Engineer, and Security Engineering. Security Engineer is “cool” to learn, in that it is very diverse and interactive. It has taught me a lot about engineering, and that there is more to engineering than innovation. It is good that an engineer needs to have a “heart” to sympathize, and the “mind” to innovate. This course, and specifically this project, has sparked a lot of my inquisitive nature, seeking to learn voluntarily and not by force – which I have enjoyed immensely.

If I have more time to sit down and talk with everyone and have more in-person office hours, I could have made this project much better with more feedbacks and knowledge exchange. My own shortcomings lie from the fact that I learn best in-person, and have someone I could converse these ideas too, however, as the world is heading towards an online-oriented environment, I too, should learn to adapt with the change of time – or else I would get swept away with the flow, unable to develop further. This class has made me realized a lot of things that I normally would not be able to learn from a classroom environment. Therefore, I would like to extend my informal gratitude to you – Professor Wayne Burleson.

Project Reflection – If I have one more week worth of time, I would have been to better organize the project, writing-wise, but I highly doubt that with my limited knowledge and capabilities regarding SCA, I would be able to do much with just one week of extension. However, it is undeniable that my writing itself would be more coherent and flows more naturally, with some extra time. There are some well-written papers regarding this project, but with my limited time, I have not been able to fully read and comprehend the depths and boons that each of them brings, that is my only regret.

If I have an extra month, I believe that I will be able to perform the experiment to the fullest, to be able to carefully analyze the data while bringing out my full potential on this subject. I will be able to do the things that I couldn’t have, such as rewriting the clumsily worded sentences or perform a more in-depth analysis of SCA. I believe that SCA is very important, as a reminder that attacks can come from the unexpected, and that while technology evolves, so will the attack methods. Should I have an extra month, my optimal work-pace will align with this extra time, this is because I will have time to carefully craft my paper with the speed of 1.5-2 pages per day. Since this project is supposed to be a group project, and the writing workload is generally divided; thus, having to write a 25-page essay is the first to me, and is a very daunting task. Nonetheless, I want my readers to know that I want to explain – to the best of my ability and time permitted – the importance of SCA, and the threats that it poses towards our technological advances.

Before I started this project, I had a back-up plan that I would go the extra mile during the DPA assignment, in case that I do not have enough time on my hand to start fresh testing. During the assignment, I made sure to add as many insights as I could into the DPA assignment, that was not required of me within the project’s guideline (such as additional research, extra conclusions, and trials). It seems that doing so was a good choice, else I daresay that I would not have completed the assignment otherwise.

VIII. Conclusion and Future Work

Conclusion – As electricity is – by no exaggeration – the heart of technological devices, it empowers and makes devices operational; thus, even before the advent of SCA, numerous countermeasures have been proposed to ensure the continuous flows of electricity – this issue remains true to this day. However, SCA takes advantage of the fundamental concept of machinery – power consumption – to break the protective encryption lock, which safeguards the private information resides within the machine. Machines are similar to humans, in the sense that we need to consume energy upon performing a task, and each task *requires* a different amount of energy consumption. Some proposals hinged on making every task consumes an equal (or added) amount of energy, which will prevent adversaries from detecting data-dependent energy consumption, however, that also means that we need to astronomically elevate the amount of energy consumption; thus, this approach was rejected by companies, as it is impractical and do not appeal to consumers at large. This means that adding noise by itself is not a solution, as this experiment shows that, to effectively hide the correlation, the noise’s magnitude must exceed that of the correlation. This is not optimal as it has too many overheads in power consumption. Therefore, instead of generating unnecessary noise, researchers found by using a preexisting source of randomization is both cost-effective and could lessen the toll on the device during the operational period. This has been shown by [12,15,16,17,18], by using hardware’s uniqueness, voltage source, clock randomization, cache detection, etc.

This experiment was able to confirm that for CPA to be able to extract the encryption/secret key resides within the encryption standard – specifically that of AES-128 – the theoretical leakage model must be similar to that of the device. While this concept is elementary, nonetheless, it is also the reason behind what sets CPA apart from DPA or SPA. According to [5], “Similarly, when a CPA attack is performed, the adversary guesses k and evaluates the correlation coefficient $P_{wk(j)}$ ” between the guessed W_i and the collected (given) power trace.

$$\rho_{WK}(j) = \frac{N \sum_{i=1}^N W_i \cdot PC_i(j) - \sum_{i=1}^N W_i \cdot \sum_{i=1}^N PC_i(j)}{N^2 \sigma_{W_i} \cdot \sigma_{PC_i(j)}}$$

where

$$\sigma_{W_i} = \sqrt{\frac{1}{N} \sum_{i=1}^N W_i^2 - \left(\frac{1}{N} \sum_{i=1}^N W_i \right)^2}$$

$$\sigma_{PC_i(j)} = \sqrt{\frac{1}{N} \sum_{i=1}^N PC_i(j)^2 - \left(\frac{1}{N} \sum_{i=1}^N PC_i(j) \right)^2}.$$

Figure 4. CPA Calculation [5]

The figure to the right, courtesy from [5], shows that the analysis that CPA utilizes when calculating the correlation between the guessed and collected power trace.

Through this assignment, I was able to extract the AES’s secret key through the S-Box leakage model, while I was unsuccessful at extracting the key via the LRS leakage model. This means that the model that has the highest correlation to the given AES’s model is S-Box, in other words, it is most likely that the AES traces leakage was through S-Box – or something similar.

To confirm, beyond the shadow of a doubt, that noise is not the reason for the failure of key extraction in other leakage models – that is to say, bad coding – I insisted on testing code robustness to reinforce my claim that the reason for failure was not due to faulty code. This is important because after numerous trials and under different conditions, the results stay the same. Which is why I decided to make a controlled and experimental group and based my analysis from their results. Here, the controlled group stands for the default state for each leakage model. It is worth noting that, the noise for these experiments are “tolerable”, meaning that the purpose of this noise only serves the purpose of checking the *program consistency* in obtaining the same output under different environment. This is important as: if the code is truly faulty, then it must detect different keys under different traces of the same encryption standard. Since these traces are fixed, thus, by adding noise, we effectively skewing these traces so that observing and analyzing these traces will be more difficult. However, as long as the noise does not exceed the correlation magnitude, these models still able to obtain the same key over and over again, proving that the code is not faulty. Thus, leaving the only possible conclusion is that the input leakage model is incorrect.

- For CPA attack with S-Box model:
 - o Controlled: Case 1 – No modifications to script, no noise.
 - o Experimental: Case 2 – Added noise and/or jitter.
- For CPA attack with LRS mode:
 - o Controlled: Case 3 – No modification to script, no noise.
 - o Experimental: Case 4, and 5 – Added noise and/or jitter.

Through numerous trials and extended research, I concluded that the reasons behind the unsuccessful attempt to extract the secret key via LRS were its uncorrelation with AES’s leakage model and not because of faulty code. I arrived at this conclusion because even under noisy environment, S-Box was still able to extract the encryption key from AES, whereas other inputs have failed – therefore, the failure was not due to noise, and the result shows a definite relation between S-Box leakage model and device’s trace leakage model.

Ultimately, a CPA attack is only successful if an input leakage model holds similarities to that of the device. In addition, noise is a double-edged sword in the quest to prevent SCA, as it increases the device’s resistance against SCA at the cost of some overheads in power consumption, area, and/or operational speed.

Future Works – In the future, I would want to delve deeper into the effects of better noise and randomization that can help devices’ resistance against SCA attacks. This paper only seeks to explain the potential of SCA and the overarching countermeasures that are currently widely employed. I would also like to further expand on the knowledge of this project to not only CPA, as CPA is certainly the best method in obtaining encryption key in a white-box analysis, however, there are certain areas where DPA is far superior. Besides, to observe the effect that noise has on CPA vs DPA is certainly a worthwhile subject to delve deeper into. Besides, I want to understand why LRS was unable to extract the correct key, even though its PGE_average = 0.

IX. Citation and Related Works

- [1] P. Kocher , J. Jaffe , B. Jun, and P. Rohatgi, “Introduction to differential power analysis,” SpringerReference, Mar. 2011.
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.225.3481&rep=rep1&type=pdf>
- [2] S. Bhunia and M. H. Tehranipoor, “Side-Channel Attacks,” in Hardware security: a hands-on learning approach, Cambridge, MA, MA: Morgan Kaufmann, an imprint of Elsevier, 2019, pp. 193–218. <https://www.sciencedirect.com/topics/computer-science/side-channel-attack#:~:text=4.2.,of%20cryptographic%20algorithms%20or%20software>.
- [3] F. Skopik, P. Smith, M. Hutle , and M. Kammerstetter, “Resilience Against Physical Attacks,” in Smart grid security innovative solutions for a modernized grid, Waltham, MA, MA: Elsevier, 2015, pp. 79–112. <https://www.sciencedirect.com/topics/computer-science/side-channel-attack#:~:text=4.2.,of%20cryptographic%20algorithms%20or%20software>.
- [4] Y. Li, M. Chen, and J. Wang, “Introduction to side-channel attacks and fault attacks,” 2016 Asia-Pacific International Symposium on Electromagnetic Compatibility (APEMC), 2016. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/7522801>
- [5] M. Alioto, M. Poli, and S. Rocchi, “Power analysis attacks to cryptographic circuits: a comparative analysis of DPA and CPA,” 2008 International Conference on Microelectronics, 2008. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/5393827>
- [6] E. Brier, C. Clavier, and F. Olivier, “Correlation Power Analysis with a Leakage Model,” Lecture Notes in Computer Science Cryptographic Hardware and Embedded Systems - CHES 2004, pp. 16–29, 2004. https://link.springer.com/chapter/10.1007/978-3-540-28632-5_2
- [7] Y. Zhou and D. G. Feng, “Side-Channel Attacks: Ten Years After Its Publication and ...,” Research Gate, 2005. [Online]. Available:
https://www.researchgate.net/publication/220333749_Side-Channel_Attacks_Ten_Years_After_Its_Publication_and_the_Impacts_on_Cryptographic_Module_Security_Testing. [Accessed: 18-Jun-2020].
<https://eprint.iacr.org/2005/388.pdf>
- [8] H. Mestiri, F. Kahri, B. Bouallegue, and M. Machhout, “A CPA attack against cryptographic hardware implementation on SASEBO-GII,” 2017 International Conference on Green Energy Conversion Systems (GECS), 2017. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/8066139>
- [9] F. R. Nuradha, S. D. Putra, Y. Kurniawan, and M. A. Rizqulloh, “Attack on AES Encryption Microcontroller Devices With Correlation Power Analysis,” 2019 International Symposium on Electronics and Smart Devices (ISESD), 2019. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/8909447>

- [10] N. Benhadjyoussef, H. Mestiri, M. Machhout, and R. Tourki, "Implementation of CPA analysis against AES design on FPGA," 2012 International Conference on Communications and Information Technology (ICCIT), 2012. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/6285774>
- [11] O. Lo, W. J. Buchanan, and D. Carson, "Power analysis attacks on the AES-128 S-box using differential power analysis (DPA) and correlation power analysis (CPA)," Journal of Cyber Security Technology, vol. 1, no. 2, pp. 88–107, 2016. <https://www.tandfonline.com/doi/full/10.1080/23742917.2016.1231523>
- [12] J. Lagasse, C. Bartoli, and W. Burleson, "Combining Clock and Voltage Noise Countermeasures Against Power Side-Channel Analysis," 2019 IEEE 30th International Conference on Application-specific Systems, Architectures and Processors (ASAP), 2019. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/8825109>
- [13] S. Guille, "Constructive Side-Channel Analysis and Secure Design," Google Books, 02-Aug-2017. [Online]. Available: [https://books.google.com/books?id=fx8vDwAAQBAJ&pg=PA197&lpg=PA197&dq=mores law and side channel analysis&source=bl&ots=BNrynzL_34&sig=ACfU3U09g68Vs8CdVR7KO3jRPKxLJnQweg&hl=en&sa=X&ved=2ahUKEwjaha6hvpnqAhX1mHIEHbYTBAgQ6AEwCXoECAoQAQ#v=onepage&q&f=false](https://books.google.com/books?id=fx8vDwAAQBAJ&pg=PA197&lpg=PA197&dq=mores+law+and+side+channel+analysis&source=bl&ots=BNrynzL_34&sig=ACfU3U09g68Vs8CdVR7KO3jRPKxLJnQweg&hl=en&sa=X&ved=2ahUKEwjaha6hvpnqAhX1mHIEHbYTBAgQ6AEwCXoECAoQAQ#v=onepage&q&f=false). [Accessed: 30-Jun-2020].
https://books.google.com/books?id=fx8vDwAAQBAJ&pg=PA197&lpg=PA197&dq=mores+law+and+side+channel+analysis&source=bl&ots=BNrynzL_34&sig=ACfU3U09g68Vs8CdVR7KO3jRPKxLJnQweg&hl=en&sa=X&ved=2ahUKEwjaha6hvpnqAhX1mHIEHbYTBAgQ6AEwCXoECAoQAQ#v=onepage&q&f=false
- [14] H. Kim, H. Yoon, Y. Shin, and J. Hur, "Cache Side-Channel Attack on Mail User Agent," 2020 International Conference on Information Networking (ICOIN), 2020. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/9016619>
- [15] T. Vateva-Gurova and N. Suri, "On the Detection of Side-Channel Attacks," 2018 IEEE 23rd Pacific Rim International Symposium on Dependable Computing (PRDC), 2018. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/8639584>
- [16] A. Gornik, A. Moradi, J. Oehm, and C. Paar, "A Hardware-Based Countermeasure to Reduce Side-Channel Leakage: Design, Implementation, and Evaluation," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 34, no. 8, pp. 1308–1319, 2015. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/7087376>
- [17] Y. Nozaki and M. Yoshikawa, "Countermeasure of Lightweight Physical Unclonable Function Against Side-Channel Attack," 2019 Cybersecurity and Cyberforensics Conference (CCC), 2019. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/8854557>
- [18] A. Baylis, G. Stitt, and A. Gordon-Ross, "Overlay-based side-channel countermeasures: A case study on correlated noise generation," 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), 2017. <https://ieeexplore-ieee-org.silk.library.umass.edu/document/8053171>

- [19] C. O'Flynn and Z. D. Chen, "Side channel power analysis of an AES-256 bootloader," 2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE), 2015. <https://eprint.iacr.org/2014/899.pdf>
- [20] C. O'Flynn, "Introduction to Side-Channel Power Analysis (SCA, DPA ...," *YouTube*, 21-Jan-2016. [Online]. Available: <https://www.youtube.com/watch?v=OlX-p4AGhWs>. [Accessed: 14-Jun-2020].