TU Delft

Delft University of Technology

Securing Power Side Channels by Design

Aljuffri, A.A.M.

DOI
10.4233/uuid:654f32ea-d3df-4804-8d67-eb2dd89d20e5

Publication date
2024

Document Version
Final published version

Citation (APA)
Aljuffri, A. A. M. (2024). *Securing Power Side Channels by Design*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:654f32ea-d3df-4804-8d67-eb2dd89d20e5

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Securing Power Side Channels By Design

Abdullah Aljuffri

SECURING POWER SIDE CHANNELS BY DESIGN

SECURING POWER SIDE CHANNELS BY DESIGN

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. dr. ir. T.H.J.J. van der Hagen,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op dinsdag 27 februari 2024 om 12:30 uur

door

Abdullah Alawi M. ALJUFFRI

Master of Science in Computer Engineering,
Delft University of Technology, Delft, Netherlands,
geboren te Jeddah, Saudi Arabia.

Dit proefschrift is goedgekeurd door de

promotor: Prof. dr. ir. S. Hamdioui
copromotor: Dr. ir. M. Taouil

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Prof. dr. ir. S. Hamdioui,	Technische Universiteit Delft
Dr. ir. M. Taouil,	Technische Universiteit Delft
Onafhankelijke leden:	
Prof. dr. ir G.N. Gaydadjiev	Technische Universiteit Delft
Prof. dr. ing. A. Obeid	KACST, Saudi Arabia
Prof. dr. G. Di Natale	U. Grenoble Alpes, France
Dr. G. Selimis	Axelera AI Eindhoven
Dr. J. Sepulveda	Airbus Defence & Space, Germany
Prof. dr. ir. R. Kooij	Technische Universiteit Delft, reservelid

Keywords: Side Channel Analysis, Power Attacks, Countermeasures, Leakage Assessment Framework
Printed by: Johannes Gutenberg
Front & Back: Beautiful cover art that captures the entire content of this thesis in a single illustration.

Copyright © 2023 by A. A. M. Aljuffri

ISBN 978-94-6384-544-1

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

Dedicated to my parent, my sisters, and my brothers

CONTENTS

SUMMARY

The security of electronic devices holds the greatest importance in the modern digital era, with one of the emerging challenges being the widespread occurrence of hardware attacks. The aforementioned attacks present a substantial risk to hardware devices, and it is of utmost importance to comprehend the potential detrimental effects they may cause. Side-channel attacks are a class of hardware attacks that exploit information unintentionally leaked by a device during its operation. These leaks manifest in various forms, including power consumption, time variations, and thermal dissipation. The fundamental danger posed by side-channel attacks is their ability to infer sensitive information from these unintended emissions. To address the heightened risks associated with side-channel attacks, this thesis focuses on three main research topics.

Side Channel Analysis: Side-channel attacks can manifest in various forms, depending on the specific leakage channels employed. The present study primarily focused on the investigation of three distinct categories of leakage, as it is hypothesized that these specific forms of leakage present the greatest potential risks. The aim of the analysis is to identify the optimal channels for creating an assessment framework. The selected leakages for analysis cover power consumption, temporal variations, and thermal attacks. Power consumption measurements provide valuable insights into the behavior and execution patterns of algorithmic operations, facilitating the identification of specific operations that are particularly vulnerable to attacks. There are other types of leakages that are similar, such as electromagnetic emissions. However, it is important to note that power consumption demonstrates considerably lower levels of noise. The use of time variations in evaluating operations is subject to certain limitations due to the need to wait for a response. Nevertheless, one notable advantage of these systems is their ability to offer convenient remote access, facilitated by their software-based calculation capabilities. Despite its inherent noise, thermal monitoring is employed in nearly all devices as a means to prevent overheating. The ability to remotely access this monitoring system is facilitated through software. Consequently, a meticulous examination is necessary to identify potential modes of assault.

Countermeasures: Cryptographic algorithms and other security primitives are the basic components of any cryptosystem. In their most optimized versions, these algorithms are frequently thought to be prone to side-channel attacks (SCAs), which necessitates the development of countermeasures. In this thesis, four countermeasures that have been developed are thoroughly analyzed. The countermeasures that were devised covered a wide range of algorithms, such as GIFT, RSA, and AES, and they were suitable for a variety of applications, including lightweight ones. The first countermeasure that has been developed makes use of an Advanced Encryption Standard (AES) implementation that is based on neural networks. This countermeasure's principal goal is to confuse the attacker by causing random fluctuations in power consumption. The second countermea-

sure is developed for asymmetric algorithms. This countermeasure's goal is to balance the leakage by making power consumption similar among all its executions. The goal of developing the third algorithm was to provide a countermeasure that is lightweight and tailored to symmetric algorithms. This countermeasure is based on the integration of balancing and randomization techniques. To ensure that the results of these two operations show balanced power behaviors in a random way, two instances of the SBOX operation are generated to complement each other. The fourth countermeasure involves the optimization of a widely known countermeasure named Domain-Oriented Masking (DOM) to adapt to lightweight applications. The countermeasure used in this research combines optimization techniques like resource sharing, module optimizations, and key-expansion bypassing.

Pre-silicon Leakage Assessment: After recognizing the importance of mitigating side-channel leakages and developing various countermeasures, the subsequent phase entails establishing a framework for evaluating these vulnerabilities. In contrast to software vulnerabilities, which can be addressed through patching at any given time, the mitigation of hardware vulnerabilities necessitates expensive modifications to the physical hardware. Hence, it is essential to develop a leakage assessment framework that can effectively evaluate the system during the design phase. In this thesis, we present an innovative and pioneering methodology that relies on the application of Generative Neural Networks (GANs). The methodology described herein signifies a substantial advancement in the pursuit of enhanced security in the field of chip design. This framework demonstrates outstanding ability to rapidly produce traces that closely correspond to those obtained from computer-aided design (CAD) processes. As a result, it enables the efficient validation of numerous countermeasures within a realistic timeframe.

SAMENVATTING

De beveiliging van elektronica heeft de grootste urgentie in het moderne digitale tijdperk. Een van de opkomende uitdagingen is de wijdverbreide opkomst van hardware-aanvallen. De eerder genoemde aanvallen vormen een aanzienlijk risico voor hardware, en het is van het grootste belang om de potentiële schadelijke effecten die ze kunnen veroorzaken, te begrijpen. Side-channel aanvallen vormen een categorie aanvallen die informatie benutten die onbedoeld door een apparaat wordt gelekt tijdens zijn werking. Deze lekken manifesteren zich in verschillende vormen, waaronder stroomverbruik, tijdsvariaties en thermische dissipatie. Het fundamentele gevaar dat side-channel aanvallen vormen, is hun vermogen om gevoelige informatie af te leiden uit deze onbedoelde manifestaties. Dit proefschrift richt zich op drie onderzoeksthema's om de verhoogde risico's van side-channel aanvallen aan te pakken.

Analyse van side-channels: side-channel aanvallen kunnen zich manifesteren in verschillende vormen, afhankelijk van de specifieke lekkanalen die worden gebruikt. Dit proefschrift richt zich voornamelijk op het onderzoek van drie verschillende categorieën lekken, aangezien wordt verondersteld dat deze specifieke vormen van lekken het grootste potentieel aan risico's met zich meebrengen. De geselecteerde lekken omvatten stroomverbruik, tijdsvariaties en thermische aanvallen. Metingen van het stroomverbruik bieden waardevolle inzichten in het gedrag en de uitvoeringspatronen van algoritmische bewerkingen. Dit maakt de identificatie van specifieke bewerkingen die bijzonder vatbaar zijn voor aanvallen makkelijker. Er zijn andere soorten lekken die vergelijkbaar zijn, zoals elektromagnetische emissies. Het is echter belangrijk op te merken dat het stroomverbruik aanzienlijk lagere niveaus van ruis vertoont. Het gebruik van tijdsvariaties bij het evalueren van bewerkingen is beperkt vanwege de noodzaak om te wachten op een respons. Desalniettemin geven deze systemen eenvoudig externe toegang door hun op software gebaseerde berekeningsmogelijkheden. Ondanks de inherente ruis wordt thermische monitoring bijna in alle apparaten gebruikt als middel om oververhitting te voorkomen. De mogelijkheid om op afstand toegang te krijgen tot dit monitorsysteem wordt vergemakkelijkt door software. Bijgevolg is een grondig onderzoek noodzakelijk om mogelijke aanvalsmodi te identificeren.

Tegenmaatregelen: Cryptografische algoritmen en andere beveiligingsmaatregelen vormen de basiscomponenten van elk cryptosysteem. In hun meest geoptimaliseerde versies worden deze algoritmen vaak als vatbaar voor side-channel aanvallen (SCA's) beschouwd, wat de ontwikkeling van tegenmaatregelen noodzakelijk maakt. In dit proefschrift worden vier tegenmaatregelen die zijn ontwikkeld grondig geanalyseerd. De bedachte tegenmaatregelen bestrijken een breed scala aan algoritmen, zoals GIFT, RSA en AES, en zijn geschikt voor verschillende toepassingen, waaronder eenvoudige toepassingen. De eerste tegenmaatregel die is ontwikkeld, maakt gebruik van een implementatie van de Advanced Encryption Standaard (AES) die is gebaseerd op neurale netwerken.

Het voornaamste doel van deze tegenmaatregel is om de aanvaller in verwarring te brengen door willekeurige schommelingen in het stroomverbruik te veroorzaken. De tweede tegenmaatregel is ontwikkeld voor asymmetrische algoritmen. Het doel van deze tegenmaatregel is om het lekken in evenwicht te brengen door het stroomverbruik vergelijkbaar te maken voor alle operaties. Het doel van de ontwikkeling van het derde algoritme was het bieden van een eenvoudige tegenmaatregel die is afgestemd op symmetrische algoritmen. Deze tegenmaatregel is gebaseerd op de integratie van balancerings- en randomisatietechnieken. Om ervoor te zorgen dat de resultaten van deze twee bewerkingen evenwichtig vermogensverbruik op een willekeurige manier laten zien, worden twee exemplaren van de SBOX-operatie gegenereerd om elkaar aan te vullen. De vierde tegenmaatregel omvat de optimalisatie van een bekende tegenmaatregel genaamd Domain-Oriented Masking (DOM) om zich aan te passen aan eenvoudige toepassingen. De in deze studie gebruikte tegenmaatregel combineert optimalisatietechnieken zoals het delen van middelen, module-optimalisaties en het omzeilen van sleuteluitbreiding.

Beoordeling van de pre-silicon lekstroom: Na het identificeren van het belang om side-channel lekken te verminderen en het ontwikkelen van verschillende tegenmaatregelen, omvat de volgende fase het opzetten van een kader om deze kwetsbaarheden te beoordelen. In tegenstelling tot softwarekwetsbaarheden, die op elk moment kunnen worden opgelost door patches, vereist het verminderen van hardwarekwetsbaarheden kostbare wijzigingen aan de fysieke hardware. Het is daarom essentieel om een lekbeoordelingskader te ontwikkelen dat het systeem effectief kan evalueren tijdens de ontwerpfase. In dit proefschrift presenteren we een innovatieve en baanbrekende methodologie die berust op het gebruik van Generative Neural Networks (GANs). De hier beschreven methodologie betekent een aanzienlijke verbetering van de beveiliging op het gebied van chipontwerp. Dit kader toont een opmerkelijk vermogen om snel signalen te produceren die nauw overeenkomen met die verkregen uit computerondersteunde ontwerpprocessen. Hierdoor maakt het een efficiënte validatie mogelijk van talrijke tegenmaatregelen binnen een realistisch tijdsbestek.

ACKNOWLEDGEMENTS

"To accomplish great things, we must not only act, but also dream; not only plan, but also believe," Anatole France. These words resonate deeply as I reflect on the journey that led to the completion of this PhD journey. It has been a path marked by both challenges and successes, a journey made possible by the unwavering support and belief of many. This thesis is not solely a product of my efforts; it stands as a testament to the collective guidance, encouragement, and belief of those who have been a part of this journey. I am immensely grateful for their invaluable contribution in turning what once was a dream into a tangible reality.

To begin, my deepest appreciation goes to my promoter, **Prof. Said Hamdioui**, whose unparalleled guidance and mentorship have been the bedrock of my PhD journey. Our interactions have left an indelible mark on my academic and personal growth. We may not have had many meetings during my PhD journey, but each meeting was not just a discussion but a profound learning experience, imparting lessons that extend far beyond the confines of this thesis. His meticulous approach to academic writing, particularly in the crafting of my first scientific paper, laid the foundation for my subsequent research endeavors. This guidance not only streamlined my writing process but also taught me how to effectively highlight my work, ensuring it resonates with and is appreciated by a wider audience. The wisdom you provide me in navigating the challenges of academic rejection has been invaluable. Your anecdote about a student whose paper was initially rejected but later won a best paper award at another conference has been a source of constant inspiration and a reminder of the subjective nature of academic review processes. This perspective has helped me maintain resilience and perseverance in the face of setbacks. For these insights and for your unwavering support in countless other ways, I am profoundly grateful.

In addition, my profound gratitude extends to my co-promoter, **Dr. Mottaqiallah Taouil**, whose harsh critique and relentless drive for excellence profoundly shaped both my academic skills and personal strength. Franklin D. Roosevelt said, "A smooth sea never made a skilled sailor," and this phrase resonates deeply when I reflect on **Dr. Taouil**'s mentorship. Our frequent interactions, which became a staple of my daily routine, were more than just meetings; they were crucibles in which my skills were honed and my convictions strengthened. The art of defending and appreciating my work, a skill so crucial in the academic world, was something I learned in these sessions. Despite being harsh in his criticism, **Dr. Taouil**'s approach was always well-balanced by his friendly demeanor and his lighthearted humor, which he brought to our discussions. This unique combination made it easier to receive and grow from his feedback. His commitment went beyond the call of duty, generously offering his time and insights during weekends and late nights. Such dedication was instrumental in enabling me to produce a substantial body of work. Initially, I admit, the intensity of his criticism was challenging to embrace.

However, as the famous saying goes, "We do not realize the value of something until we see its fruit," this became evident as I observed the remarkable improvement in my work and thought process. For his invaluable contributions, tireless support, and the lasting impact he has made, I am eternally thankful.

I was lucky to have a post-doc like **Dr. Cezar**, whose expertise and enthusiasm for our field were not only inspiring but also contagious. His approach to research, characterized by a deep analytical mind and a relentless pursuit of knowledge, has profoundly influenced my own academic journey. **Dr. Cezar's** ability to delve into the complexities of our subject matter, while maintaining clarity and coherence, provided me with a model of scholarly excellence to aspire to. Beyond his academic prowess, it was his approachability and willingness to share insights that truly set him apart. His mentorship extended beyond the confines of our research projects, offering guidance on navigating the broader aspects of academic life and career development. His dedication to mentoring, despite his own demanding schedule, is something I deeply respect and am grateful for. The guidance and support from **Dr. Cezar** have been pivotal in shaping not just this thesis but also my future aspirations as a researcher. For all these reasons, I extend my heartfelt gratitude to **Dr. Cezar**, whose influence will undoubtedly continue to resonate with me in my future endeavors.

As I reflect on the journey of my PhD, I am reminded of the invaluable role played by my colleagues, whose support extended far beyond the academic realm.

To **Dr. Arwa, Dr. Mahroo, Haji, Michael Mainemer, Eralb, Bianco, Reynaldi, Reborto, Dr.Moritz, Dr. Guilherme, Dr. Anh, Dr. Innocent, Dr. Jintao, Dr. Lei,Dr. Daniel, Dr. Abdulghader, Mouath Abu Lebdeh, Dr. Shayestah, Yun, Rujuta, Dr. Hande, Dr. Milica, Dr.Mahdi, Dr. Micheal (Taha), Oscar, Amin, Asmae, Fouwad, Gijs, Mark, Simon, Arne, Abhairaj, Geerten, Ramon, Luc, Medina, Ahmed, Abid, Alexndra, Michael Miao, Nourdin, Ahmed Aouichi, Luiza, Bruno, Sumit, Yash, Erbing, Mainak, Troya, Joyce, Lidwina, Trisha, Laura, Francis, and Erik**

Each one of you has been a vital part of my experience, contributing in ways that have enriched both my research and my personal growth. To those of you who shared insightful discussions and brainstorming sessions, your intellectual contributions helped clarify my thoughts and refine my research. The academic camaraderie we developed, marked by stimulating debates and collaborative work, has been a cornerstone of my scholarly development. Beyond the confines of our academic endeavors, I am equally grateful for the moments of levity and companionship we shared. Whether it was through casual conversations by the coffee machine, shared meals, or recreational activities, these interactions provided much-needed breaks from the rigors of research. They reminded me of the importance of balance and the value of building meaningful relationships. I am thankful for the diversity of perspectives you all brought into my life, the laughter we shared, and the unwavering support you offered. These experiences have shaped me in innumerable ways and will be cherished as an integral part of my PhD journey.

Embarking on this journey, I hadn't fully grasped the profound impact that teaching others would have on my own learning and personal growth. In this spirit, I would like to express my deepest gratitude to **Pradeep, Marc, Mudit, Erik, Rouyu and Laura** for providing me with an experience that was so life-changing. Through working together with each of you, I have been on a journey of mutual learning and discovery, where teaching

has become a channel through which I have gained a more profound comprehension of the subject matter. You were more than just students in my PhD experience; you were partners in a journey of shared development and education. Your contributions went beyond the role of students. To have had the chance to collaborate with such intelligent and enthusiastic minds is something for which I am extremely grateful. The knowledge that I gained from our time spent together is something that I will keep with me throughout the entirety of my professional life.

As I look back on my PhD journey, I am profoundly grateful for the friendships that blossomed during this significant chapter of my life. Being an international student presented its own set of challenges and adventures, and it was through these experiences that I formed bonds with an incredible group of people who became more than just friends; they became my extended family away from home.

To **Dr. Mohammad Al-Failakawi, Dr. Abdullah Alattas, Dr.Shareef, Dr. Ahmed Alwsheel, Dr. Abdulrahman Alsiri, Dr. Nawaf Almotairi Dr. Fouad Alasiri, Dr. Mohammed Almansori, Dr. Ahmed felimban, Dr. Yasser Algafas, Dr. Mohammed Alasiri, Abdullah Banjar, Kahlid Al-ameer, Yasser Alhiji, Dr. Fahad, Mohammed Alharthi, Sultan, Aymen, Dr. Ali, Dr. Abdulaziz, Naif ferhan, Mohhamed & hamzah Hemida, Rayan, Mouath, Dr. Arash, Dr. Prem, Dr. Mohammed Hamed, Hani, Dr. Fathi, Kobaib, Amer Alalem, Ronaldo, Sarah, Jolanda, Jessica, Gamzah, Dr. Javad, Dr. Ali, Emma, Raquel, Suzanna, Kitty, Leanart, Julia, Karen, Rustam, Linda, Marieke, and Nele** thank you for being a part of my journey. Your friendship was a beacon of support and joy amidst the highs and lows of my PhD experience. You introduced me to new cultures, shared local traditions, and helped me navigate the nuances of living in a foreign country. The moments we shared, from exploring the city, indulging in culinary adventures, to simple evenings spent in each other's company, provided a sense of belonging and community that was invaluable. Your willingness to listen, empathize, and offer a shoulder to lean on during challenging times was a source of immense comfort. Celebrating festivals, holidays, and personal milestones together, we created a mosaic of memories that I will treasure forever. I am grateful for the laughter, the conversations, the shared experiences, and the unwavering support. These friendships have not only enriched my stay but have also left an indelible mark on my heart. As I move forward, I carry with me the lessons learned, the joy experienced, and the beautiful friendships forged during this journey.

Throughout the course of my PhD studies, **the Saudi Culture Attaché** provided me with invaluable assistance and support, and I would like to express my deepest gratitude to them. My academic journey would not have been the same without their direction and assistance; they never failed to supply me with the resources and support I required to successfully navigate the challenges that come with pursuing a doctoral degree. The constant help that I received from the Attaché, which included everything from administrative assistance to cultural guidance, made my transition and stay abroad a great deal easier, which in turn enabled me to concentrate more intently on my academic pursuits. I would like to express my profound gratitude to the Attaché's team for their unwavering support and encouragement, which have been crucial to my academic achievement. Their dedication and commitment to providing assistance to Saudi students studying abroad is truly admirable.

I would like to extend my sincere appreciation to **King Abdulaziz City for Science and**

Technology (KACST) for the incredible opportunity they provided by funding my PhD studies. Their financial support has been a cornerstone of my academic journey, enabling me to dedicate myself fully to my research without the burden of financial constraints. The generosity of **KACST** not only facilitated my pursuit of higher education but also underscored their commitment to advancing knowledge and supporting scholars in their academic endeavors. This scholarship has been more than just a financial aid; it has been a vote of confidence in my potential, for which I am deeply honored. I am immensely thankful for their belief in my abilities and for investing in my future, thereby contributing significantly to the field of science and technology.

Finally, and most importantly, my heartfelt gratitude is reserved for my family: my parents, and my brothers and sisters. This journey would have been unimaginable without their unwavering love, encouragement, and sacrifice. To my parents, who have always been my pillars of strength and my guiding lights, your endless support and belief in my abilities have shaped the person I am today. Your wisdom, patience, and nurturing have been my constant source of motivation and resilience. To my siblings, thank you for your endless cheer, understanding, and the countless moments of light-hearted relief you provided during this intense journey. The sacrifices you all have made, the emotional support you have unfailingly provided, and the confidence you placed in me have been the bedrock of my accomplishments. Your unwavering faith in my potential, even in the face of challenges, has been a source of immense strength. The values you instilled in me, the home you have always kept open for me, and the laughter and love we shared, have been my sanctuary. I am forever indebted to my family for their selfless love, for being my first teachers, my biggest fans, and my strongest supporters. This achievement is as much yours as it is mine, and I dedicate this milestone to you with all my love and gratitude.

1

INTRODUCTION

This chapter presents a brief overview of the core topic explored in this thesis, as well as an explanation of its importance, an evaluation of relevant literature, a discussion of existing challenges and opportunities, and a description of the primary research topics and contributions that have been covered in this dissertation. Section 1.1 discusses the motivation behind and importance of cybersecurity, the rising interest of attackers towards hardware attacks, and sheds light on the rising prominence of power-based side channel attacks. Section 1.2 reviews the opportunities and challenges related to the side channel analysis domain. Section 1.3 outlines the research areas that are addressed within the context of this thesis. Section 1.4 covers the contributions presented in this thesis. Section 1.5 provides an overview of the thesis outline.

1

1.1. MOTIVATION

The ever-increasing complexity of today's technology results in vulnerabilities in hardware components, in addition to those in software, that pose a growing risk to the security infrastructure of electronic devices. This section provides an overview of the severity of hardware-level attacks, with a particular emphasis on the analysis of side-channel attacks. The aim is to highlight the need for further progress in hardware security in order to thwart the evolving sophistication of attack mechanisms.

1.1.1. THE THREAT OF HARDWARE LEVEL VULNERABILITIES

The inception of cybercrime in the digital era can be traced back to 1955 [1], when Allen Scherr exploited the MIT computer network system, unveiling an early vulnerability in computing networks. His actions underscored a moment in digital security, laying bare the relative ease with which individuals could compromise digital systems using simple means of the time, such as punch cards, setting an upsetting precedent for the future. This incident presented new cybersecurity issues, illustrating that as technology evolves, so will the means for using it for evil.

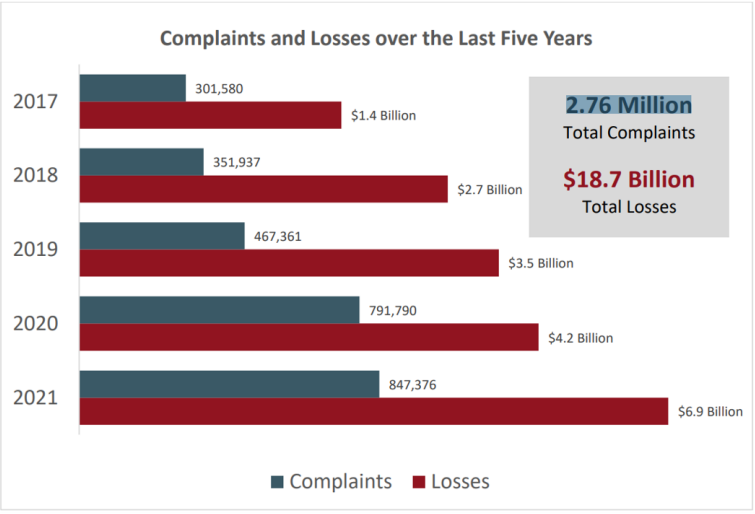


Figure 1.1: Cybercrime records from 2017 to 2021 based on FBI report [2]

Over the following five decades, the landscape of cybercrime has not only broadened but also intensified markedly. According to a report from the FBI [2], the number of cybercrime occurrences in the United States has reached an astounding 2.76 million instances, with financial losses equal to $18.7 billion in just five years (Figure 1.1 shows the complins and losses from year 2017 to 2021). Globally, the cybercrime economy is expected to reach $10.5 trillion by the year 2025 [3], making it among the third-largest economies in the world, along with the United States and China. The continual advancement in technology today leads to the emergence of vulnerabilities not just in software, but also in hardware components, thereby escalating the risks to the security framework

of electronic devices. The nature of hardware vulnerabilities is distinct from that of software due to the inability to update the physical hardware components. Therefore, upon discovering a bug, a simple patch cannot rectify the issue; instead, the physical hardware must be replaced. A 2019 report by Dell [4] elucidates the gravity of hardware security challenges, revealing that 63% of organizations reported experiencing at least one data breach in the preceding year attributable to hardware security vulnerabilities. This statistic underscores the pressing need for a more robust and proactive approach towards hardware security. Unlike software which can leverage community-driven vulnerabilities assessment programs such as bug bounty [5], addressing hardware vulnerabilities often requires the employment of specialized security engineers. The engagement of security experts, coupled with the potential necessity for design revisions, typically leads to a reduction in design efficiency and an extended time-to-market, hence increasing financial costs. In essence, addressing hardware vulnerabilities presents a more complex, time-consuming, and costly endeavor when compared to managing software vulnerabilities. This illustrates the various problems involved in protecting the security infrastructure in the face of the constantly changing technological landscape.

1.1.2. SIDE CHANNELS: THE UNDERESTIMATED HARDWARE ATTACKS

In recent years, the field of cybersecurity has seen an increase in the prevalence of hardware attacks. Examples of this type of attack include **Intellectual Property (IP) Piracy** [6], in which malicious entities copy and distribute hardware designs without authorization; **Hardware Trojans** [7], which are malicious alterations to the hardware that can cause unauthorized behavior or data leakage; and **Side Channel Attacks (SCAs)** [8], which exploit the physical characteristics of hardware implementations to gather sensitive information. Unlike other hardware attacks, **SCAs** are well-known for being undetectable and can be executed using relatively inexpensive equipment. In order to get insight into the ease of SCAs, one can consider the Simple Power Analysis (SPA) [9]. SPA is an example of a specific subset of SCAs that analyze the power consumption patterns exhibited by devices with the ultimate objective of uncovering confidential information. This attack allows the attacker to infer the power consumption patterns through simple observation, revealing the secret information of the cryptographic algorithm, such as RSA, by analyzing a single trace (i.e. one run). The implementation of RSA relies on the use of *Multiply* and *Square operations*. Because *Multiply* consumes more power, the attacker can identify the executed operations and subsequently obtain the key value, as depicted in the Figure.1.2.

Furthermore, physical access is no longer required for side channel attacks to be effective in many cases.. To understand the severity of this, we can examine the Spectre and Meltdown vulnerabilities that surfaced in 2018 [10]. These vulnerabilities belong to a category of side-channel attacks called microarchitectural attacks. In this type of attack, the adversary makes use of the microarchitectural properties of modern CPUs in order to obtain sensitive information and leak it. As shown in Figure 1.3, upon discovery, approximately 2.7 billion devices were found to be susceptible to these vulnerabilities [11]. Spectre and Meltdown epitomized the idea that side channel attacks have evolved beyond the need for physical access to the hardware, presenting a significant advancement in the capabilities of attackers. These illustrations emphasize the urgent need to

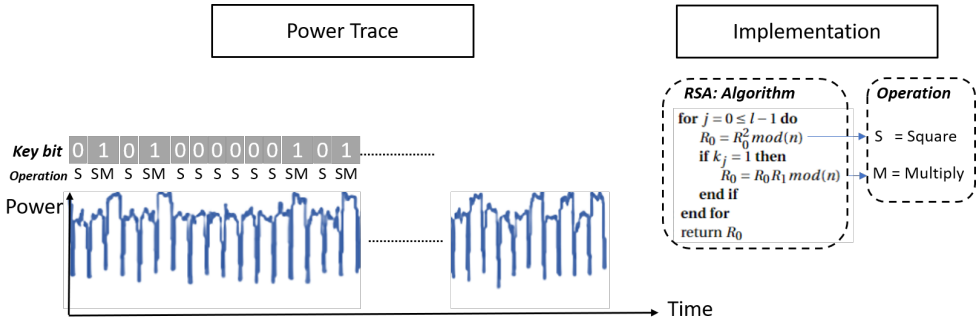


Figure 1.2: Simple Power Analysis [9]

develop effective defenses in order to reduce the growing risks posed by such attacks. The comprehensive comprehension and subsequent mitigation of potential hardware vulnerabilities play a crucial role in establishing a robust cybersecurity framework that can effectively withstand the ever-changing landscape of attacks.

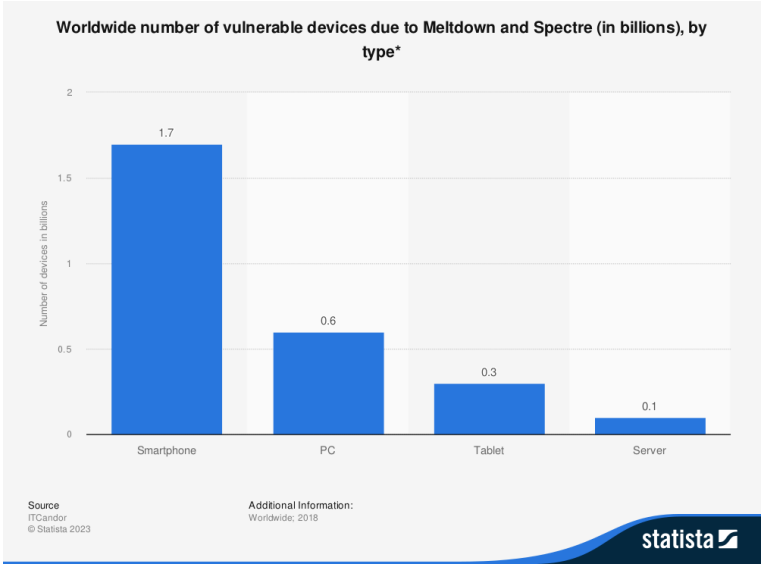


Figure 1.3: Global vulnerable devices by Meltdown and Spectre [11]

1.2. OPPORTUNITIES AND CHALLENGES

The domain of power attacks within the cybersecurity landscape presents both significant challenges and opportunities for organizations and individuals. With the ongoing development of technology, there is an ever-increasing demand for fresh techniques

and methods of defense against attacks. In this section, we will examine three essential domains of defending against side channel attacks' threats. Firstly, we shall delve into **countermeasures**, which act as a shield against potential exploitation. Secondly, the spotlight will shift to **pre-silicon leakage assessment**, serving as both a detection and preventive methods. Lastly, **post-silicon leakage assessment** will be discussed as a validation and evaluation tool, acting as a subsequent layer of assurance.

1.2.1. COUNTERMEASURES

Security primitives, including cryptographic algorithms, form the fundamental building blocks of any cryptosystem. In their most optimized forms, these algorithms are often seen as vulnerable to Side Channel Attacks (SCAs), necessitating the need for customized countermeasures. The demand for such countermeasures amplifies with the broadening spectrum of cryptographic algorithms and the varying application needs, which span from lightweight to highly sensitive applications.

- **Algorithms Variance:** Cryptographic algorithms such as symmetric encryption (AES [12], DES, GIFT [13]), asymmetric encryption (RSA [14], ECC [15]), and hashing algorithms (SHA-256 [16], MD5 [17]) manifest disparate levels of vulnerability to SCAs [9]. For instance, asymmetric algorithms like RSA are notably susceptible to timing attacks [18], while symmetric algorithms like AES [9] may be more prone to power analysis attacks. The structural and operational differences across these algorithms necessitate algorithm-specific countermeasures.

- **Implementations Levels:** The implementation of cryptographic algorithms can be achieved through either software or hardware, each presenting its own set of advantages and challenges. The effectiveness of software countermeasures [19] is frequently constrained by the hardware that lies behind them, particularly the instructions and architecture of the Central Process Unit (CPU). If a CPU has a cache that leaks information through timing or power channels, for example, it may be exceedingly difficult to mitigate this issue using software by itself. Hardware countermeasures [20], on the other hand, offer a solution that is both more efficient and more robust. However, it has a higher overhead area and is less flexible to update in response to new threats.

- **Applications Scope** In lightweight applications like IoT devices, the trade-off between security and computational resources is highly skewed towards minimizing resource utilization. Countermeasures against SCAs in such settings often focus on simple, resource-efficient strategies like balanced power consumption and basic masking techniques, which attempt to obfuscate the correlation between physical emanations and secret information. Conversely, in high-sensitivity applications like financial systems or military communications, the emphasis drastically shifts towards maximizing security. Advanced countermeasures like higher-order differential power analysis (DPA) [21] resistance, employing algorithmic transformations, and hardware-software co-design approaches are adopted. Additionally, proactive measures like rigorous testing and validation against known SCAs are integral in these settings.

1

1.2.2. PRE-SILICON LEAKAGE ASSESSMENT:

As stated before, it is important to note that hardware vulnerabilities, namely side channels, cannot be addressed in the same manner as software vulnerabilities due to the inherent inability to patch hardware. Hence, in order to mitigate the risk of side channel attacks, it is imperative to do an analysis of the electronic chip during the design phase. To accomplish this objective, it is imperative to devise methodologies for assessing leakage. However, these advancements present challenges and opportunities, as the existing design tools lack a leakage assessment feature. The challenges and opportunities can be observed in three distinct ways, namely Simulation Accuracy, Modeling Complexity, and Analyzing Time. These three aspects will now be examined in further depth.

Simulation Accuracy: In the pre-silicon stage, the quality of the simulation is of the utmost importance in order to provide a realistic portrayal of the behavior of the hardware and its interaction with the environment. The stage at which the design is being simulated determines the accuracy of the outcome. It is known that simulation at lower levels, such as transistor-level, provides a better realization of the design behavior. Another factor that can affect the simulation's accuracy is the sampling rate. The higher the sampling rate, the higher the accuracy of the results. Accurate simulation models are useful in identifying potential side-channel vulnerabilities that power analysis attacks might exploit. These vulnerabilities could be exploited in a variety of ways. On the other hand, reaching a high level of precision is filled with difficulties. One major issue is the speed of the simulation. Simulating at a lower level of the design and the increase in the sample rate affect the performance negatively. The possibility lies in the development of balanced simulation tools and methods that are able to model switching activities with high precision and reasonable speed, which can therefore provide a valid foundation for leakage evaluation.

Modeling Complexity: It is impossible to overstate how difficult it is to simulate all of the many operational states that can be present in a semiconductor device. In order to determine whether or not a system is vulnerable to power attacks, each operational state may display a different set of power consumption patterns. These patterns have to be represented appropriately. The currently available design tools are not designed for power analysis. Hence, it could not have the level of sophistication required to manage such a high level of modeling complexity, which could result in certain vulnerabilities going undiscovered. Nevertheless, this obstacle presents an opportunity for creativity in the form of the development of fresh modeling approaches and tools. It is possible to establish new frameworks that provide a more nuanced and comprehensive understanding of the power consumption model. This will result in an improvement in the quality and effectiveness of leakage evaluations.

Analyzing Time: The amount of time needed to carry out comprehensive leakage analyses is a considerable obstacle, particularly when considering the pressure that exists to reduce the amount of time it takes to bring a product to market. Extensive simulations and analyses may cause the design phase to take a much longer amount of time, which may compromise the hardware product's ability to compete effectively. In-depth leakage evaluations also need a significant amount of resources, which may drive up expenses even more. The flip side of this coin is that there is a great opportunity to build analysis tools and procedures that are faster and more efficient. It is possible that

the amount of time spent analyzing can be greatly cut down by utilizing developments in computational capabilities and parallel processing. However, this would have to be done without sacrificing the comprehensiveness of the leakage evaluation. In addition, the use of machine learning and artificial intelligence might be exploited to automate and speed up certain portions of the study, thereby rendering pre-silicon leakage assessment more useful and cost-effective.

1.2.3. POST-SILICON LEAKAGE ASSESSMENT

The assessment of the resilience of chip design against side channel vulnerabilities involves the examination of side channel leakage using commonly accessible security techniques and equipment, such as those offered by Rambus [5] or Riscure [6]. Nevertheless, these methodologies are confronted with a myriad of challenges, including, but not limited to, leakage trace misalignment, noise accumulation, etc. The underlying cause of these issues can be attributed to the chip's design, which does not adequately accommodate side channel testing processes. This field of study, also referred to as post-silicon leakage assessment, is still in its early stages of development. Therefore, this has created several chances for the implementation of various techniques in the fields of noise cancellation, leakage augmentation, trace alignment, and other related areas.

1.3. RESEARCH TOPICS

The primary objective of this thesis is to provide an answer to the essential question "Which strategies are the most effective at protecting against side channel attacks?" In order to provide a full response to this important question, it is essential to investigate a range of characteristics that fall under the umbrella of the vulnerability category known as side channels. These aspects are summarized in the following sub-questions: Identifying the most suitable side channel for conducting leakage assessment that can effectively detect the majority of vulnerabilities.; developing strategies to strengthen the countermeasures; and locating the method that is the most efficient to assesses these countermeasures at the design stage.

1.3.1. SIDE CHANNELS ANALYSIS

Researchers and individuals have been investigating different types of side channel attacks in order to determine which approach demonstrates the highest rate of success. One of the forms of attacks is Photonic Emission Attacks, which encompass the analysis of light generated by hardware components during their operational state in order to deduce critical information. Scan-Based Attacks refer to a type of security threat that involves the analysis of hardware responses to specified scan patterns. The primary objective of these attacks is to find weaknesses inside a system or extract sensitive information. Timing Attacks involve the measurement of the system's response time during specific operations in order to infer valuable information. Power Analysis Attacks entail the measurement of a device's power consumption to gain insights into its internal processes and potentially uncover sensitive data. Electromagnetic Field Attacks exploit the electromagnetic emissions emitted by a device. Acoustic Attacks involve the analysis of

sounds emitted by a device, such as the noise produced by a CPU or keyboard, to gather sensitive information. The aim of this research is to examine various forms of attacks, namely Power attacks, Time attacks, and Thermal attacks, in order to identify one approach for implementing a leakage assessment technique that can effectively detect the majority of vulnerabilities.

1.3.2. COUNTERMEASURES

In the previous section, we explored the intrinsic variability of countermeasures, which is determined by their specific algorithms, levels of implementation, and applications. The presence of variability emphasizes the need for a comprehensive strategy for implementing countermeasures, ensuring that they are effectively customized to address the unique challenges presented by various applications. In the context of this research, we undertook a comprehensive examination and implementation of a variety of strategies to address the issue at hand. The purpose of these countermeasures is to establish a robust protective barrier that spans across a range of algorithms, levels of implementation, and fields of application. Our research primarily centered on several algorithms aligned with NIST standardization, such as AES and GIFT, which are based on symmetric cryptography. Additionally, we also explored RSA, which operates on the principles of asymmetric cryptography. In terms of scope, our study encompassed both software and hardware implementations, exploring the intricate dynamics associated with each. Regarding the applications, our development encompassed both the lightweight and high-sensitivity domains, in addition to the normal range of applications. Our objective is not simply to provide a variety of solutions, but rather to offer a systematic approach that can be utilized for various algorithms, levels of implementation, and applications to strengthen defenses against side channel attacks.

1.3.3. ASSESSMENT FRAMEWORK

Earlier, in the opportunities and challenges section, it was mentioned that there is a requirement for researching both pre- and post-silicon leakage assessment. Because it is of the utmost importance, the pre-silicon leakage assessment will be the primary focus of this study. While post-silicon assessment can be conducted through standard methodologies yielding comparable results, the pre-silicon stage necessitates substantial enhancements to ensure the fabrication of a robust chip resilient to side channel attacks. The numerous facets, such as speed and precision, are going to be the primary focus of our attention. The purpose of this project is to develop a high-speed leakage analysis tool that can evaluate the security primitives component of the chip with the best feasible degree of precision. To get to the conclusion that we are going to look into some methods, such as artificial intelligence. The focus of our investigation will be directed towards examining several aspects, particularly the factors of speed and accuracy, in order to design an advanced tool for analyzing leaks at high speeds. The purpose of this tool is to thoroughly examine the security primitives component of the chip with a high level of accuracy. This will help strengthen the protection against vulnerabilities related to side channel attacks during the early phases of chip development. The use of a proactive strategy plays a crucial role in mitigating potential security concerns, as it es-

tablishes a strong architectural basis that is essential in preventing side channel attacks.

1.4. THESIS CONTRIBUTIONS

This section covers the significant contributions made by the present thesis based on the previously stated research topics

Investigating and Analysis of pre-processing technique on Power Side Channels [22]: This study presents a detailed investigation of information leakage that occurs due to the power behavior of the cryptographic algorithm. The discussion offers an analysis of several pre-processing techniques, including three pre-processing techniques that have not been studied yet for DL-based SCAs. Hence, five different methods are explored: i) Data augmentation [23, 24], ii) data transformation [25, 26], iii) data concatenation [27], iv) stacked auto-encoder [28]; and v) stacked auto-encoder with encoder only [29]. Note that data augmentation and stacked auto-encoder are already applied in the literature. Data transformation has been explored in some power attacks like CPA [25] and MLP-based [26], but not yet in DL-based attacks. The other two techniques come from the image processing field due to their outstanding results. To our best knowledge, these three methods are being applied for the first time in DL-based SCAs.

Investigating and Analysis of Time Side Channel Attacks [30]: This study centers on the vulnerbilities arising from disparities in the processing time of electronic devices when executing various operations. The presence of small temporal disparities can unintentionally reveal significant details regarding the functioning or algorithms employed by a particular device. In the study, we propose GRINCH, the first cache-time base attack on GIFT. Caches are usually shared memories that are used to speed up the execution of cryptographic algorithms. However, they become a security threat when mutually accessed by multiple processes. A malicious process may gather information to reveal the secret key by: observing the execution time (time-driven attack) [31], exploiting the access pattern (access-driven attack) [32], or inferring the sequence of hits and misses (trace-driven attack) [33]. GRINCH crafts specific inputs to the cipher to extract sensitive data by observing its cache accesses. Hence, it is a cache time attack.

Investigating and Analysis of Thermal Side Channels [34]: In this study, we perform a comprehensive investigation on the practicality of performing thermal SCAs. We target asymmetric cryptographic algorithms as they are typically more computationally intensive than symmetric ones, which in theory means that they generate more heat. Since temperature has a slow response compared to power, intensive computational tasks provide better quality traces, as shown in [35]. Hence, we show how known SCA techniques like Simple Power Analysis (SPA), Correlation Power Analysis (CPA) and Deep-Learning Power Analysis (DL-SCA) can be adapted for thermal attacks. Additionally, we propose a novel thermal attack by modifying the CPA attack; it achieves a successful and complete key recovery. We refer to this attack as Progressive Correlation Thermal Analysis (PCTA). All these attacks have been evaluated on unprotected and protected versions of an RSA software implementation. Finally, we present a comparison of all these SCAs techniques using both thermal and power leakage to clarify how powerful temperature-based attacks can be.

Development of Symmetric based Countermeasures [19]: This study proposes a rad-

ical new countermeasure type that aims to break the linear correlation between the power consumption and the leakage model. We realize this by substituting the SBOX operation of AES with a neural network which we call S-NET (short for substitution neural network). Due to the chaotic nature of S-NET and removal of the linear power-leakage relation, we classify this countermeasure as *confusion*. The main contributions of this paper can be summarized as follow: First we explained the Proposed of S-NET: a new countermeasure based on *confusion*. It nullifies power attacks by invalidating the existing power-leakage models. Followed by Implementation of S-NET. This includes designing, training, and testing of an appropriate neural network. Finally Validating S-NET security using conformance testing by applying signal-to-noise ratio analysis as a leakage assessment and evaluation style testing by applying key ranking analysis based on the most popular power attacks.

Development of Asymmetric Based Countermeasure [36]: This paper proposes a new countermeasure for asymmetric algorithms that maintains a constant execution pattern regardless of the key without adding extra cost. It achieves this by considering multiple bits simultaneously and the reordering of operations (e.g., square and multiplication in RSA, and double and add in ECC). We refer to this countermeasure as *multi-bit blinding*. The proposed method can be applied to both RSA and ECC. However, in this paper, we focus only on the RSA as a low cost solution has a greater impact when larger key sizes are used. The main contributions of this paper are: First, Proposal of a new countermeasure (i.e., multi-bit blinding) for asymmetric algorithms against power-based side channel attacks. Folllowed by Demonstration of the proposed method using two different implementations of the RSA algorithms: one based on naive square and multiplication operations and the other based on Montgomery multiplication [37]. Finally Validation of the proposed method using two types of side channel attacks techniques: profiled [38] and non-profiled attacks [39].

Development of lightweight based Countermeasures [40, 41]: Two studies are considered to be lightweight countermeasures. The initial study examines the vulnerabilities present in software implementations of GIFT [41], a lightweight block cipher. The analysis is conducted utilizing both non-profiled attacks, such as CPA, and profiled attacks, such as deep learning attacks. A novel set of countermeasures is subsequently proposed. The countermeasure presented in this study is formulated through an analysis of two widely utilized techniques in countermeasure development, namely balancing and masking. By merging these two approaches, a novel variation called balanced dual-mask is proposed. The seconed study, presents a novel low-area, low-power and low-latency AES hardware accelerator [40]. Our method takes advantage of the fact that the key remains unchanged throughout a communication session, eliminating the need for repeated execution of the key expansion module. Additionally, we integrate an improved version of the Domain-Oriented Masking (DOM) which is one of the most advanced countermeasures against side channel attacks (SCAs). Our DOM-based AES design is more area-efficient in comparison to the original DOM design.

Development of a leakage assessment technique based on Artificial Intelligence [42]: This study presents a novel method to speed up the CAD based assessment by generating reliable power traces at design. We first train a Generative Adversarial Network (GAN) based on CAD-based power traces and their corresponding switching activity. Subse-

quently, the GAN is used to generate new power traces using the switching activity as input only.

1.5. THESIS ORGANIZATION

The remaining parts of this report are broken up into the aforementioned five chapters. The second chapter of this thesis presents background information and an outline of the subjects that are relevant to the area of research that is being conducted here. In the third chapter, we will discuss the technique and implementation of our proposed atacks. In the fourth chapter, the defenses created against power attacks are broken down and explained. The leakage assessment that was designed to evaluate the proposed countermeasures is explained in detail in chapter five. The last chapter concludes and summarizes this thesis. The following list offers a more in-depth explanation of the subjects that are covered in each chapter.

Chapter 2: This chapter presents the essential contextual information pertaining to the subject matter addressed in this thesis. start with an examination of fundamental cryptographic algorithms such as AES, RSA, and GIFT. The subsequent discussion delves into the elucidation of side-channel attacks, including a description of both profiled and non-profiled attack methodologies. Subsequently, a classification of countermeasures employed to mitigate side channel threats is presented. In conclusion, elucidate the techniques employed for conducting leakage assessments.

Chapter 3: This chapter is dedicated to examining three distinct channels, namely power, time, and thermal. The selection of these channels has been undertaken with the aim of developing evaluation methodologies for a singular channel.

Chapter 4: This chapter outlines the four countermeasures that were developed during the course of the present study. First, it introduces a neural network-driven version of the Advanced Encryption Standard (AES) algorithm with the objective of obfuscating the adversary. Then a description of a countermeasure approach for addressing the issue of balancing the power consumption in asymmetric algorithms such as RSA and ECC After that, a countermeasure that integrates randomization and balancing techniques in a lightweight manner Finally, an optimized implementation of the DOM-based Advanced Encryption Standard (AES).

Chapter 5: This chapter centers on the investigation of pre-silicon leakage assessment methodologies and proposes a mechanism for analyzing micro-electronic chips against power attacks in the design phase.

Chapter 6: it provides a summary of this thesis and discusses potential future work.

2

BACKGROUND

This chapter provides the necessary background on the topic covered in this thesis. Section 2.1commences with an examination of fundamental cryptographic algorithms such as AES, RSA, and GIFT. Subsequently, Section 2.2 delves into the illustration of side-channel attacks, including an explanation of both profiled and non-profiled attacks. Following this, Section 2.3 provides a classification of countermeasures implemented to mitigate side channel threats, classifying them according to their behavior into balancing and obfuscating techniques. Finally, in Section 2.4, explain the leakage assessment methods.

This chapter is partially based on survey published on 16th International Conference on Design & Technology of Integrated Systems in Nanoscale Era, DTIS 2021 [43].

2.1. CRYPTOGRAPHIC ALGORITHMS OVERVIEW

In a digital world that is filled with adversarial risks, cryptographic algorithms have become an essential component in the process of protecting communication, information, and systems. These algorithms make it possible to encrypt data, which protects not only its privacy but also its integrity and, in some instances, its authenticity. There are a number of cryptographic algorithms, but only a few of them, such as the Advanced Encryption Standard (AES) [12], Rivest-Shamir-Adleman (RSA) [14], and the lightweight cipher GIFT [13], have gained popularity due to the powerful security features and efficient operation that they offer. Next, each of these three algorithms will be explained in further detail.

2.1.1. ADVANCED ENCRYPTION STANDARD (AES)

AES [12] is a symmetric cryptographic algorithm that is used in the cyber world for the purpose of encrypting and decrypting data in order to protect them from cyberattacks. It has a fixed data block size of 128 bits, and key lengths of 128, 192, or 256 bits. The key length determines the number of rounds required: 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. The 128-bit data block is divided into 16 bytes, which are mapped to a 4 × 4 array referred to as the State array. The diagram of AES encryption and decryption flow is presented in Figure 2.1. Each round of encryption includes four primary modules: *SubBytes*, *ShiftRows*, *MixColumns*, and *AddRoundKey*, except round 0 and last round (see Figure 2.1).

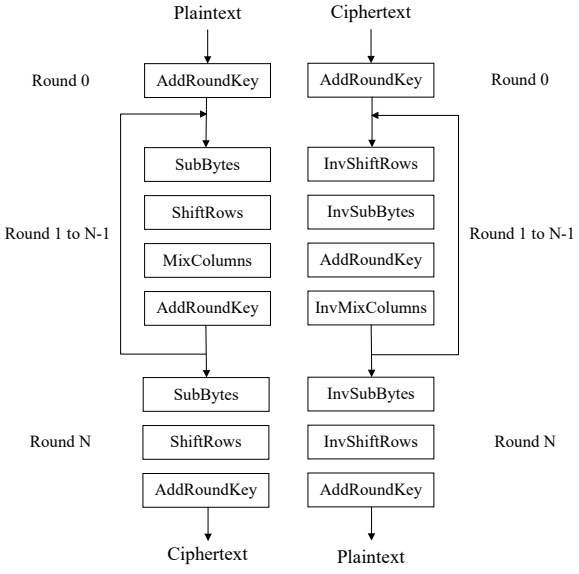


Figure 2.1: AES Encryption & Decryption Flow Diagram

AddRoundKey module involves bit-wise XOR operations of the round key and State array. *SubBytes* module is the only nonlinear module in the AES and plays a crucial role

in defending against linear crypt-analysis[21]. When performing *SubBytes* module, each byte in the state array is substituted with another byte using 16-byte SBOX. The SBOX is generated using a combination of a multiplicative inverse in Galois Field GF(2^8) and an affine transformation [44]. *ShiftRows* module is a transformation that cyclically shifts the second, third, and fourth rows of the State array by one, two, and three bytes to the left, respectively, while leaving the first row unchanged. The *InvShiftRows* module is computed by performing the corresponding rotations to the right. *MixColumns* module and *InvMixColumns* module perform a modular polynomial multiplication in Galois Field GF(2^8) on each column of the State array. Equation 2.1 represent the *MixColumns* module transformation, and equation 2.2 represent the *InvMixColumns* module transformation, where $0 \leq j \leq 3$.

$$\begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{bmatrix} \tag{2.1}$$

$$\begin{bmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} \tag{2.2}$$

2.1.2. RSA AN ASYMMETRIC ALGORITHM

The concept of asymmetric cryptography (i.e. public-key cryptosystems) was introduced by Whitfield Diffie and Martin Hellman in 1976 [45]. Their goal was to have a solution for the secret key distribution problem of symmetric algorithms, especially when an unsecured communication channel is used. Asymmetric cryptography works as follows: each entity has two keys, namely a public and private key. As the name implies, the public key is publicly available to everyone and is mainly used for encrypting the messages. The secret key is only known by the receiver and is used to decrypt the messages. A special mathematical relation exists between the secret and public key that allows such computations to take place in a secure manner. Anyone can encrypt a message using the public key. However, only the possessor of the private key belonging to that public key can decrypt the message. Similarly, the concept can be used to sign documents where the owner can sign the document using his secret key. The signature can be verified publicly using the public key.

Examples of public-key cryptosystems are RSA [14], named after the initials of its inventors and ECC [15] (an abbreviation of Elliptic Curve Cryptography). RSA's security is inherited from the hardness of the integer factorization problem [46], while the security of ECC comes from the elliptic curve discrete logarithm problem [46]. With respect to the implementation, the RSA algorithm is based on modular exponentiation, which is performed by square and multiply operations (see Algorithm 1). The algorithm can be used both to encrypt a message and decrypt a ciphertext. As mentioned before, in this paper we analyze the proposed countermeasure only for RSA implementations. while the implementation of ECC is based on scalar multiplication, which is realized by double and

addition operations (see Algorithm 2).

Algorithm 1 Square-Multiply

INPUT (M, k, n); **where** M presents the text, k the key which can be represented by its binary represenation as $k_0 \cdots k_{l-1}$ where $k_j \epsilon \{0,1\}$, and n the modulus.
OUTPUT(R_0); **where** $R_0 = M^k mod(n)$
$R_0 = 1$
$R_1 = M$
for $j = 0 \leq l-1$ **do**
    $R_0 = R_0^2 mod(n)$
    if $k_j = 1$ **then**
        $R_0 = R_0 R_1 mod(n)$
    end if
end for
return R_0

Algorithm 2 Double-Addition

INPUT (P, d); **w**here P is a point on an elliptic curve and d a scalar which can be represented by
its binary represenation as $d_0 \cdots d_{t-1}$, where $d_j \epsilon \{0,1\}$
OUTPUT (T); **w**here $T = d \cdot P$
$T = P$
for $j = 0; j \leq t-1$ **do**
    $T = double(T)$
    if $d_j = 1$ **then**
        $T = add(T, P)$
    end if
end for
return T

2.1.3. GIFT A LIGHTWEIGHT CRYPTOGRAPHY

GIFT is a lightweight block cipher designed by Banik et al [13] as an improved to PRESENT. For performance reasons, GIFT substitution blocks are smaller than PRESENT and use less number of rounds. This makes the GIFT design very compact with a high throughput. There are two versions of GIFT, namely GIFT-64 and GIFT-128. The GIFT-64 uses 28 rounds with a 64-bit block size, while the GIFT-128 40 rounds with a 128-bit block size. The key size is the same in both versions (i.e., 128 bits). As shown in Figure 2.2, each round of the GIFT cipher consists of four functions [13]: *SubCells, PermuBits, AddRoundKey*, and Round Constant. Each function will be described briefly next.After each round the key is updated and hence each round uses a different key. Next, each function will be described briefly for the GIFT-128 version as it is the focus of this paper. For further details we refer the reader to [13].

SubCells: This function processes the 128-bit round input based on 4-bit data segments. Each 4-bit segment is replaced using a substitution box (SBox). The inputs and outputs of the SBox have a non-linear relation.

PermBits: This function processes its input state at bit-level. The SubCells' outputs are shuffled based on a fixed reordering scheme.

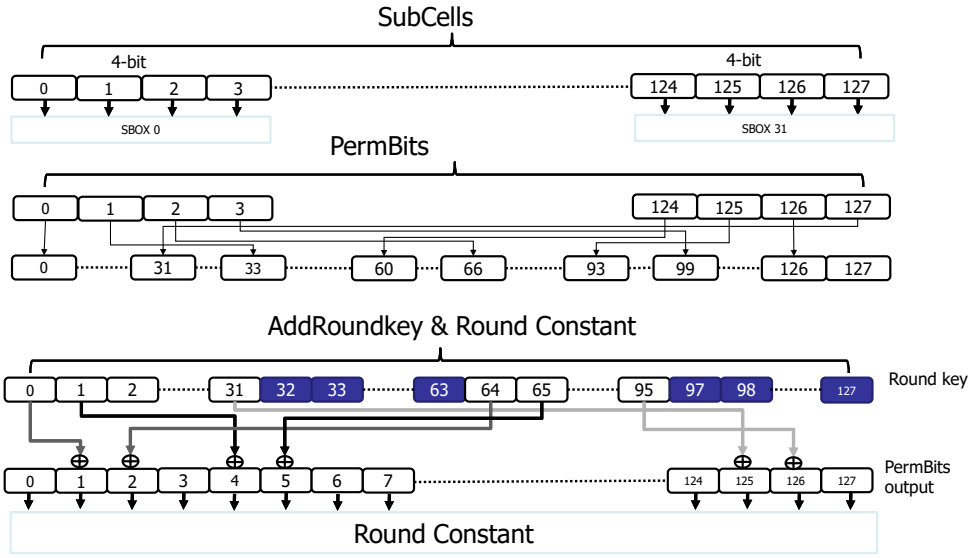


Figure 2.2: Round Operations of GIFT-128 Cipher

AddRoundKey: This function processes its input state in a 4-bit segment based fashion. Only the middle two bits of each segment are XORed with specific bits of the key as illustrated in Figure 2.2. Note that in each round only 64 bits of the key are used. For round 0 these are bits 0-31 and 64-95 as indicated in the figure. A key scheduler is used to update the round key.

Round Constant: After the XOR operation with the key, a selected number of bits (i.e., 3,7,11, 15,19, 23, and 127) are XORed with the round constant. The two least significant bits (LSB) of each segment are XORed (i.e., exclusive-or operation) with specific bits of the key. Figure 2.2 shows that the two LSB bits of the first segment are XORed with key-bit 0 and key-bit 16 (see purple blocks in the Sub-Key Adding part of the figure). For the following segment, the key-bit 1 and key-bit 17 are used. This interleaved AddRoundKey process uses 32 bits of the key per round. Additionally, in this step, the most significant bit of each segment (MSB) is XORed with a specific round constant (see yellow blocks in the Constant Adding part).

Key Schedule After each round, the key is updated based on two steps as illustrated in Figure 2.3. In the first step, the key is circularly rotated by 32 bits to the right. In the second step, the last eight segments (i.e., bits 64-95) are updated as follows (see also the bottom part of the figure): (1) the first four segments are reversed, i.e., bits 64-67 are moved to bits 76-79, bits 68-71 to bits 72-75, bits 72-75 to 68-71, and bits 76-79 to 64-67; (2) the last four segments (i.e. bits 80-95) are updated by circular shifting a segment to the left by two and XORing it with the next segment that is circularly shifted by two to the right (see Figure 4.15c). Note that the last segment (i.e., bits 91 to 95) uses the segment with bits 80-83 as its neighbour.

This step updates the key for the next round. First, the entire key is 32-bits circularly rotated to the right, thus replacing the used key-bits by the next 32 bits. Thereafter, the

32 used key bits (now located in the most significant part of the key) go through a local circular rotate operations, where the 2 MSB bytes are rotated by 2, and the following 2 bytes by 12, as shown in the bottom part of Fig. 2.2.

2.2. SIDE CHANNEL ATTACKS

Power based side channel attacks are attacks where a malicious adversary takes advantage of the power consumption to deduce secret information. These attacks can be classified in non-profiled and profiled attacks as shown in Figure 2.4. Each class is briefly explained next.

2.2.1. NON-PROFILED ATTACKS TECHNIQUES

In these attacks, an attacker gets access to a target electronic device that runs a cryptographic algorithm. Thereafter, the attacker tries to perform a key recovery by correlating a leakage model with obtained power traces during the execution of the cryptographic algorithm. Famous examples of these types of attacks are Simple Power Attack(SPA) [9], Differential Power Attack (DPA) [9], Correlation Power Attack (CPA) [47], Collision Power Attack [48], Zero Value Attack [49], and Machine learning Attack [50]. Each one of these attacks will be explained briefly next.

SIMPLE POWER ATTACK (SPA)

An SPA attack can be carried out by merely observing changes in power usage throughout the execution of the target operation (e.g., RSA encryption). It's worth noting that in this attack no particular mathematical computations are required. The attack on the unprotected RSA implementation based on the multiply-square algorithm is a well-known example [9]. Observing the peak power values of the square and multiply operations during encryption and decryption allows the attacker to retrieve the key [9].

DIFFERENTIAL POWER ATTACK (DPA)

In DPA attacks [9], the attacker selects a small portion of the key (i.e., 8-bits for AES), divides the traces in two sets for 256 hypothetical key values based on a single bit at the output of the SBOX and selects the key belonging to the two sets where the mean difference between them is the highest. This process is repeated until the full key is recovered.

CORRELATION POWER ATTACK (CPA)

Correlation power side channel attacks [47] work as follows. The attack on AES starts similarly as DPA, but instead of creating two sets based on single bit at the output of the SBOX, the used key is estimated using the Pearson coefficient correlation, which is computed using Equation 2.3. In the equation, $h_{k,i}$ represents the hamming weight/distance of the i^{th} intermediate operation (e.g., SBOX in AES, square and multiply in RSA), k the subkey value of the encryption/decryption execution, $t_{k,j}$ the sample point j within the sub-trace k, and n the number of traces. In asymmetric algorithms, the key is recovered

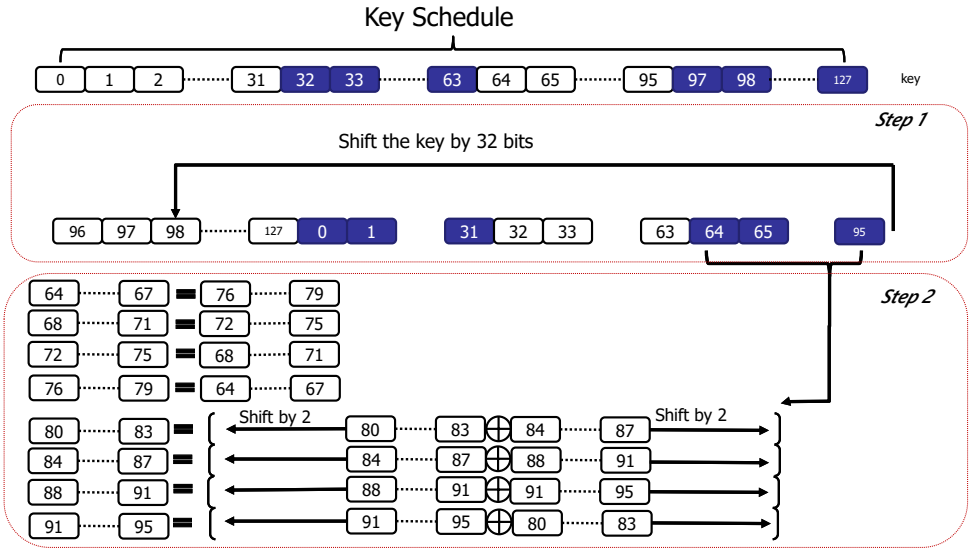


Figure 2.3: Round Key Update Process

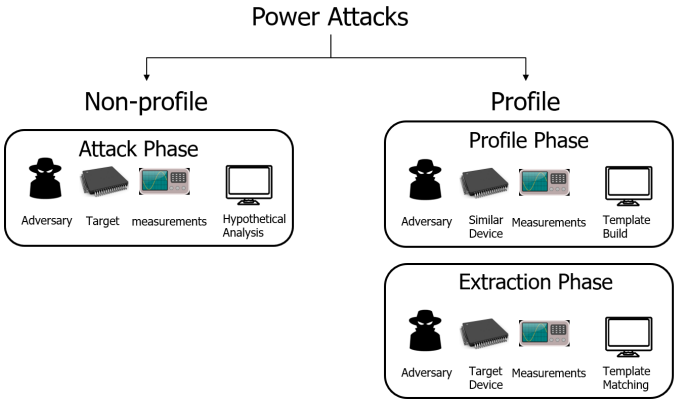


Figure 2.4: Classification of Power Attacks

in a bit-by-bit, in contrast to symmetric algorithms where this is determined by the width of the SBOX output (which equals 8 for AES).

$$r_{i,j} = \frac{\sum^{n}(h_{k,i} - \mu_{h_i})\dot{(}t_{k,j} - \mu_{t_j})}{\sqrt{\sum^{n}(h_{k,i} - \mu_{h_i})^2 \dot{\sum}^{k}(t_{k,j} - \mu_{t_j})^2}} \tag{2.3}$$

COLLISION POWER ATTACK

Collision attacks [48] aim at situations where two encryptions with different inputs and an unknown key will produce the same intermediate values (e.g. hamming weight/distance).

If an adversary can identify from the power consumption two encryption operations that contain this occurrence, the collision can be exploited. Since a collision only exists for a subset of the potential key space, each successful collision allows the attacker to narrow the key search space.

ZERO VALUE ATTACK

Zero value attacks [49] take advantage of the fact that known plaintext (e.g. setting it to zero) will lead to the leakage of information that exposes the secret key. These type of attacks mainly target implementations that contain countermeasures and aim to remove the randomization generated by those countermeasures. An example of such an attack can be found in multiplicative masking where zero input values cannot be randomized using multiplicative mask [51].

MACHINE LEARNING ATTACK

Machine learning attacks [50] use the leakage model to distinguish between traces and derive the key from them. These attacks mainly target asymmetric algorithms such as RSA and ECC. For example, k-means is a clustering algorithm [50] that is commonly used to apply such attacks. Starting with an initial guess/prediction, it splits the training set into k distinct clusters. For each collected trace, it iteratively detects the nearest cluster center (centroid) and updates the centroids based on the mean of all training instances assigned to it until no changes occur anymore. To put it another way, the aim is to discover a partitioning that minimizes the total cluster variance. To determine the distance between two traces, the squared Euclidean distance can be used. Once the clusters are created, the partial traces that belong to the two clusters represent either a square or multiplication operation. Once these operations have been defined, the key can be recovered in a bit-by-bit fashion [50].

2.2.2. PROFILED ATTACKS

In contrast to non-profiled attacks, in a profiled attack an adversary uses a similar or identical device under his control to create a leaking template known as the **profiling phase**. After that, the attacker correlates the power traces of the target device and compares them with the template to recover the key; this phase is also known as **extraction phase**. Both phases are explained next.

- PROFILING PHASE

  In this step, the adversary uses a similar or identical device that he or she completely controls to develop a behavioral model of the targeted device. This phase consists of the following steps:

  Step 1: In this step, the adversary looks for a device that behaves in a similar way as the target device.

  Step 2: In this step, the adversary selects and defines the point of attack (e.g., the output of *SubByte* operation in AES algorithm or *square* and *multiply* functions in RSA).

Step 3: In this step, the adversary measures several power traces of the chosen target point of attack.

Step 4: In this step, the adversary assigns a label to each trace acquired in Step 3. Depending on the cryptographic methodology employed, the label can be computed in a variety of ways. The hamming weight/distance of the SBOX output is the most commonly used label in AES algorithm [52]. Asymmetric algorithms, on the other hand, often use the main operations as labels, e.g., *square* and *multiply* in RSA [53].

Step 5: Finally, The adversary designs/builds a template to characterize the traces. The model is build from the traces and labels collected in Steps 3 and 4, respectively.

- EXTRACTION PHASE

During this phase, the adversary attempts to extract the secret information from the target device by applying the steps below:

Step 1: in this step, the adversary locates the device of attack.

Step 2: in this step, the adversary identifies the point of attack, i.e., the operation used to extract the labels during the profiling phase. For instance, the output of SBOX function in AES.

Step 3: in this step, the adversary measures several power traces that contain the point of attack. This step requires the traces to be sliced when asymmetric algorithms are used (e.g., slicing *square* and *multiply* operations in RSA). Slicing is not required for symmetric algorithms; there each power trace is represented with a single label.

Step 4: in this step, the adversary guesses the label value of each measured trace of Step 3. For AES algorithm, the labels can be seen as the results of the hamming weight/distance, while for asymmetric algorithm the labels represent the executed function (e,g., *square* and *multiply* in RSA).

Step 5: Finally, the adversary derives the secret key from the obtained labels. The key is retrieved in a bit-by-bit fashion when asymmetric algorithms from the identified operations of the previous step. The returned bits must be concatenated from left to right or right to left, depending on the methodology employed to recover the whole key. Symmetric algorithms need an additional steps, as the leakage model results (e.g., hamming weight) must be converted to a sub-key value. This additional step is depicted in Algorithm 4. The subkey is obtained after calculating the likelihood of key values. To predict a $subkey$ value, the likelihood of all potential subkey scenarios from $subkey = 0$ to 255 are evaluated by computing the leakage model results for each plaintext/ciphertext.

There are many examples of profiled attacks both for symmetric and asymmetric algorithms. In this section we selected the most famous ones (i.e., template

Algorithm 3 Symetric Algorithms: Key Extraction

```
1: procedure KEY_EXTRACT(Prediction_set, pt_array)
2:     P_k[0,255] = key probability
3:     Prediction = the results of the trained model on the attack traces
4:     pt = is the plaintext used in the encryption process.
5:     for each sub-key do
6:         P_k[0,255] = 0
7:         for j in trace-set do
8:             X_{0,255} = predict(trace)
9:             for k=0 to 255 do
10:                HW_k = HW(SBOX[pt[j] ⊕ k])
11:                P_k[k] = P_k[k] + log(Prediction_j[HW_k])
12:            end for
13:        end for
14:        guess_subkey = max(P_k)
15:    end for
16: end procedure
```

based attacks (TBA) [52], machine learning Attacks(ML-SCA) [54], and deep learning based side channel attacks (DL-SCA) [52]). Note that there are many variations proposed. Next each of them will be briefly described.

TEMPLATE-BASED ATTACK

In this attack, the *multivariate normal distribution* is used to create a profile. The profile consists of multiple covariance matrices C and mean vectors m of the points of interest of the collected power traces. First, the measured traces are grouped based on their Hamming weight/distance (HW/HD) value. Next the covariance and mean are computed for selected samples (i.e., the points of interest) within the traces for every HW/HD group. They are identified by C_h m_h for HW/HD with value h.

During the attack phase, the adversary uses the probability distance to correlate measured power traces with the profile. This is shown in Equation 2.4. In the equation, h denotes the template number (i.e., the corresponding HW/HD set) and t an attack trace. The value of the leaking model is determined by the template that produces the highest results. Note that the traces used for attack must be aligned with the traces used for profiling.

$$f(t) = \frac{1}{\sqrt{(2\pi)^n \times det(C_h)}} \times exp(-\frac{1}{2} \times (t - m_h)' \times C_h^{-1} \times (t - m_h)) \quad (2.4)$$

MACHINE LEARNING ATTACK

In machine learning (ML) attacks [54], the multivariate normal distribution is replaced by ML techniques such as Support Vector Machine (SVM). SVM is a binary classifier. First a feature selection method is used to reduce the dimension (i.e., trace length) of

the power trace. Thereafter a classifier is used to learn the features. During the profiling phase, the classifier creates two hyperplanes in a high-dimensional space with the goal of classifying the data. The data separation takes place in such a way that the hyperplanes are furthest away from each other. During the extraction phase, the classifier is used to classify the attack traces based on the distance to both planes. The percentage of correct classifications among the power traces from the test sets are used to determine the success rates. Note that SVMs are designed to perform a binary classification and hence can be used in three ways to perform an attack on symmetric algorithms [55]: (1) separate the results of hamming weight/distance to two groups (i.e., less than or greater than 4), (2) separate the results of the hamming weight/distance based on even or odd, and (3) separate the results of hamming weight/distance based on the value of the fourth least significant bit.

DEEP LEARNING

During the profiling phase, the attacker builds and trains a neural network. The attacker must first specify the neural network's structural parameters (such as depth, width, and activation function). After that, training is performed on traces that have labels attached to them. The attacker separates the dataset (i.e., traces and their labels) into a training set (usually 80 percent to 90 percent of the total dataset) and a validation set. The attacker ends the training when the training and validation accuracy is high enough. In extraction phase, the attacker applies the traces collected from the target device to the trained neural network. The results obtained from the neural network are subsequently used to extract the key.

2.3. SIDE CHANNEL COUNTERMEASURES

Several countermeasures to power attacks have been suggested over the last two decades. As shown in Figure 2.5, these countermeasures can be classified based on two metrics: technique (i.e,. obfuscation and balancing) and implementation level (i.e., software, hardware architecture, circuit/implementation, and technology). Next, the different countermeasures will be discussed based on their technique.

2.3.1. OBFUSCATION

Countermeasures based on obfuscation attempt to randomize the power behaviour irrespective of the performed operation. There are many examples of such techniques at different implementation levels available in the literature. At software level, one of the famous examples of obfuscating the power consumption for mainly symmetric algorithms is using masking [56]. Masking works by splitting the algorithm calculations' sensitive intermediate operations into $d+1$ random shares in such a way that analyzing d shares reveals no information about the secret value. Other examples of software level obfuscating are random order execution [57], random delay insertion [58], message and/or exponent blinding [53] and SBOX confusion [19]. In [57], random instructions with random register accesses are inserted between the original instructions sequence of the encryption/decryption process, which changes the power behaviour

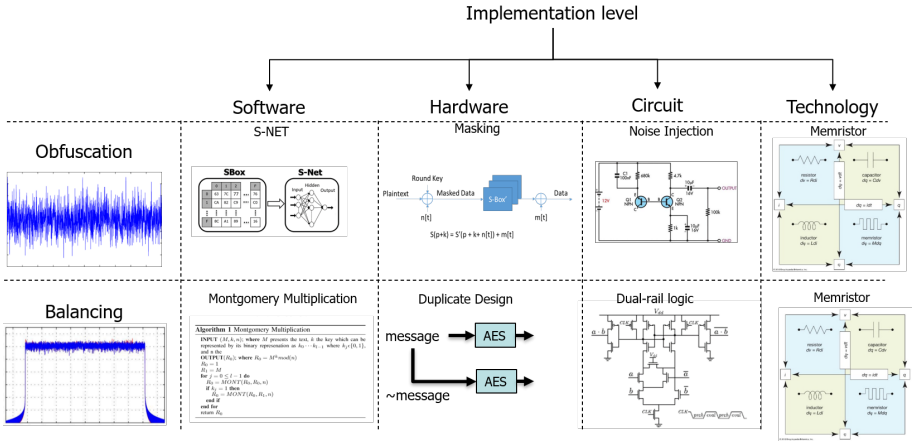


Figure 2.5: Classification of Countermeasures

each time. In [58], the power behaviour is altered by inserting random NOP instructions which causes misalignment in the power traces. In [53], the key and/or the message of asymmetric algorithms are randomized in each execution. In [19], the SBOX of AES algorithm is implemented using a neural network which confuses the power behaviour of the leakage model. Note that this countermeasure, unlike the others, targets the leakage model. In hardware, similar to software, masking [59] is the most popular countermeasure. Another example of a hardware based countermeasure is random delay insertion [60], where the delay is inserted by logic gates. Note that the other software level techniques can be also implemented in hardware. At circuit level, the power consumption can be obfuscated by modifying the logic cells as is the case for masked dual-rail pre-charged logic [61] or by having an additional source in the system to injected noise [62]. At technology level, emerging devices such as memristors can be used for obfuscation by exploiting cycle-to-cycle variation.

2.3.2. BALANCING

The goal of balancing techniques is to keep the power usage as stable as possible during sensitive operations. Similarly to obfuscation techniques, there are many examples of such techniques studied in the literature at every implementation level (i.e., software, hardware architecture, circuit, and technology). One of the famous countermeasures in software is Montgomery multiplication [63] where both operations of asymmetric algorithms (e.g., square and multiply in RSA and double and addition in ECC) are executed in the Montgomery domain. This results in a similar power behaviour for both operations. Hence, it is harder for an attacker to distinguish between them. Another example of a balancing technique at software level for asymmetric algorithms is multi-bit blinding [36]. This technique always executes the same sequence of operations regardless of the key bit values, by considering two bits at a time and re-order their operations. For symmetric algorithms, a multi-core can be used [64] where two encryptions are executed

on different cores simultaneously. One with original message while the other one with its complementary message. At the hardware level, the same techniques used in software can be implemented. A clear example can be seen in the duplicate design [65], where instead of having two software encryptions, two actual hardware implementations are used to run the message and its complementary at the same time. At circuit level many techniques were proposed such as *power equalizer* [66], *dual-rail logic* [67], and *Adiabatic Logic* [68]. In *power equalizer* [66], the power is balanced using the on-chip power supply. In *dual-rail logic* [67], the power is balanced by redesigning logic cells such that they take both the input and their complement values as inputs. In *adiabatic logic* [68], the power is balanced by designing CMOS cells in such a way tha they both charge and discharge at the same time to disguise power irregularities. At technology level, researchers are exploring emerging technologies such as Memristor [69] to minimize the power leakage, which increases the attack difficulty. Note that circuit and technology level techniques can be applied to both symmetric and asymmetric algorithms.

2.4. LEAKAGE ASSESSMENT STYLES

There are currently several options to evaluate countermeasure implementations. They can be grouped into three categories based on their style: evaluation-style, conformance-style, and formal style. Each style is briefly explained next.

2.4.1. EVALUATION STYLE

In evaluation-style testing, power traces are tested using actual side channel attack scenarios, such as those described in Section 2.2. They show whether the implementations are resistant to such attacks or not. The attacks can be performed in a profiled or unprofiled manner as discussed in Section 2.2. The attacks can be performed after the chip is manufactured using off-the-shelf security tools and equipment (e.g., equipment of Rambus [70] and Riscure [71]) or during the design process using CAD-based solutions [72].

2.4.2. CONFORMANCE STYLE

On the other hand, conformance-style testing examines whether traces are compliant with specific leakage criteria without taking actual attacks into account. Test Vector Leakage Assessment (TVLA) [73] and signal-to-noise ratio (SNR) analysis [74] are two examples of this form of analysis. Due to space limitations, we focus only on TVLA.

TVLA is based on Welch's t-test, which examines whether two populations have similar distribution. Welch's t-test is used in to identify whether power traces of an encryption/decryption algorithm execution leak information about the secret key. The leakage is measured using two sets of power traces, one with fixed plaintext/ciphertext and the other with random plaintext/ciphertext. Note that the key value is the same in both sets. Equation 2.5 shows the equation used to perform this test. In the equation, $\bar{X}_1$, S_1^2, and N_1 represents the mean, the variance, and the total number of used *fixed plaintext/ciphertext* traces, respectively, while $\bar{X}_2$, S_2^2, and N_2 represents the mean, the variance, and the total number of used *random plaintext/ciphertext* traces, respectively.

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{S_1^2}{N_1} + \frac{S_2^2}{N_2}}} \tag{2.5}$$

2.4.3. FORMAL STYLE

The aim of formal verification-based testing is to analyze the leakage of an implementation mathematically. Formal verification examples can be found in [75, 76]. In [75] the authors use formal verification to verify hardware masking countermeasures. In [76], the authors present a satisfiability modulo theories (SMT) solver to evaluate software masking countermeasures.

3

SIDE CHANNELS ANALYSIS

Depending on the leakage channels that are used, side channel attacks can take a variety of different forms. Variations in time, power consumption, and electromagnetic emissions are just a few examples. For the intent of this chapter, we narrow our focus to three specific channels: power, time, and thermal. These channels have been chosen with the objective of formulating assessment methodologies for a single channel. Section 3.1 introduces and investigates side channels in general and provides justifications for the chosen channels under study. Section 3.2 provides a detailed examination of attacks that exploit power consumption. Section 3.3 presents an analysis of time-based side channel attacks. Section 3.4 focuses on thermal-based side channel attacks. Finally, Section 3.5 provides an explanation why power side channel is chosen for leakage assessment.

This chapter is partially published on [22] [30] [34].

3.1. INTRODUCTION

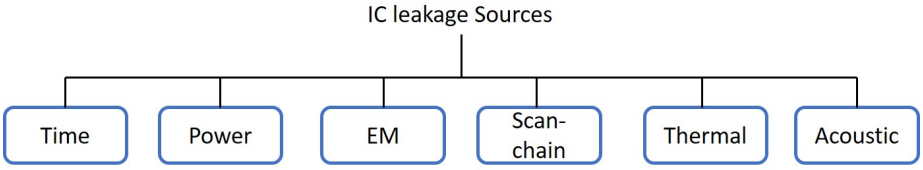


Figure 3.1: Leakages Sources of Side Channel Attack

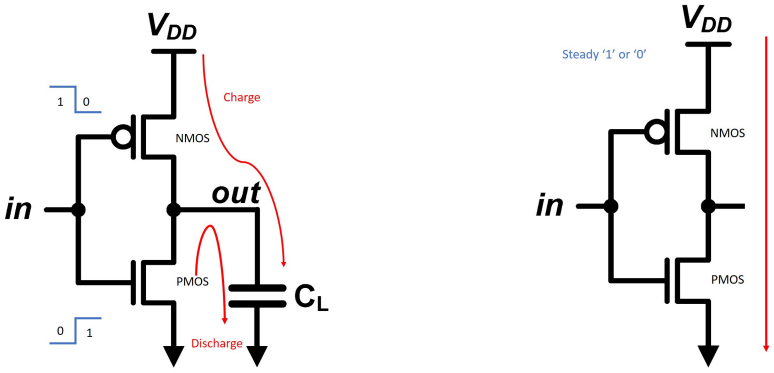
Microelectronic systems, particularly those employed in secure computing applications, are vulnerable to a range of leakage channels that have the potential to inadvertently disclose confidential data. These are:

- **Variations in Time** [18]: Differences in the amount of time it takes to complete a task can be examined and used to deduce information such as cryptographic keys.
- **Power Consumption** [9]: Different operations typically call for a different amount of electrical power. Attackers are able to determine what operations are being conducted and perhaps what data is being processed by monitoring these variances, which is a technique known as power analysis.
- **Electromagnetic(EM) Radiation** [77]: While they are functioning, electronic equipment give off electromagnetic radiation. It is feasible to deduce the actions being performed and even the data that is being processed by capturing and analyzing this electromagnetic radiation.
- **Scan-chain** [78]: Attackers are able to manipulate scan chains, which are generally used for testing reasons in chips, to either read or modify the internal state of a chip. Scan chains are utilized in chips.
- **Thermal** [79]: The amount of heat that electronic components are able to dissipate might change depending on how they are used. Keeping an eye on this thermal output can provide useful information about the computations and the data.
- **Acoustic** [80]: The sound or vibration created by electronic components can be utilized to infer the kind of processes that are being carried out, despite the fact that these sounds and vibrations are frequently quite slight.

  This study primarily examined three specific types of leakage, as we believe these particular forms of leakage pose the highest potential hazards. The leakages that have been chosen for analysis are power consumption, temporal variations, and thermal attacks. Power consumptions offer access to the behavior and execution patterns of all of the algorithm's operations, which makes it much simpler to zero in on those that are most vulnerable to attack. Similar leakages exist, such as electromagnetic emissions; nevertheless, the power consumption exhibits significantly lower levels of noise. Time variations have limitations in terms of evaluating all operations as they require waiting for a response. However, they do have

the advantage of providing easy remote access, as they can be calculated from the software side. Thermal, despite the fact that it is a very noisy leakage channel, is monitoring practically every device to prevent it from overheating. This monitoring can also be accessed remotely by software. As a result, it requires cautious investigation for possible forms of attack.

3.2. POWER BASED ATTACKS



(a) Dynamic Power (b) Static Power

Figure 3.2: Power Consumption Of CMOS Inverter Gate

In microelectronic circuit, power is consumed either when the transistor is changing its logical state or when the power is used to charge the load capacitance as can be seen in Figure 3.2. The total dynamic power can be calculate using the Equation 3.1, where C_L is the load capacitance, C is the internal capacitance of the chip, f is the frequency of operation, and N is the number of bits that are switching.

$$P_{dynamic} = P_{capicatance} + P_{transient} = (C_L + C) \times V_{dd}^2 \times f \times N^3 \tag{3.1}$$

A power analysis attack is a kind of side-channel attack in which an adversary attempts to compromise a system by recording and analyzing the patterns of power consumption that the system exhibits (see Figure 3.3). When a piece of hardware such as a microprocessor or cryptographic hardware performs operations, the amount of power that it consumes can change depending on the data that it is processing as well as the activities that it is carrying out at the same time. Attackers can acquire insights into the nature of the calculations that are being performed by precisely measuring the oscillations in the amount of power that is being consumed. This could potentially allow them to extract sensitive information such as cryptographic keys. This paper provides a comparison analysis of several pre-procesing techniques, including three pre-processing techniques which have not been studied yet for DL-based SCAs. Hence, five different methods are explored: i) Data augmentation [23, 24], ii) data transformation [25, 26], iii) data concatenation [27], iv) stacked auto-encoder [28]; and v) stacked auto-encoder with en-

coder only [29]. Note that data augmentation and stacked auto-encoder are already applied in the literature. Data transformation has been explored in some power attacks like CPA [25] and MLP-based [26], but not yet in DL-based attacks. The other two techniques come from the image processing field due to their outstanding results. To our best knowledge, this three methods are for the first time applied in DL-based SCAs. The main contributions of this work are:

- Proposal of data transformation using wavelet transform to improve DL-based SCAs.
- Proposal of data concatenation by augmenting the original trace with its fast Fourier transformation (FFT) to improve DL-based SCAs.
- Proposal of stacked auto-encoder using the encoder only to improve DL-based SCAs.
- Comparison of the proposed techniques with the state-of-the-art for both symmetric and asymmetric ciphers.

This section is organized as follows. Subsection 3.2.1 provides a state-of-the-art regards power side channels. Subsections 3.2.2 and 3.2.3 explain the deep learning attack model. Subsections 3.2.4 and 3.2.5 describes the pre-processing techniques. Subsection 3.2.6 provides the experiments setup. Subsections 3.2.7 and 3.2.8 discuss the results.

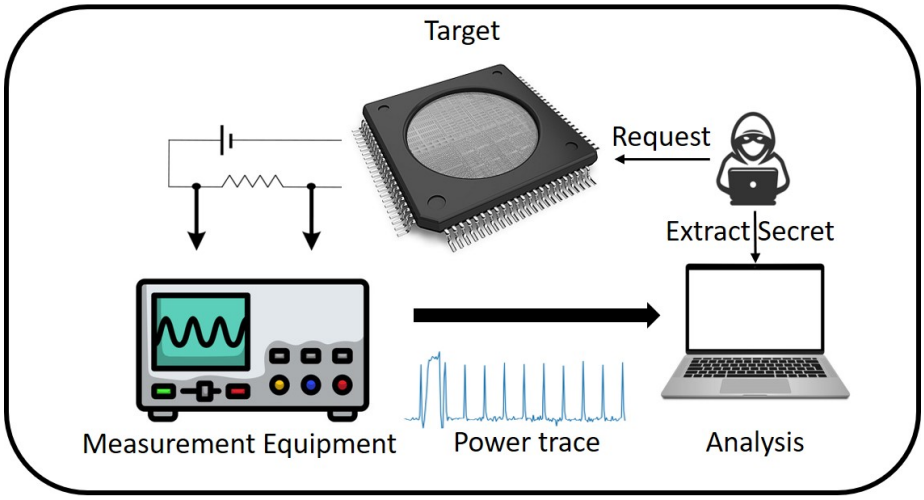


Figure 3.3: Power Side Channel Attack

3.2.1. STATE OF THE ART

Paul Kocher [9] introduced the concept of Simple Power Attack (SPA) and Differential Power Attack (DPA). These methods observe the relation of the performed operation in

the power traces. In a further work, a statistical method was proposed to correlate hypothetical power estimations with the real consumption. This attack is called Correlation Power Attack (CPA) [47]. In CPA, leakage models like hamming weight and hamming distance are used to correlate guessed keys with the power traces. Both models can be used to estimate the power behavior of a specific operation, for instance, the result of the first round of the popular AES cipher. On the other hand, countermeasures have been proposed to make DPA and CPA attacks less effective [81–83]. In 2002, the concept of profiled attacks was proposed by Chari et al. [52]. Their attack is referred to as Template Based Attack (TBA). TBA builds a customized power model of a device similar to the target device, and correlates the power measurements of the target device of the victim with the customized power model. TBA uses the multivariate gaussian distribution function to build the power profile. However, its mathematical complexity limits the attack accuracy [38]. To overcome this drawback, Martinasek et al. [84] proposed in 2013 the usage of machine learning to enhance the profiling attacks on AES. They used a multilayer perceptron (MLP) neural network. Initially, they obtained an 80% accuracy only, but after further optimizations reached a 100% accuracy [23] using a pre-processing technique consisting of averaging of power traces. Thereafter, Gilmore et al. [85] presented a profiled attack using MLP model that focused on symmetric cryptographic algorithms. In 2016, Maghrebi et al. [38] proposed the usage of Deep Learning in profiled Side-Channel Attacks. They successfully attacked different symmetric encryption implementations using a Convolution Neural Network (CNN). DL-based SCA improved the amount of traces required to accomplish the attack on both protected and unprotected symmetric cryptographic algorithms. In 2017, Cagli et al. [24] used data augmentation (i.e., generating new input samples to the neural network from the existing power traces) to further improve the accuracy of DL-based SCAs. They proposed two strategies to perform data augmentation, one by using random re-alignment of existing traces and the other by adding random noise. Note that profiled based side-channel attacks are less powerful when countermeasures like power obfuscation and randomized keys are used [86]. Some of the above deep learning based attacks used pre-processing techniques to enhance the accuracy. However, a systematic approach that compares the pre-processing techniques for both symmetric and asymmetric algorithms is missing, and hence it is not clear under which circumstances they are useful.

3.2.2. DEEP LEARNING BASED POWER ATTACKS

DL-based SCAs follow the same steps used in the conventional profiled-based attacks [52], i.e., they consist of a **profiling** and **extraction** phase [38]. Both phases are explained next.

PROFILING PHASE

In this phase, the attacker creates a behavioral model of the selected target device using a similar or identical device that he/she fully controls. This phase works as follows:

Step 1: In this step, the attacker searches for a sample device that is similar to the target device.

Step 2: The attacker chooses and locates an intermediate point of attack (e.g., *SubByte* operation in AES or exponent operations in RSA).

Step 3: The attacker records multiple power traces of the selected target operation.
Step 4: The attacker designs a deep learning neural network and trains it to characterize the traces. To be able to train the deep learning neural network, the attacker needs to associate each collected trace during Step 3 with a label. The label can be calculated differently based on the used cryptographic algorithm. For example, the common label for AES is the hamming weight of the *SubByte* operation's output [38], while RSA typically uses the main operations as labels, namely *square* and *multiply* [87].
Step 5: Finally, the attacker constructs a neural network and trains it. The attacker first needs to define the structural parameters (e.g., depth, width, activation function) of the neural network. Thereafter, training is performed on the traces collected during Step 3 with the associated labels calculated in Step 4. To train the neural network, the attacker divides the dataset (i.e., traces and their labels) into a training set (normally 80% to 90% of the complete dataset) and a validation set. The training and validation phases are completed when the attacker achieves an acceptable accuracy level.

EXTRACTION PHASE

The attacker aims to recover the secret information from the target device during this phase using the following steps:
Step 1: The attacker identifies the target device. This device has to be similar to the profiled one.
Step 2: The attacker locates the intermediate operation, i.e., the operation used to train the neural network during the profiling phase. For instance, the *SubByte* of AES algorithm.
Step 3: The attacker generates a new set of traces on the target device for the intermediate operation. Note that in asymmetric algorithms, traces are divided based on their main operations (i.e., *square* and *multiply* for RSA), while such a partitioning is not needed for symmetric algorithms.
Step 4: The attacker predicts the key of the newly generated traces using the previously trained neural network. The result of this step is the label of the intermediate operation. This label is a binary value for the RSA algorithm or the leakage model value (i.e., hamming weight) for AES algorithm.
Step 5: Finally, the attacker reveals the secret key information. In asymmetric algorithms, the key is recovered bit by bit based on the predicted operations. In order to recover the full key, the retrieved bits have to be concatenated either from left to right or right to left based on the algorithm used. Symmetric algorithms require more steps, as the predicted leakage model value has to be converted from the hamming weight to a sub-key value. This extra task is shown in Algorithm 4. After calculating the probability of the target traces and knowing the plaintext/ciphertext used in the algorithm encryption/decryption process, the subkey is retrieved next. To guess a $subkey$ value, for every plaintext/ciphertext in pt array we loop over all possible subkey scenario $subkey = 0$ to 255 and calculate the leakage model results of that subkey. Based on the output of the leakage model, we select the corresponding probability from the prediction array and add it to the probability array. The key probability results are accumulated for each element in the pt and $Prediction$ arrays. The subkey with highest probability is selected as subkey. The previous process is repeated for each subkey.

Algorithm 4 Symmetric Extract Key bytes

procedure KEY_EXTRACT($Prediction_{set}, pt_{array}$)
  $P_k[0,255]$ = key probability
  $Prediction$ = the results of the trained model on the attack traces
  pt = is the plaintext used in the encryption process.
  for each sub-key **do**
    $P_k[0,255] = 0$
    for j in trace-set **do**
      $X_{0,255} = predict(trace)$
      for k=0 to 255 **do**
        $HW_k = HW(SBOX[pt[j] \oplus k])$
        $P_k[k] = P_k[k] + log(Prediction_j[HW_k])$
      end for
    end for
    $guess_{subkey} = max(P_k)$
  end for
end procedure

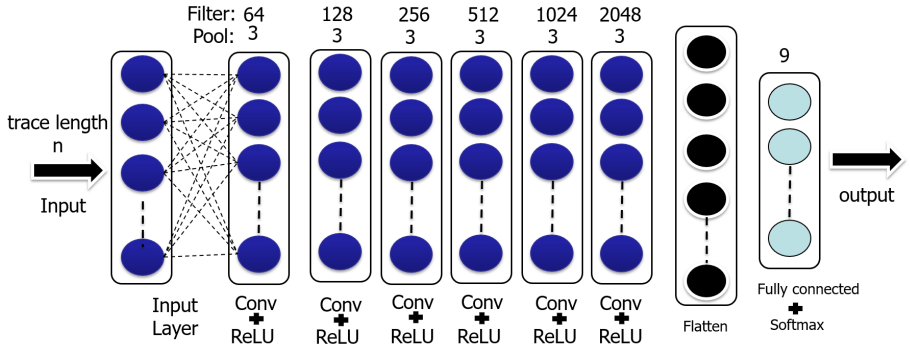


Figure 3.4: Baseline CNN with its hyper-parameters

3.2.3. BASELINE CNN

The baseline CNN is the reference neural network used for comparison when the pre-processing techniques are applied. The baseline itself does not use any pre-processing technique, which means that its inputs are the raw data values obtained from the collected power traces. As observed in Figure 3.4, the baseline CNN consists of 9 layers in total. The number of neurons in the first layer matches the trace length. Thereafter, it contains six convolutional layers, ReLU activation functions, and pooling layers. Note that for simplicity that several layers are presented together in the figure. Each of these layers contains a filter whose size is depicted on the top of the respective layer. For example, the first layer presents 64 filters. In the sequence, the pooling layer reduces the data width by a factor of 3. Note that the minimum layer width is equal to or greater than one. As the data goes through the network, more filters are added. Although the 7^{th} layer in

the figure comprises much more filters than the first layer, its width is much less due to the pooling layers involved in the process. The last part of the CNN consists of a flatten and a Softmax layer. The flatten layer converts the tensor representation (i.e., multi-dimensional matrix) of the data used in the previous layers to a single dimension vector representation. The Softmax layer is the final layer of the neural network and consists of 9 output neurons used to distinguish between the 9 classes for AES. These 9 classes represent the prediction probabilities of different hamming weights or hamming distances. A similar neural network is constructed for RSA that contains 2 output neurons.

The training parameters are shown in Table 3.1. Glorot [88] is the first parameter. Glorot is a sophisticated technique (as compared to e.g., random initialization) that initializes weights based on the width of its preceding and successive layers. Thereafter, the loss function is defined by the categorical entropy technique to compute the error function. The third parameter applies Adam optimization [89], which is a special technique that uses adaptive learning rates for each parameter and typically gives good results. Lastly, dropout is used to regularize the neural network; a dropout ratio of 50% is selected. Note that other dropout ratio values may be used. However, 50% typically provides good results [90].

3.2.4. TRADITIONAL PRE-PROCESSING TECHNIQUES

Three traditional pre-processing techniques are described in this subsection. They are data augmentation, data transformation and data concatenation.

DATA AUGMENTATION

This technique was already proposed by the authors in [23, 24]. It uses the average of traces to improve the extraction of the most meaningful features. All traces belonging to the same group, i.e., with the same hamming weight, are averaged to remove noise and misalignment between them and hence a cleaner version can be obtained. The traces that contain the average values for each hamming weight are replicated in such a way that they form half of the total number of traces for each hamming weight. Subsequently, random noise is added to those clean traces. Finally, the baseline CNN is trained using both the original traces (50% of the total traces) and the newly generated traces (the other 50% of the total traces). Note that the newly generated traces have the same label classifier (i.e. the same hamming weight) as the traces they were based on. The training phase is stopped when the accuracy of the model does not increase anymore.

DATA TRANSFORMATION

This technique aims to provide the CNN with a different data representation of the training set [91]. Our method applies a wavelet transformation for each power trace. The neural network is subsequently trained with only the wavelet samples with its associated labels calculated from their original counterparts. The model is trained until the accuracy saturates.

Table 3.1: Training Related Hyper-parameters Used for Classification

Training hyper-parameters	Baseline CNN	SdAE + CNN	SdAE v2 + CNN
Initialization	Glorot	Glorot	Glorot
Loss Function	categorical	binary and categorical	binary and categorical
Optimization	Adam	Adam	Adam
Regularization	Dropout (0.5)	Dropout (0.5)	Dropout (0.5)

DATA CONCATENATION

This technique aims to expand the data provided to the CNN [27]. By mixing different data representations of the same training set, improvements in prediction accuracy are expected. Our technique combines the FFT representation and the original trace to be represented as a single input. The model is trained until no further improvements are observed with respect to the accuracy.

3.2.5. HYBRID NEURAL NETWORKS

In this subsection, two hybrid neural networks are presented. Each contains the baseline CNN preceded with another neural network; they are referred to as the SdAE + CNN and SdAE v2 + CNN. SdAE refers to the Stacked Denoising Autoencoder [92], while the SdAE v2 to the same SdAE but with the encoder part only. The Stacked Denoising Autoencoder consists of an encoder and decoder and one of its main purpose is to filter input samples to enhance the quality. For example, SdAE has been used successfully to solve problems related to image colorization and noise reduction [92]. Another benefit of SdAE is that the encoding results in a lower dimension data, i.e., a compressed format of the input sample with a smaller number of points. On the other hand, the decoding part reconstruct the information to its original dimensions. This method has two main benefits. First, the encoding part compresses the data and extracts the most meaningful points in the trace. Note that the decoder decompress the compressed features to the full trace. Second, the SdAE can constructed in such a way that it restores missing parts or remove noise of traces. Next, we describe both hybrid networks in more detail.

SDAE + CNN

: In this strategy, the goal is to remove noise and misalignment of the input samples by training the SdAE. Two steps are required to achieve such results. First, the data-set is divided into groups (i.e., classes) based on their hamming weight. Second, the average of the traces is taken for the all input samples within the same group. The assumption here is that these average traces are much cleaner. Subsequently, the SdAE is trained with the normal input samples, but the error at the output is calculated using the average trace values. Once the training is completed, the result of the SdAE is classified using the CNN model. The overall training procedure of this approach consists of the following two phases:

Phase 1 - Training the SdAE model: In this phase, the SdAE is trained to reconstruct

the power trace without noise and misalignment issues. This is achieved by using the data (i.e., recorded power traces) as input data-set to the train model, while using the average trace to calculate the loss function. After this training step reaches an acceptable accuracy, the next phase can start.

Phase 2 - Training the CNN model: In this phase, the output of the trained SdAE is connected as an input to the CNN model. Subsequently, the training mode of the weights and biases is switched off for the SdAE part of the hybrid network. Finally, the hybrid model (SdAE + CNN) is trained (i.e., only the CNN part) using the recorded power traces as input, while their hamming weights are used as the labels to calculate the error. The training stops when the accuracy saturates.

SDAE V2 + CNN:

This approach aims at exploring the usefulness of the encoded version of SdAE; note that this encoded version holds the most important characteristics of the input trace. In this phase, the pre-processing based on trace averaging is not needed. Instead, the SdAE is trained based on the original data. After the training is completed, only the encoder part of the SdAE is connected to the CNN as shown in Figure 3.6. The training process is also performed using two phases:

Phase 1 - Training the SdAE model: In this phase, the SdAE is trained similarly to the previous approach. However, the input data (i.e., the power traces) is used to compute the error at the output during training. After the training reaches an acceptable accuracy, the encoder part is removed from the SdAE.

Phase 2 - Training the CNN model: In this phase, the trained encoder part is connected to the input layer of the CNN. Subsequently, the training for the SdAE encoder is switched off, i.e., their weights and biases are fixed to make sure that they are not adjusted during the training process of the hybrid model. After that, the hybrid model is trained (i.e., only the CNN part is being updated).

The structural related hyper-parameters of both approaches SdAE + CNN and SdAE v2 + CNN are illustrated in Figure 3.5 and Figure 3.6, respectively. Note that the CNN in Figure 3.5 is exactly the same as the one used in Figure 3.4, as the output layer in the SdAE has the same width as its input layer. The CNN in the hybrid neural network of Figure 3.6 is based on the same concept as the one in Figure 3.4, but its input width is smaller as the encoder of the SdAE compressed the data. More properties of both hybrid neural networks can be found in Table 3.1.

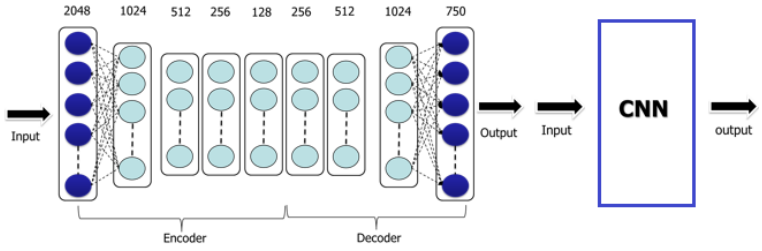


Figure 3.5: SdAE + CNN and their hyper-parameters

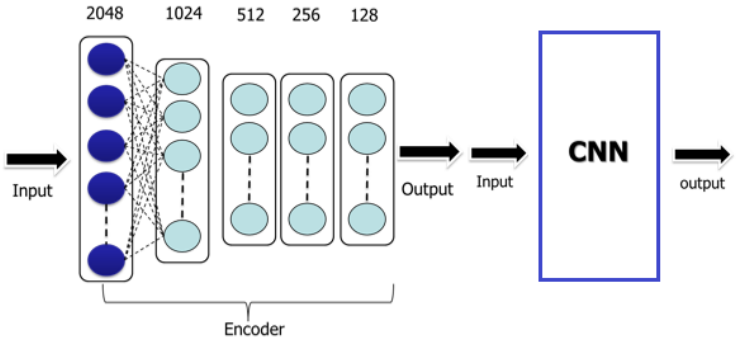


Figure 3.6: SdAE v2 + CNN and their hyper-parameters

3.2.6. EXPERIMENTAL SETUP

The experiments are conducted in two parts. In the first part, we evaluate the pre-processing techniques based on the mathematical approaches described in Subsection 3.2.4. In the second part, we test the accuracy of the hybrid neural networks described in Subsection 3.2.5. Both parts are evaluated using four different power traces; two are from AES cryptographic algorithms and two from RSA cryptographic algorithm. Their key characteristics are summarized in Table 3.2 and explained next:

1. **DPA Contest V2 [93]**: The traces in this training set are based on AES using a 128-bit key size. These traces are provided by DPA Contest V2; an open source framework that allows developers to compare their implementation attacks using a common benchmark. The traces represent a hardware implementation of the decryption process. In these traces, no countermeasures against side channel attacks have been used. Each trace consists of 3250 data points.

2. **ChipWhisperer 1 [94]:** Similar to the previous data set, the traces here are also based on a non-protected AES encryption implementation with a 128-bit key size. However, here a software implementation is used instead. The data has been recorded using ChiphWhisperer tool which is an open-source tool for side-channel power analysis and glitching attacks. ChipWhisperer-Lite kit-board is used to measure the traces. The length of each recorded trace equals 3000 points.

3. **ChipWhisperer 2:** Unlike the first two types of traces, the traces in this training set are based on an asymmetric algorithm. Here traces are used of a 512-bit key software implementation of the asymmetric cryptographic algorithm RSA. The traces are collected on the same platform used for the ChipWhisperer 1 data set. The trace length is 3000 points.

4. **Pinata [95]:** A similar RSA software implementation has been used for this data set as ChipWhisperer 2. The difference is that the Pinata board from Riscure [96] is used for collecting the traces. Each trace contains 8000 sample points.

Table 3.2: Utilized Power Traces and Their Characteristics

Traces	Crypto (key size)	Platform	Trace length
DPA Contest V2	Unprotected AES (128)	Hardware	3250 points
ChipWhisperer 1	Unprotected AES (128)	Software	3000 points
ChipWhisperer 2	Protected RSA (512)	Software	3000 points
Pinata	Protected RSA (512)	Software	8000 points

The ChipWhisperer data sets are measured in a controlled environment and hence provides cleaner traces. The other two data sets have been gathered from an uncontrolled environment which might come with certain restrictions.

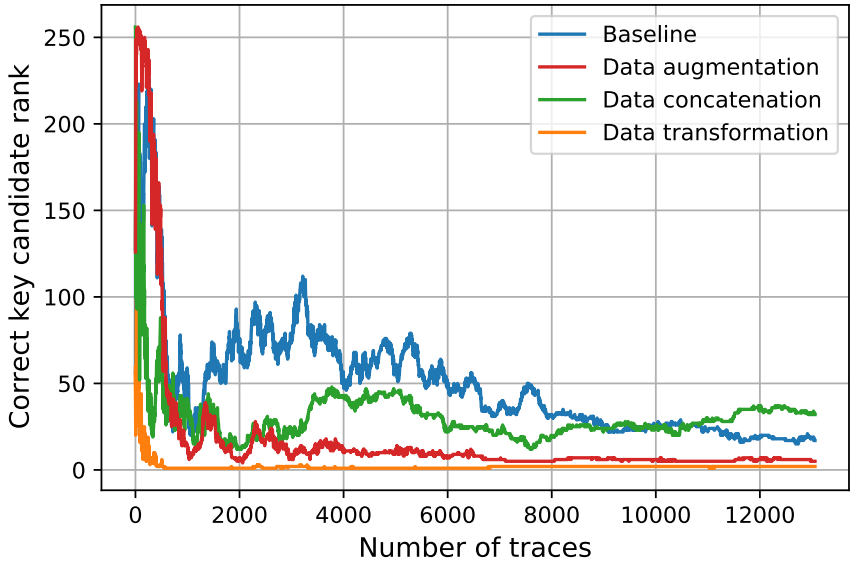


Figure 3.7: AES key rank analysis for pre-processing techniques

3.2.7. RESULTS OF TRADITIONAL PRE-PROCESSING TECHNIQUES

Table 3.3 provides the accuracy analysis results of the traditional based pre-processing techniques. The techniques are applied on both symmetric and asymmetric data sets.

For the symmetric data sets, the table shows the training accuracy, validation accuracy, and the maximum rank. The rank specifies how close the guessed key is to the correct key. A key rank of 0 means that the correct key has been guessed, while a key rank of 255 means that the correct key has the lowest guess probability. The lower the key rank,

Table 3.3: Pre-processing techniques comparison and results.

Evaluated Technique	Step	AES Evaluation (15 byte)		RSA Evaluation	
		DPA Contest v2	CW 1	Pinata	CW 2
Baseline CNN	Training	33.7%	98.62%	99.9%	100%
	Validation	23.5%	80%	75%	100%
	Final	Rank 17	Rank 0	80%	100%
Data Augmentation	Training	41.14%	98.78%	98.9%	100%
	Validation	27.5%	60%	75%	100%
	Final	Rank 5	Rank 0	70%	100%
Data Transformation	Training	40.17%	99.3%	99%	100%
	Validation	26.02%	85%	95%	100%
	Final	Rank 2	Rank 0	94%	100%
Data Concatenation	Training	30.1%	98.12%	33%	100%
	Validation	25.7%	70%	34%	100%
	Final	Rank 32	Rank 0	33%	100%
SdAE	Training	37%	90%	99.8%	100%
	Validation	29.5%	80%	95%	100%
	Final	Rank 80	Rank 0	90.1%	100%
SdAE v2	Training	28.1%	90%	98.6%	100%
	Validation	24.6%	70%	94.3%	100%
	Final	Rank 58	Rank 0	92.3%	100%

the more successful the attack is. From the table it can be observed that for DPA contest V2 the pre-processing techniques significantly improve the results. The data transformation technique (i.e., wavelet transformation) does not only improve the training and validation accuracy but more importantly, reduces the key rank from 17 to 2. The data augmentation also improved the neural network accuracy but ended up in a slightly higher rank (i.e., 5). The third pre-processing technique, i.e., data concatenation, actually worsened the results.The rank analysis results of the four techniques on DPA contest V2 data set are shown in Figure 3.7. The figure shows clearly that the data transformation has a strong positive impact on the accuracy of the attack. Note that it was difficult to see the effect of the pre-processing techniques on the ChipWhisperer 1 data-set, as the results of the baseline was already good.

For the asymmetric data set, instead of the key rank, the percentage of the key recovery is used. We refer to it as success rates. The results are similar to the symmetric data-set; the data transformation technique showed a significant improvement as compared to the baseline results, i.e., a success rate improvement from 75% to 95% for the Pinata traces. Here, the data augmentation has a marginal impact on the results, and the data concatenation technique again impacted the results negatively. Note that again no differences have been observed for the ChipWhisperer 2 traces.

3.2.8. RESULTS OF HYBRID NEURAL NETWORKS PRE-PROCESSING TECHNIQUES

Table 3.3 also provides the results analysis of the SdAE and SdAE v2 pre-processing techniques. Similarly to the traditional techniques, the results of both AES and RSA for the chipWhisperer platform did not provide distinguishable results as the traces were already clean to start with. However, for the DPA contest V2 and Pinata traces, both hybrid networks were able to improve the validation accuracy. Despite this improvement, the rank analysis shows that these techniques did not improve the attack. In case the hyperparameters of these hybrid neural networks are changed they will most likely perform better.

3.3. TIME BASED ATTACKS

A timing side-channel attack (T-SCA) is a specific type of side-channel attack that leverages the timing variations of a cryptographic device in order to extract confidential information. The timing attack, as depicted in Figure 3.8, is predicated upon the duration required for the execution of an algorithm. An illustration of this concept can be observed in the context of a password checker, where the verification process is conducted on a character-by-character basis. If the checking procedure terminates prematurely upon reaching the initial faulty character, the duration required can be significantly reduced in comparison to cases when the incorrect character is located further along the string. The variance in processing time can be leveraged by an attacker to deduce the accuracy of individual characters inside a password, hence reducing the number of possible password combinations. The seemingly slight variations in execution time can yield vital information to malevolent individuals, thereby undermining the security of password verification devices. T-SCAs have the capability to be deployed on various devices, encompassing smart cards, microcontrollers, FPGAs, and even cloud-based computing platforms. Time-based side channel attacks (T-SCAs) operate by quantifying the duration required to execute a cryptographic transaction. Subsequently, the assailant leverages the aforementioned data to illicitly obtain confidential information pertaining to the device, including the secret key. In this paper we propose GRINCH, the first cache attack on GIFT. Caches are usually shared memories that are used to speedup the execution of the cryptographic algorithms. However, they become a security threat when mutually accessed by multiple processes. A malicious process may gather information to reveal the secret key by: observing the execution time (time-driven attack) [97], exploiting the access pattern (access-driven attack) [32], or inferring the sequence of hits and misses (trace-driven attack) [33]. GRINCH crafts specific inputs to the cipher to extract sensitive data by observing its cache accesses. Hence, it is an access-driven cache attack. In summary, the contributions of the paper are:

- Analysis of GIFT vulnerabilities
- Implementation of the GRINCH attack
- Evaluation of the impact of the cache configuration on the attack efficiency

- Practical demonstration of the attack with two hardware platforms (SoC and MP-SoC) in an FPGA
- Proposal of two countermeasures to protect GIFT

The rest of the section is organized as follows. Subsection 3.3.1 provides state of the art regarding time side channel. Subsections 3.3.2, 3.3.3, 3.3.4, and 3.3.5 present the threat model and GRINCH attack. Subsection 3.3.7 presents the validation results. Finally, Subsection 3.3.8 provides potential countermeasures.

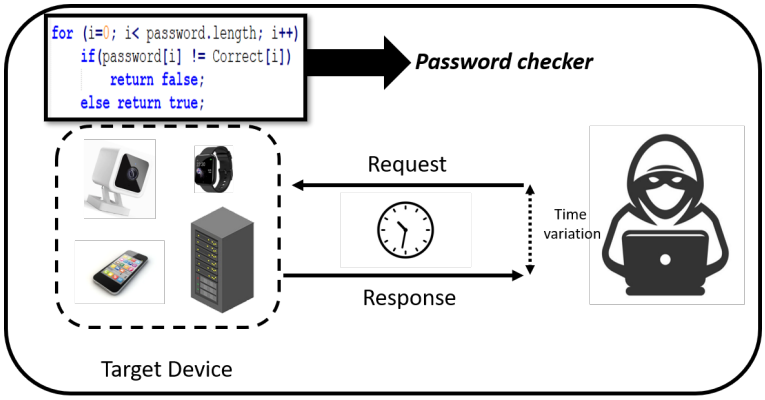


Figure 3.8: Time Side Channel Attack

3.3.1. STATE OF THE ART

The first recorded T-SCA was published in 1996 [18] by Paul Kocher. In his paper, Kocher described an attack against the RSA digital signature algorithm that exploited the timing variations of the algorithm to extract the secret exponent. Since Kocher's paper was published, there has been a significant amount of research on T-SCAs. In 1998 [98], J.F. Dhem demonstrated the feasibility of timing attacks by attacking a smartcard that stored a private RSA key. Schindler [99] also presented timing attacks on RSA implementations that use the Chinese Remainder Theorem (CRT). In recent years, researchers have developed timing attacks against a wider range of cryptographic algorithms and systems. For example, in 2003 [100], Brumley and Boneh demonstrated that timing attacks can be used to reveal RSA private keys from an OpenSSL-based web server over a local network. In 2005 [101], Acimez and Schindler proposed an efficient attack on RSA implementations that use CRT with Montgomery's multiplication algorithm. The importance of securing these environments became abundantly clear with the unveiling of microarchitectural vulnerabilities, notably the Spectre [102] and Meltdown [103] vulnerabilities in 2018, which highlighted the potential for timing attacks even on modern CPUs. Furthermore, in the past few years, the academic community has paid significant attention to machine learning-driven timing attacks. In 2017, researchers developed a deep learning-based timing attack against the AES encryption algorithm. The attack was shown to be effective against a variety of AES implementations, including those that were protected

with countermeasures. In 2020, researchers developed A Novel Timing Attack on ECC Cryptography using Deep Learning [104]. It is important to note that T-SCAs are a constantly evolving threat, and new attack techniques are being developed all the time. It is important for the designers and implementers of cryptographic systems to stay up-to-date on the latest T-SCA research and to take appropriate countermeasures.

3.3.2. CACHE VLUNERBILITY ON GIFT CIPHER

Similarly as in many SPN-based ciphers, GIFT includes a non-linear substitution operation (i.e., SubCells or S-Box) that substitutes each 4-bit segment with another 4-bit number. A commonly used software implementation of GIFT is based on transformation tables, where the *SubCells* operation is implemented with a lookup table whose entries are the input rounds. In such implementation, a single lookup table is used in such a way that it is accessed by each segment. Our attack focuses on the first four rounds by monitoring the key-dependant S-Box cache accesses.

The input of the S-Box (also called *index*) is the result of XORing the previous round state (round input) with the secret key. In the first round, the plaintext is used as the state and no key operation is involved. From the second round onwards, the *index* is computed from the previous state and the secret key. Therefore, when the round input and the index of the substitution table are known, it is possible to retrieve the key by calculating $Key \leftarrow Index \oplus Input$.

Fortunately, GIFT cipher uses an S-Box of 16 values, which is considered very tiny when compared with the 256 values S-Box used in the AES. As a result, the probability that an encryption uses all S-Box addresses in the first rounds is very high. Hence, an attacker that is looking for the used cache addresses after the end of the encryption process would not be able to extract useful information. Nevertheless, today's systems employ task scheduling, where tasks are pre-empted to run multiple tasks concurrently. Therefore, it is possible to access the cache while the cipher is still in its intermediate state.

GRINCH strategy is based on running multiple encryptions with different messages, each carefully crafted in order to activate the same *index* of the S-Box (in a certain segment of a certain round). The attacker can eliminate key candidates from the encryption based on the selected S-Box address until a single index remains, which the attacker subsequently can use to retrieve part of the key (see Subsection 2.1.3). Finally, the same process is repeated by targeting different segments until the full key is recovered.

3.3.3. THREAT MODEL

IoT devices contain System-on-Chips (SoCs) that include single or multiple heterogeneous processing units, memories, peripherals, hardware accelerators and other IP hardware cores [105]. SoCs may include memory hierarchies comprising several levels of cache (e.g., L1 to L3) and DRAMs. When a cache miss occurs, data is searched throughout the cache levels and eventually looked up in the DRAM when needed.

GIFT cipher is designed to be used in such IoT devices. They use an operating system to manage and schedule multiple applications. Note that trusted cryptographic applications (e.g., GIFT cipher) share the hardware platform together with potential malicious

or untrusted third-party software that could gather, process and communicate data. Taking all of this into consideration, we can assume that trusted and non-trusted applications can run on the same hardware platform, sharing resources like on-chip communication structures (e.g., bus or Networks-on-Chip), interfaces and cache memories. In this work, we define two main processes named victim and attacker. The victim process encrypts/decrypts messages using GIFT cipher. The attacker process runs external malware that manipulates the data to be encrypted and accesses the shared cache memory. In summary, the considered threat model has teh following characteristics:

- The cache used by the victim is accessible by the attacker.
- Attacker can measure the cache access time (to differentiate cache miss and hit).
- Attacker can create/manipulate plaintexts.
- Optionally, the attacker can flush the cache.

3.3.4. METHODOLOGY

The GRINCH attack focuses on identifying the index of one single S-Box access (i.e., one segment) of the second round. Note that in the second round the key is used for the first time. If such an index is identified, two bits of the key can be retrieved (see Figure 2.2). However, an attacker only can control the plaintext and not the state of the second round. Hence, to control a single access of the S-Box in the second round, the attacker has to carefully select four segments of the plaintext (i.e., input of first round). These four input segments determine the value of one segment of the second round due to the S-Box and permutation operation of the first round. As the key is unknown to the attacker, it is not possible to know upfront which S-Box index will be used. To solve that issue, an attacker can perform many encryptions while crafting the input in such a way that the targeted segment is kept active and stimulates the same target key bits (i.e., the two bits involved in the AddRoundKey). These plaintext manipulations create a condition where only one index will be accessed in the cache during all performed encryptions. When more encryptions are performed, more key candidates can be eliminated until a single candidate remains. In such cases, it is possible to reverse engineer the two bits of the key using the index and the state of the involved segment. This process is repeated 15 times for the other segments to recover the complete 32 bits of the key. Once the attacker knows 32 bits of the key, the process can be repeated in order to attack the next round by computing the input state of the following round. Note that the key is shifted 32 bits to the right after each round (see also Figure 2.2). By changing the plaintext to match the targeted segments in the third round again 32 key bits can be recovered. After applying the same trick four times, the entire 128-bit key can be retrieved.

The methodology of GRINCH consists of five steps as shown in Figure 3.9 and described next.

Step 1 - Generate Plaintext + Encrypt: The goal is of the Step 1 is to craft the plaintext so to force the same S-Box accesses for certain key-bits. The methodology starts by defining the target key-bits, as shown in Algorithm 12. The first part of the algorithm identifies the offset of the key-bits K_i, K_j in the AddRoundKey (e.g., the offset of key bit 0 is 0 and

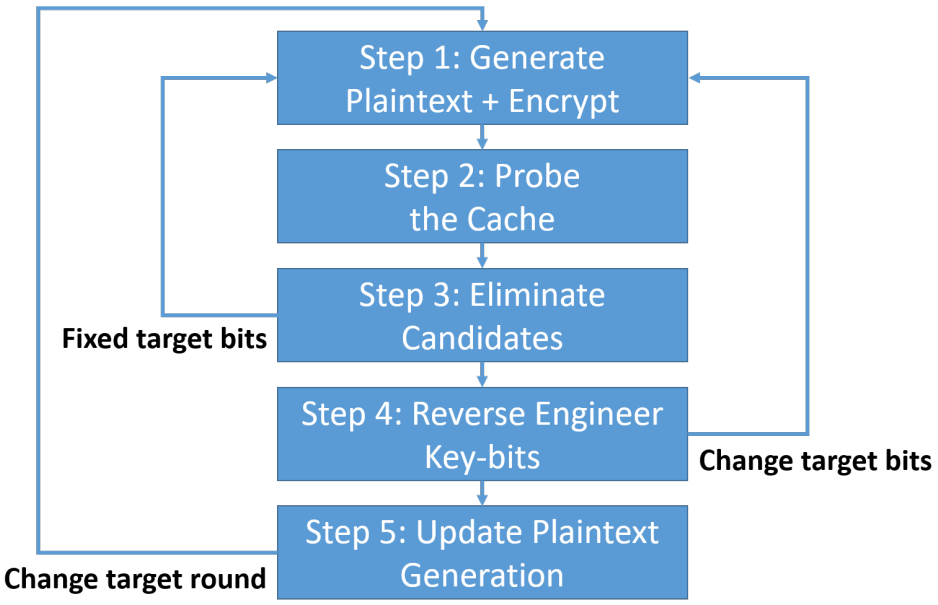


Figure 3.9: GRINCH attack methodology

the offset of key bit 16 is one (see AddRoundKey in Fig. 2.2. This is realized through the $StatusBitXorKey(K)$ function (lines 2 and 3). Thereafter, these bits are inversely permuted (lines 4 and 5) to obtain their indexes (bit_A) and (bit_B) before the PermBits function, which is equal to the output of the S-Box layer. For the attack to succeed, the bits bit_A and bit_B should always keep the same value so that the target key bits of the S-Box in the next round remain unchanged. In this attack we set these bits to 1. Hence, the inputs of the S-Boxes for these two bits must be careful chose and always lead to a 1 at the output (loop on lines 5-12). As a result, for each bit, a list of valid inputs and that will always force the same XOR operation with the target key-bits K_i, K_j is generated. Thereafter, all the plaintexts are generated based on these lists, as in Algorithm 6. For each plaintext segment, a random value is applied when it is not part of a segment where bit_A or bit_B is located, and an arbitrary index from the list is used when it is in order to make sure that the value at bit_A and bit_B is always 1 after the S-Box. At the end of this procedure, a single plaintext is generated and used for the attack.

Step 2 - Probe the Cache: To obtain the information of accessed addresses of the S-Box, the attacker can perform classical cache attacks such as Prime+Probe and Flush+Reload. The former method accesses an address of the cache that evicts the victim's information. During or after the victim's operation, the attacker accesses the same address and observe if it has been used. If the victim used that address, the attacker experiences a cache miss. Flush+Reload uses the same principle, but the first part is performed with a specific command to erase (parts of) the cache (i.e., flush operation). For the GRINCH attack, the Flush+Reload method is better choice. As a flush operation is faster, the attacker can probe the cache earlier. The earlier the attacker is able to probe the cache, the

Algorithm 5 Set target bits algorithm

```
1: procedure SET_TARGET_BITS(K_i, K_j)
2:     a ← StatusBitXorKey(K_i)
3:     b ← StatusBitXorKey(K_j)
4:     bit_A = Inv_Permutation(a)
5:     bit_B = Inv_Permutation(b)
6:     for each element X inside SBOX do
7:         if X[bit_A] == 1 then
8:             list_A.append(Inv_SBOX[X])
9:         end if
10:        if X[bit_B] == X then
11:            list_B.append(Inv_SBOX[X])
12:        end if
13:    end for
14:    return List_A, List_B
15: end procedure
```

Algorithm 6 Plaintext generation algorithm

```
1: procedure GENERATE(List_A, List_B)
2:     for i ← 0; i < 16; inc i do
3:         if i == segment(bit_A) then
4:             Plaintext[i] ← list_A[random()]
5:         else
6:             if i == segment(bit_B) then
7:                 Plaintext[i] ← list_B[random()]
8:             else
9:                 Plaintext[i] ← random()
10:            end if
11:        end if
12:    end for
13:    return Plaintext
14: end procedure
```

easier it is to monitor individual rounds.

Step 3 - Eliminate Candidates: The goal of the Step 3 is to identify the unique index that is accessed in many different encryptions. Since the target bits involved in the add round key are fixed, one of the S-Box indexes will be present in all performed encryptions. After some iterations, it is possible to identify the index related to the target key-bits by eliminating the indexes that do not appear in all cases.

Step 4 - Reverse Engineer Key-Bits: Since the attack methodology always forces the target key-bits to be XOR-ed with ones (i.e., bit_A=bit_B=1), the attacker can simply reverse engineer these key-bits by inverting the related bits of the obtained index. This can be expressed by $Key[i] \leftarrow Index[a] \oplus 1$ and $Key[j] \leftarrow Index[b] \oplus 1$; or $Key[i] \leftarrow \neg Index[a]$

and $Key[j] \leftarrow \neg Index[b]$.

Step 5 - Update Plaintext Generation: After attacking the first round, the attacker needs to repeat the process targeting the following rounds. The complete key can be retrieved after four iterations. However, each time the target round changes, the plaintext generation algorithm has to be updated. The revealed 32 key-bits (from Step 4) need to be used to generate new plaintexts that can be used to attack the next round, i.e., the attacker can compute the intermediate round values to generate the plaintexts that force the values on the target bits in the round under attack.

3.3.5. CHALLENGES

By analyzing the GRINCH attack methodology, the Step 2 (Probe the Cache) might be challenging due to the required timing precision in accesses the cache during the victim's operation, and due to the configuration of the cache memory. Some strategies to overcome such issues are discussed next.

Cache Probing Precision: Depending on the system configuration, the task to access the cache during the first rounds of GIFT cipher might be not feasible. Nevertheless, the attacker can still try other approaches. An interesting option is to apply power analysis to observe the cache accesses. The work in [106] has demonstrated that the power consumption may clearly reveal when cache misses and hits happens.

Cache Configuration: The configuration of the cache memory affects the attack. An important parameter is the size of the cache line. Since the GIFT S-Box only contains 16 bytes, a cache line could contain multiple elements. As a result, the accessed index is obfuscated. Nevertheless, the attack is still possible as long as the whole S-Box fits in a single cache line. Note that only the two least significant bits of the index are not controlled by the attacker, as they depend on the key-bits. This means that independently of the cache line size, the maximum number of candidates is 4. As a result of this, the attacker can continue to the next round and assume all possibilities.

3.3.6. EXPERIMENTAL SETUP

The GIFT software implementation was obtained from the public repository in [107]. It has the *SubCells* and *PermBits* operations implemented through look-up tables. GIFT was deployed into two SoC platforms: i) *single processor SoC*, comprised by a processor, a shared cache L1, I/O peripherals (i.e., UART serial) and a bus as communication structure; and ii) *multi-processor SoC (MPSoC)*, a tile-based structure comprising seven processors, a shared cache L1 and I/O peripherals. All these components are interconnected through a mesh-based Network-on-chip (NoC) that uses XY deterministic routing. Both SoC platforms use the RISCY core as the processing unit. RISCY is a RISC-V architecture from the Pulpino project [108]. The shared L1 cache used in both platforms is a 16-way set-associative memory with 1024 cache lines where each cache line contains in the default case a single word consisting of 8 bits. GRINCH was executed on both platforms while performing encryptions with GIFT. Three different experiments are performed in this work:

1. **Attack Effort versus Attack Efficiency:** This experiment analyzes the amount of encryptions that are required by GRINCH to perform a full recovery of the GIFT

key. This amount depends on the cache probing efficiency. We evaluate the impact of different probing moments and the impact of a flush operation during the attack. This scenario uses the cache line size of 1 word.

2. **Attack Effort versus Cache Configurations:** This experiment evaluates the impact of the cache line size on the attack. The effort is measured in terms of the amount of encryptions required to perform a full key recovery. Cache line sizes of 1, 2, 4 and 8 words per cache line are analyzed.

3. **Practical Attack Analysis:** This experiment runs the attack on the two hardware platforms (i.e., single processor SoC and MPSoC) on an FPGA. This evaluation provides practical attack efficiency results for different clock frequencies. For the single processor SoC, a task scheduler was used to emulate the RTOS operating system [109]. RTOS is a popular OS for embedded and IoT systems, which uses a *quantum time* (i.e., the timing slot that a process gets assigned to the processor) of 10 milliseconds.

For the first two experiments, RTL simulations were used to collect clean data. The third experiment, the attack was executed in an FPGA platform. The target FPGA is the Genesys 2 board which contains a Xilinx Kintex 7 device [110]. In all experiments, the pre/post-processing analysis (i.e., plaintext generation and reverse-engineering of the keys) were performed in Python.

3.3.7. RESULTS

The results of the three set of experiments are presented next.

ATTACK EFFORT VERSUS ATTACK EFFICIENCY

Figure 3.10 shows the required amount of encryptions to perform a full key recovery of the GIFT cipher when the first round is attacked; hence, only 32 bits of the key. The horizontal axis shows the moment in time (in rounds) in which the attacker can probe the cache status. The earlier this moment, the better the attacker's efficiency. In case the attacker is able to probe the cache in the first round, approximately 100 encryptions are needed to recover 32-bit keys (as can be seen in the figure). To recover the whole key, 400 encryptions would be required. The later you probe the cache, the more contaminated the results are. The efficiency of the attack depends on the amount of noise (e.g., multiple processes disputing the processor) and the operating system configuration (i.e., defined quantum time). Considering the first round attack (i.e., the first round can be probed), only the cache sets accesses of the second round contain useful information for the attacker. The cache sets accessed in the subsequent rounds are additional sources of noise, which is reflected in the results as extra effort. Additionally, as the S-Box lookup table is small, late cache probing results in that most likely all S-Box content is present in the cache, which makes it extremely hard for the attacker to eliminate candidates (see exponential increase in complexity vs cache probing time in the figure). Moreover, the experiment also evaluated the effect of the *flush* operation. The absence of the flush operation increases the attack effort since it adds noise (includes "dirty" accesses from the first round) to the information gathered by the attacker. Note

that the first round depends only on the input and it is not useful for the attacker. Hence, it only increases the effort to succeed. Our experiment does not evaluate the efficiency of rounds higher than 10, as the amount of encryption required for retrieving the key at round 10 is already too high to be considered practical in an IoT environment. Results show that the attack is practical if the adversary probes the cache before the fifth round when the *flush* operation is used, and before the forth round, otherwise.

ATTACK EFFORT VERSUS CACHE CONFIGURATIONS

Table 3.4 shows the attack efficiency for different cache configurations. Results show that the increase of the cache line size decreases significantly the efficiency of the attack; this is measured by the amount of encryptions required to perform a full key recovery. Note that the experiments with more than 1 million encryptions were drop-out before finishing as they are not considered practical. However, the attack is still practical when the attacker probes the cache within the first or second rounds. Therefore, the success of GRINCH depends both on the precision and ability of the attacker to probe the cache in the correct moment of time and on the cache memory configuration.

PRACTICAL ATTACK ANALYSIS

Table 3.5 shows the practical implementation of GRINCH. The results show the round number which was successfully probed by the GRINCH. For the single processor SoC, the GRINCH was able to probe the cache during the second round when operating at the lowest frequency (10 MHz). This result is interesting as many IoT devices are expected to operate at this frequency. In contrast, when the SoC is operating at higher frequencies, the GRINCH was only able to probe the cache at the fourth and eighth rounds for 25 MHz and 50 MHz, respectively. For the multi-processor SoC, the GRINCH was very efficient and probed the cache during the first round. Since the malware runs on its own dedicated processor, the attacker can write content to the shared cache as desired. As observed during experiments, in the fastest scenario (i.e., encryption running at 50 MHz), the time between different rounds was about 1.2 milliseconds. This time is much higher than accessing the shared memory on a different tile, which took approximately 400 nanoseconds consisting of the processor delay, Network-on-Chip latency and cache memory response time.

3.3.8. POTENTIAL COUNTERMEASURES

From the analysis of the GIFT cipher and the observed results it is possible to create two protection strategies. The first countermeasure is to eliminate the look-up table vulnerability. For the S-Box, the proposed method is to set the cache line to 8 bytes and reshape the S-Box from 16 rows of 4 bits to 8 rows of 8 bits. As an overhead, you have to select the right 4 bits at the output. The second countermeasure is to modify the *UpdateKey* operation of the GIFT cipher. Currently, the first four rounds uses directly the bits of the key, which makes GRINCH attack possible. If the UpdateKey of the first round prepare the sub-key to be used in the next round by applying some computation with bits that were not used yet, the key retrieval would not be possible. This requires however cryptanalysis that goes beyond the scope of this paper.

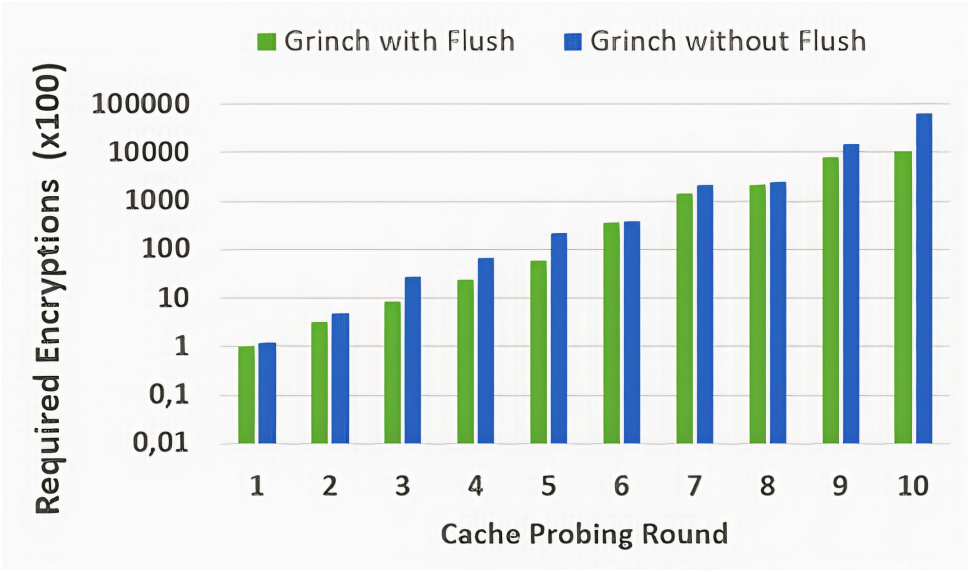


Figure 3.10: Required encryptions to break 1st GIFT Round.

Table 3.4: Required encryptions to attack the first round.

	Attack Efficiency - Probing Round				
Cache Line Size	**1**	**2**	**3**	**4**	**5**
1 Word	96	312	840	2,448	5,864
2 Words	136	1112	11440	188536	>1M
4 Words	136	123848	>1M	>1M	>1M
8 Words	113000	>1M	>1M	>1M	>1M

Table 3.5: Attack efficiency (rounds) of performed attacks.

	Clock Frequency		
Platform	**10 MHz**	**25 MHz**	**50 MHz**
Single-processing SoC	2	4	8
Multi-processing SoC	1	1	1

3.4. THERMAL BASED ATTACKS

A thermal side-channel assault (TSCA) is a form of side-channel attack that leverages the thermal disspations produced by electronic devices in order to retrieve confidential data, such as cryptographic keys (see Figure 3.11). Electronic devices produce thermal heat due to the power consumed by the circuit during the execution of an operation. The amount of heat produced is dependent upon several variables, including the device's technology, the magnitude of the task being executed, and the surrounding atmospheric temperature. TSCAs exploit the fact that different operations generate different amounts

of heat. As an illustration, the execution of a multiplication operation results in a greater amount of heat generation compared to the execution of an addition operation. The utilization of a thermal sensor by an attacker enables the quantification of the heat emitted by a device during the execution of a task of high sensitivity. Through the examination and interpretation of thermal data, the attacker possesses the capability to get pertinent insights into the ongoing operational activities.

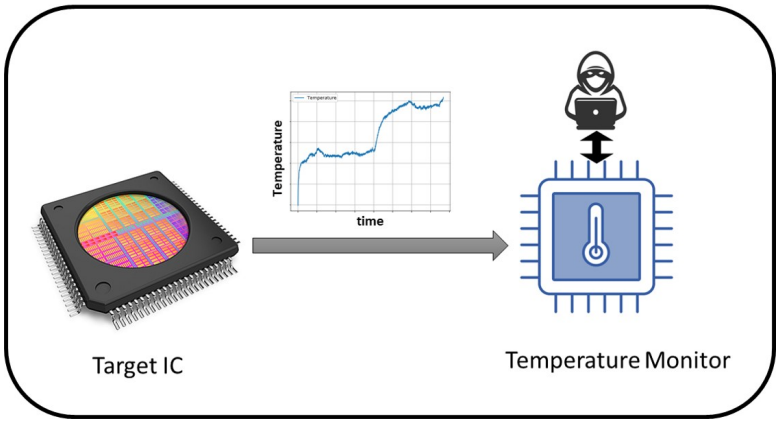


Figure 3.11: Thermal Side channel Attacks

In this study, we perform a comprehensive investigation on the practicality of performing thermal SCAs. We target asymmetric cryptographic algorithms as they typically are more computationally intensive than symmetric ones, which in theory means that they generate more heat. Since temperature has a slow response compared to power, intensive computational tasks provide better quality traces as shown in [35]. Hence, we show how known SCA techniques like Simple Power Analysis (SPA), Correlation Power Analysis (CPA) and Deep-Learning Power Analysis (DL-SCA) can be adapted for thermal attacks. Additionally, we propose a novel thermal attack by modifying the CPA attack; it achieves a successful and complete key recovery. We refer to this attack as Progressive Correlation Thermal Analysis (PCTA). All these attacks have been evaluated on unprotected and protected versions of an RSA software implementation. Finally, we present a comparison of all these SCAs techniques using both thermal and power leakage to clarify how powerful temperature based attacks can be. The main contributions of the paper can be summarized as follows:

- Evaluation and adaptation of simple power analysis (SPA), correlation power analysis (CPA) and deep learning based power analysis (DL-SCA) for thermal analysis.
- Proposal of a novel attack called progressive correlation thermal attack (PCTA) and its variant for power (PCPA) as a side channel.
- Evaluation and comparison of thermal side channel attacks on unprotected and protected RSA crypto-system implementations.
- Comparison between the attack accuracy of thermal and power attacks.

3.4.1. STATE OF THE ART

The authors in [111] used specific programs to heat the processor and encode information through the cooling fan behavior; no fan activity represents a binary zero, while the opposite a binary 1. The authors successfully demonstrate that the heat can be manipulated by an attacker and exploited externally by simply observing the system's behavior, i.e., the cooling system. In [35], the authors showed that processors inside a multi-core system can leak or exchange information through temperature side-channel. They focused in the paper mostly on a methodology to calibrate and perform inter-processor communication using temperature as a covert channel. They even showed that isolated cores are vulnerable to such a covert channel. In the last part of the paper, the authors also presented an attack to identify running applications on the processor by correlating thermal traces. They use a profiled based methodology, where the attacker has access to previous application traces. Despite their success, this attack is considered a coarse-grain side channel which is not as critical as a fine-grain side channel attack (e.g., the ones that are needed to target cryptographic keys). The authors mentioned that they were not able to perform a fine-grain SCA due to a low resolution and a low available sampling rate in most temperature sensors, and conclude that fine-grained attacks might not be practical. In [112], the authors also proposed a methodology to create covert channels through thermal leakage on multi-core systems. Their threat model considers a spy process inside an isolated core leaking sensitive data. One of their most interesting contributions is the design of a transfer function that models the thermal leakage behavior. The objective was to estimate the maximum data capacity of the channel; this defines an upper bound on the amount of data that can be leaked through the covert channel. As a result, the authors improved the transmission rate by a factor of 20 as compared to previous work, achieving a rate of 27 bps with an 11% error rate. In summary, all these papers have presented interesting contributions on how thermal leakage can be exploited to transmit hidden messages. Their successful results indicate that meaningful data can be leaked through such channels, which motivates the scientific community to do more research on thermal side channel attacks.

3.4.2. CHALLENGES OF THERMAL SCA

Similarly, as is the case for power attacks, the Hamming weight (HW) and Hamming distance (HD) can be used to model the thermal activity of a processor [113]. If an operand with a large HW value is used in a serial multiplier, it will produce more heat with respect to the case where the HW is low. Consequently, the temperature will rise. However, there are a few differences compared to the power consumption that cannot be modeled by HW or HD such as the accumulative effect of temperature over time. Therefore, a model closer to the physical behavior is needed to help us to understand the thermal leakage.

One way of modeling the physical behavior of the thermal leakage is by analyzing the system as an RC-network [114]. This network behaves like a low pass filter with a cut-off frequency somewhere in the kilo Hz range. This frequency response could pose a problem since most computers tend to run in the GHz range. With a low pass filter (even if it is first order), it is very hard to measure any useful data when a system is running at 800 MHz. Luckily, it is not required to capture every clock cycle for side channel analysis, as long as multiply and square operations can be differentiated. If these operations take

long enough, differences should be visible in the thermal traces.

Another issue with thermal leakage is that they have an integrative behavior, i.e., the temperature accumulates when a task is executed (e.g., RSA decryption). The contribution of thermal leakage from a previous operation is still present in its following operation. Hence, it is likely that the same operation (e.g. a multiplication) at a different time moment will have a different starting temperature. To overcome this, two approaches are possible. In the first approach, pauses can be inserted between the operations. This can be achieved by periodically stopping the clock or by introducing pauses after each operation (e.g., periodically forcing an interruption in the processor). The second approach consists of rapidly cooling the system after each operation. This can be realized by adding external cooling such as a high-speed fan.

The last issue thermal traces suffer from is the variation of temperature offset with time. While the power is regulated by a voltage controller, the temperature of the environment and the processor are not directly controlled. As a result, the offset varies multiple times during execution. Today, most systems employ dynamic clock and voltage scaling to keep the temperature in a certain safe range [115]. Consequently, the thermal leakage also presents drifts in the offset during execution, but in a much regular behavior. Therefore, thermal side channel analysis requires a mechanism to filter out such drifts, as they most likely will not able to work on traces with varying offsets.

3.4.3. THREAT MODEL

Our threat model for thermal SCAs consists of the following assumptions:

- The attacker has direct access to the target device in order to record the thermal traces of the executed decryptions.
- The attacker has access to the ciphertext.
- The attacker can acquire a similar device. We target devices using off-the-shelf parts such as ARM, AVR, etc.
- The attacker has the ability to slow down the target operation (i.e., RSA encryption/decryption). It can be achieved by manipulating the external crystal, by forcing the target device to compute many tasks in parallel, or by provoking interruptions calls.

3.4.4. SIMPLE THERMAL ATTACK (STA)

Similarly as for SPA, STA aims to distinguish the operations by visually inspecting the thermal traces. In order to investigate how difficult it is to make visual inspections on thermal traces, we performed a small experiment. Under an ARM-based System-on-Chip inside the PYNQ-Z1 board [116], we run a test application and record its thermal traces. The temperature was measured by the internal analog-to-digital converter connected to an embedded temperature sensor which is integrated into the target board [116]. The application consisted of two loops without any code inside, as shown in Algorithm 7. After each loop, a pause was inserted. Such a pause makes the visual analysis better as both operations will show isolated behavior in the traces.

Algorithm 7 Simple Thermal Side Channel Analysis test.

```
for i := 0 ⇒ N do
    i++
end for
pause
for i := 0 ⇒ 2N do
    i++
end for
pause
```


The differences in temperature from executing different operations can clearly be identified (as seen in Figure 3.12). Figure 3.12a shows the thermal trace when no cooling is applied. As can be seen from the figure, after some time the temperature starts to increase. This impacts the quality of the traces negatively. In order to minimize this, a cooling fan was installed on top of the processor chip. Consequently, the overall temperature reduced by almost 10 degrees Celsius with no temperature drifts as shown in Figure 3.12b; this clearly improves the quality of the traces. In both figures, there are two distinguishable shapes visible, a shorter and a longer one corresponding to the different loop sizes. Another interesting observation is that the peak temperature differs slightly for both loops; the system reaches a temperature that is approximately 0.5 degrees higher when the longer loop is executed.

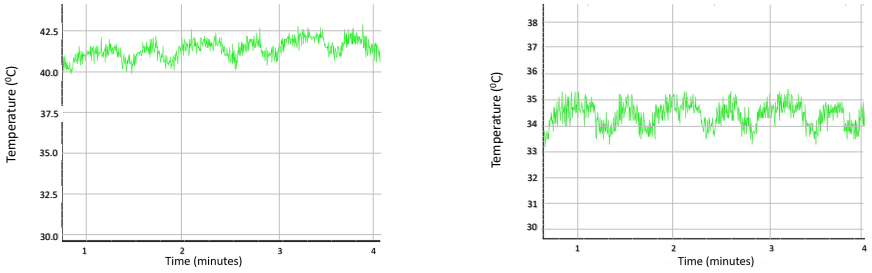
Methodology: STA can be described by the following sequence of steps:

1. Setup target device with a cooling system
2. Collect thermal traces when running the target operation
3. Find Points-of-Interest (optional)
4. Use visual inspection to retrieve the key

STA requires the same setup used in the test experiment, which uses a cooling system in the target device. STA focus on an unprotected RSA decryption, where periodic interruptions are placed between square and multiply operations when collecting the traces. Hence, the attacker can evaluate each operation independently using visual inspection. For example, Figure 3.13 shows the thermal trace of both operations. Note that the raw temperature on the y-axis can be transformed to degree Celsius using Equation 3.2. However, for simplicity and since the relation between them is linear, we used the raw data generated by the XADC [117] for our security analysis.

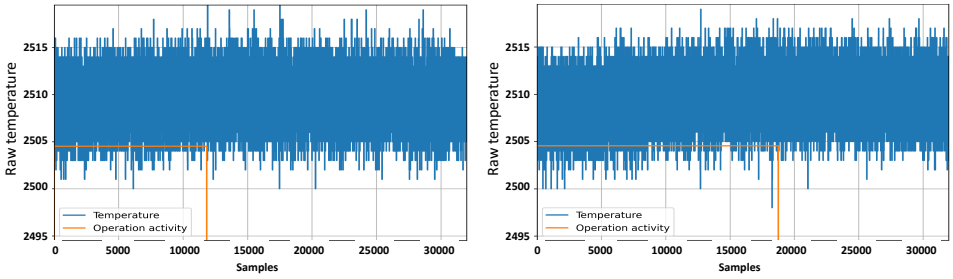
$$T = \frac{Raw\,Temperature \cdot 503.975}{4096} - 273.15 \tag{3.2}$$

As observed in the figure, each operation has a different behavior. However, visual inspection is very challenging since the differences can only be identified by some specific points instead of by the entire shape of the traces. Additionally, note that the temperature offset varies over time. This is known as the temperature drift effect, which is



(a) Without Cooling (b) With Cooling

Figure 3.12: Simple Thermal Trace Analysis.



(a) Multiply Operation (b) Square Operation

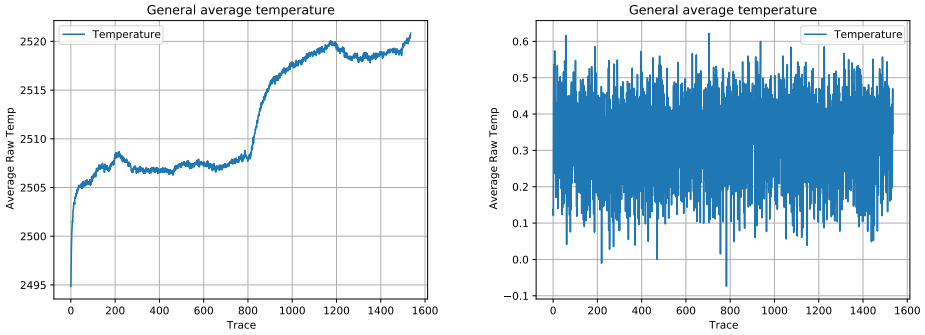
Figure 3.13: Comparison of square and multiply operations of RSA.

caused by the system regulating the voltage (and indirectly temperature) over time. For STA, temperature drift does not affect the analysis much. Therefore, STA is practical. Additionally, STA can be improved if some extra processing steps are used to identify the most important points that can be used to distinguish between the operations. This step is defined in our methodology as *Find the Point-of-Interest (POI)*, and several techniques can be applied such as the sum of pairwise differences or the sum of squared pairwise T-differences (SOST) [118].

3.4.5. CORRELATION THERMAL ATTACK

As shown in STA, the thermal traces contain a temperature drift. As a result, the thermal traces have to be pre-processed before Correlation Thermal Attack (CTA) can be applied. Figure 3.14 shows the temperature drifting behavior again but for multiple operations. Each point represents the average temperature of 1600 thermal traces.

One way of solving this temperature drift is by subtracting the average of each operation of each trace. This is a method used to remove noise [119]. However, this is not suitable here as the average of each trace is relevant for the correlation process with the Hamming weights (HW). Another way of removing the drift offset is by subtracting the first value of each operation in their partial traces. However, this can be tricky due to the presence of noise. To make this more robust, it is better to take the average of the first



(a) Without Temperature Drift Compensation (b) With Temperature Drift Compensation

Figure 3.14: Thermal Traces for Multiple Operations

m-samples and subtract that value from the trace. This reduces the effect of noise in calculations. This technique is based on the auto-zero amplifier [120] and can be observed in Equation 3.3. Instead of charging a capacitor in the sampling phase, the first values are accumulated and normalized.

$$T_{\text{filtered offset}} = \frac{1}{m}\sum_{i=0}^{m} t_i - T \tag{3.3}$$

Figure 3.14b shows the same information as 3.14a but compensated for the temperature drift. As can be seen, the drift is almost completely removed. To understand how this approach works, let's revisit Figure 3.13b. In the figure, the orange line indicates when the processor is active (i.e., approximately the first 18000 samples). The temperature offset only starts to rise somewhere between 10000 and 15000 samples (due to the integrator effect of the thermal behavior). This means that the first 10000 samples can be used to reduce the drift effect. In Figure 3.14b, the drifting effect was removed by subtracting the average of the first 12000 samples (i.e., m equals 12000 in this case). Note that the average of each trace is unequal to zero; therefore, they can be used for correlation.

Methodology: The Correlation Thermal Analysis can be described by the following sequence of steps:

1. Setup target device with a cooling system
2. Collect thermal traces when running the target operation
3. Remove temperature drift
4. Estimate Hamming weight of the target operations (i.e., square and multiply)
5. Correlate thermal trace with hypothetical guesses
6. Classify key bits as "1" or "0" according correlation results

After solving the temperature drift issue, the Hamming weight (HW) values have to be computed. Since there are only two possible operations (i.e., square or multiply), it makes sense to calculate the HW of the result of the square and multiply operations to use for the correlation process. Since it is not possible to compute all possible HW due to RSA key sizes, an estimation is derived from the average of a random simulation process. To make a proper estimation of these values, a simulator was created. After selecting ten different key pairs, which were generated randomly with the OpenSLL python library [121] (with key length 2048), the average HW of the results of the square operations was 922, while 461 for the multiply operations. Depending on the implementation, the actual HW might differ. However, as long as the ratio between square and multiply HW is around 2:1, the attack will work. After the Hamming weights are created for both scenarios (i.e., square and multiply) and the thermal traces are collected and processed, it is possible to calculate the correlation matrix r. Like CPA, we use Equation 2.3 to compute r. For each trace, there are two possible values defined as r_{square} and r_{multiply}. If $r_{\text{square}} \geq r_{\text{multiply}}$ the key-bit guess is 1, else the key-bit guess is 0. The pseudo code is provided in Algorithm 8.

Algorithm 8 Correlation Thermal Analysis for a Naive Implementation of RSA

```
for i := 0 ⇒ length(trace) do
    r_multiply ← correlation(h_multiply, trace[i])
    r_square ← correlation(h_square, trace[i])
    if r_square ≥ r_multiply then
        key_guess[i] ← 1
    else
        key_guess[i] ← 0
    end if
end for
```

3.4.6. DL-BASED THERMAL ATTACK

The goal of a side channel attack is to classify parts of a trace in such a way that they lead to the key. Machine learning (ML) and deep learning (DL) are very suitable methodologies to realize this. The most effective solutions use supervised learning (i.e., when the attacker has a similar device to train the network) and Convolutional Neural Networks (CNNs) [122]. CNN is a popular and effective way of (image) classification and recognition [123]. Although CNNs can be very complex, they have one advantage over the techniques like CPA/CTA as they can learn the leakage behavior.

A representative example of our target CNN is presented in Figure 3.15. The input is a 1-dimensional thermal trace followed by convolutional layers interleaved with pooling layers. Each convolutional layer contains a Rectified Linear Unit (ReLU) activation function [124], batch normalization and gaussian noise insertion. Batch normalization and gaussian noise are used to avoid overfitting. Our first attempts used drop-out; however, after many trials the combination of batch normalization and gaussian noise provided the best results (about 90% accuracy during the training phase). The last part of the CNN

consists of the classification, which contains flattened, fully connected and SoftMax layers. The SoftMax layer maps the output values between 0 and 1.
Methodology: The DL-based Thermal Attack can be described by the following sequence of steps:

1. Setup template device with a cooling system.
2. Collect thermal traces when running the target operation with known key.
3. Remove temperature drift.
4. Slice and label the traces.
5. Separate part of the traces into a training and validation set.
6. Train the CNN using the training set.
7. Validate the CNN using the validation set.
8. Setup target device with a cooling system.
9. Collect thermal traces when running target operation.
10. Remove temperature drift. (create evaluation set)
11. Apply the evaluation set to the trained CNN.
12. Collect more traces to perform a vertical attack

The attack starts by recording traces from the template device, where the input and key applied are known. Subsequently, pre-processing is applied to remove the temperature drift. For this purpose the same technique as described in Section 3.4.5 (see Equation 3.3) is used. Thereafter, the traces are divided into two sets, i.e., a training and validation set.

The next step in the process is to determine the labels to perform the training. In the unprotected RSA implementation, it makes sense to look at two possible label structures:

- Labels method A: *square* and *multiply*
- Labels method B: *square square*, *square multiply* and *multiply square*

Method A has as advantage that it is the most simple approach. It just requires having traces with a single square or multiply operation. In case a certain operation is wrongly predicted, it might be that two multiply operations follow each other up; this is however not possible and hence should be corrected. This is not possible in method B, which automatically eliminates/ignores incoherent results. However, method B requires a larger and more complex CNN. In this work, we used method A for labeling.

The next step is to train the neural network. The training parameters are the following:

- Initialization: Glorot [88] is used to initialize the weights and biases. Glorot is an advanced technique (as compared to e.g. random initialization), where the initialization values are computed based on the width of its preceding and successive layers.

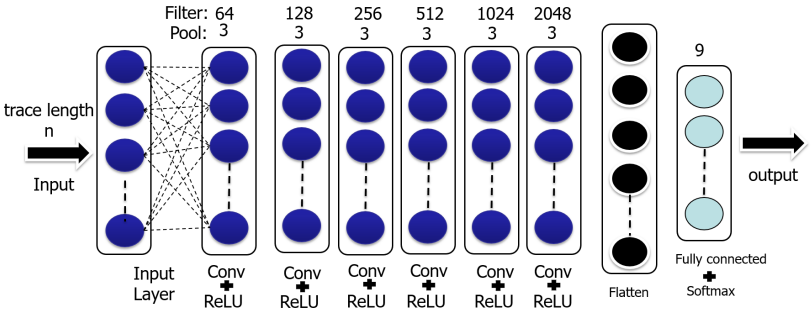


Figure 3.15: Convolution Neural Network.

- Loss Function: The loss function is defined by the categorical entropy technique to compute the error function.
- Optimization: For optimization, Adam [89] is used; it is a special technique that uses adaptive learning rates for each parameter which typically gives good results.
- Regularization: Both batch normalization and gaussian noise are applied. Batch normalization originally was introduced to reduce the random effects of initialization parameters and input data [125]. However, it has been successfully applied to improve generalization [126]. The gaussian layer adds noise to the data before it enters a neuron. As a consequence, instead of only being able to classify the used image, the neural network is also able to classify small changes on that image [127].

After the training achieves sufficient accuracy results (i.e., around 90%), we validate the trained CNN using the validation traces. Thereafter, the attacker can collect the thermal traces from the target device. The attacker has to remove the temperature drift from the collected traces. This trace set is defined as an evaluation set. Subsequently, the attacker applies the evaluation set to the CNN. Finally, more traces of different executions are collected and fed into the CNN. As more traces are used to determine the key, the higher is the chance of a successful attack. This strategy is defined as a vertical attack [128].

3.4.7. PROGRESSIVE CORRELATION THERMAL ATTACK (PCTA)

In the PCTA attack, a fixed amount of key-bits are processed sequentially. As a consequence, when a sub-key of 10 bits is guessed incorrect, the following sub-keys will also be incorrect since they depend on each other. Therefore, it is essential to come up with a proper estimation. To do this, we increase the differences in HW values by using a specific message (i.e., ciphertext since we attack RSA decryption). This specific messages is crafted in the form of $m = N - 1$ [129]. An example of this can be seen in Algorithm 9.

Analyzing the values of R0 and R1 in Algorithm 9, it looks that each round computes exactly the same values. The answers are always $N - 1 = 76$ and 1. However, the multiplications to compute such results are slightly different, which can be:

Algorithm 9 An Example of Calculating RSA With a Specific Message

Variables Declaration:

$d = 43; (b101011)$ ▷ Private Key d
$N = 77;$ ▷ Public Key N
$c = 76;$ ▷ Ciphertext c
$R_0 = 1;$
$R_1 = c = 76;$

Decryption:

$$e[0] = 1 \begin{cases} R_0 = (R_0 * R_1) \mod N \Rightarrow (1 * 76) \mod 77 = 76 \ \{M2\} \\ R_1 = (R_1 * R_1) \mod N \Rightarrow (76 * 76) \mod 77 = 1 \ \{M3\} \end{cases}$$

$$e[1] = 1 \begin{cases} R_0 = (R_0 * R_1) \mod N \Rightarrow (76 * 1) \mod 77 = 76 \ \{M2\} \\ R_1 = (R_1 * R_1) \mod N \Rightarrow (1 * 1) \mod 77 = 1 \ \{M1\} \end{cases}$$

$$e[2] = 0 \begin{cases} R_1 = (R_0 * R_1) \mod N \Rightarrow (76 * 1) \mod 77 = 76 \ \{M2\} \\ R_0 = (R_0 * R_0) \mod N \Rightarrow (76 * 76) \mod 77 = 1 \ \{M3\} \end{cases}$$

$$e[3] = 1 \begin{cases} R_0 = (R_0 * R_1) \mod N \Rightarrow (1 * 76) \mod 77 = 76 \ \{M2\} \\ R_1 = (R_1 * R_1) \mod N \Rightarrow (76 * 76) \mod 77 = 1 \ \{M3\} \end{cases}$$

$$e[4] = 0 \begin{cases} R_1 = (R_0 * R_1) \mod N \Rightarrow (76 * 1) \mod 77 = 76 \ \{M2\} \\ R_0 = (R_0 * R_0) \mod N \Rightarrow (76 * 76) \mod 77 = 1 \ \{M3\} \end{cases}$$

$$e[5] = 1 \begin{cases} R_0 = (R_0 * R_1) \mod N \Rightarrow (1 * 76) \mod 77 = 76 \ \{M2\} \\ R_1 = (R_1 * R_1) \mod N \Rightarrow (76 * 76) \mod 77 = 1 \ \{M3\} \end{cases}$$

- M1: $1 \cdot 1 \bmod N \equiv 1 \bmod N$
- M2: $1 \cdot (N-1) \bmod N \equiv (N-1) \cdot 1 \bmod N \equiv N\text{-}1 \bmod N$
- M3: $(N-1) \cdot (N-1) \bmod N \equiv 1 \bmod N$

This means that there are only three different HW values possible when we look at the output of the multiplication (without the modulo operation). Note that depending on the previous key bit, there are only two possible combinations of operations, either M2 followed by M1 or M2 followed by M3. This behavior is shown in Table 3.6. Consequently, when multiplications M1 and M3 are identified the current key bit can be retrieved. Luckily, M1 and M3 have a very different HW value which improves the chance of a successful key correlation.

Note that this approach can be extended by considering 10 bits of the key simultaneously instead of 1 bit. However, this requires that 2^{10} different HW values have to be computed.

Methodology: The Progressive Correlation Thermal Analysis can be described by the following sequence of steps:

1. Setup template device with a cooling system.

2. Prepare the input message of the target operation.

Table 3.6: Operations according to current key-bit e[i] and previous key-bit e[i-1].

e[i] \ e[i-1]	0	1
0	M2 M1	M2 M3
1	M2 M3	M2 M1

3. Collect thermal traces when running the target operation.
4. Remove temperature drift.
5. Define sub-key size.
6. Perform CTA on the sub-key.
7. Use the resulting guessed sub-key to perform CTA in the next sub-key.
8. Repeat step 7 until the end of the trace.
9. Collect more traces to perform a vertical attack.

The PCTA attack is very similar to the CTA attack but attacks only part of the key at a time using a specific message. To successfully retrieve the key, a vertical attack is preferred where the sub-key is evaluated using multiple traces simultaneously. The sub-key that is the most common among the different traces is most likely the correct sub-key. Note that it is theoretically possible to compare the sub-keys of multiple traces on a bit level instead of a sub-key level. However, this will not improve the results as the derived sub-key could end up not being a result of any of the traces. Therefore comparing each sub-key of all the gathered traces is the preferred method. Finally, the strategy that is used to craft the input message adds a new requirement to the threat model. It does not necessarily make the attack unpractical but might limit its application in the field.

3.4.8. MEASUREMENT SETUP AND PERFORMED EXPERIMENTS

All experiments were performed on the PYNQ-Z1 development board [116]. The PYNQ board runs bare metal C(++) code on one of the two available ARM-A9 cores. We used the board to run both the unprotected RSA (i.e., square and multiply) and the protected RSA (i.e., Montgomery Ladder) implementations using a 1024-bit key. PYNQ-Z1 has an embedded analog-to-digital converter XADC [117], which is connected to power and temperature sensors. It has a resolution of 12 bits and a sampling rate up to 1 mega samples per second (MSPS). In order to read out the data from XADC with minimal noise effects, the Zeroplus Logic Cube (LAP-C 16032) digital logic analyzer is used [130]. The collection of traces, the steps to process them and the attacks have been coded in Python scripts. The measurement setup can be seen in Figure 3.16. Note that the experiment was conducted at room temperature using a clock frequency of the system equals 650MHz and a supply voltage of 1 Volt.

Using this setup, the following experiments were conducted:

Figure 3.16: The Measurement Setup With Logic Analyzer

- Correlation Thermal Attack: In this experiment, CTA is used to attack the unprotected RSA implementation. Nine traces were used.
- DL-based Thermal Attack: In this experiment, the unprotected RSA implementation is attacked using deep learning. Two different sets of traces were used, one with a known key and one with an unknown key. The former set was used for training and validation, while the latter set was used for evaluation. This experiment uses five traces for training and validation and nine traces for evaluation.
- Progressive Correlation Thermal Attack: This experiment use PCTA to attack the Montgomery Ladder RSA implementation. Five traces were used.
- Comparison between Power and Thermal attacks: In this experiment, the results of all evaluated thermal attacks (i.e., the experiments above) are compared with their equivalent power attacks, and hence we also collected power traces.

To evaluate these experiments, the following two metrics are used:

- Performance: The performance is defined as the attack accuracy, i.e., the percentage of key bits that are guessed correctly. A successful attack requires a 100% accuracy for the RSA cryptosystem. Due to the usage of very large key sizes in RSA, even when 95% to 98% of the key-bits have been guessed correctly, it is still hard to brute-force the wrongly predicted key bits when their locations are not known.
- Error histogram: This histogram shows the occurrences of wrong guesses for the trace samples. It provides evidence that vertical attacks are practical, as long as the errors are more or less random (i.e., have a uniform distribution).

3.4.9. CORRELATION THERMAL ATTACK RESULTS

The results of the CTA experiment are presented in Table 3.7. Among the nine attacked traces, the performance varies between 92 and 95 percent. In case the traces are attacked

isolated, such results are insufficient for a successful attack. However, when analyzing the error histogram depicted in Figure 3.17, we observe that the errors of these traces are distributed uniformly along the key-bits. Consequently, this means that the key can be reconstructed using a vertical attack with majority voting. Note that the histogram also shows that the first two bits are wrongly guessed in most traces. This makes sense as the input of the square and multiply is not limited by the modulo yet. However, this does not prohibit the reconstruction of the key, as the location of such faulty bits is fixed; hence they can be easily brute-forced. Therefore, our CTA attack achieves a 100% accuracy with only nine traces when a vertical attack is used, as shown in the last column of the table.

3.4.10. DL-BASED THERMAL ATTACK RESULTS

The DL-based thermal attack has two phases. The training and attack phases.

Training phase: In order to train the network, these traces had to be separated in two groups. The first group consisting of 90% of the total traces for training and 10% for validation. To make sure that the validation group was uniform, all the traces were first randomly shuffled and afterward split. This makes sure that the CNN is able to handle all kinds of offset. After shuffling, we calculated the percentage of ones and zeros at each dataset. The training of the CNN is considered successful when at least 90% accuracy is achieved. Figure 3.18 shows the accuracy results of the training phase; it shows the results for both the training and validation sets. Both cases achieved a maximum accuracy above 90% which indicates that the attack can be successful.

Attack phase: In this phase, the evaluation set consisting of nine traces is used to retrieve the key. Table 3.8 shows that the results of the individual traces have a 91% to 95% accuracy. We also applied error analysis to understand the error distribution behavior. The combined error histogram of all the traces can be seen in Figure 3.19. It shows again a uniform distribution of the errors except for the first bits. Hence, a vertical attack is possible. Since the key in all executions is fixed, the CNN was able to completely retrieve the key when a vertical attack was applied on these nine traces.

3.4.11. PROGRESSIVE CORRELATION THERMAL ATTACK RESULTS

PCTA is highly dependent on the correct key guesses during the iterations over the subkeys. This means that one incorrect guess makes the attack unsuccessful. Table 3.9 shows for five different traces the number of key-bits that have been predicted correctly until the first faulty prediction occurred. Note that in the best case, only 25% of the key-bits have been predicted correctly.

To increase the accuracy, also here information from multiple traces could be combined. The first method would consist of simply adding traces together [131]. This increases the differences between low-intensity and high-intensity operations. However, it requires all traces to be perfectly aligned in time, and more importantly, they must have the same offset. The last requirement is a problem in the case of temperature because our pre-processing does not completely remove the temperature drift effect. Therefore, instead of adding all the traces together, we applied a vertical attack with majority voting during every intermediate step (i.e., at each sub-key correlation).

In this setting, it was possible to retrieve the full 1024-bit key with 5 traces only. The

Table 3.7: Results of CTA on the unprotected implementation of RSA

1	2	3	4	5	6	7	8	9	Vertical
93.1%	92.8%	92.8%	93.0%	94.4%	92.8%	94.3	93.6%	92.9%	100%

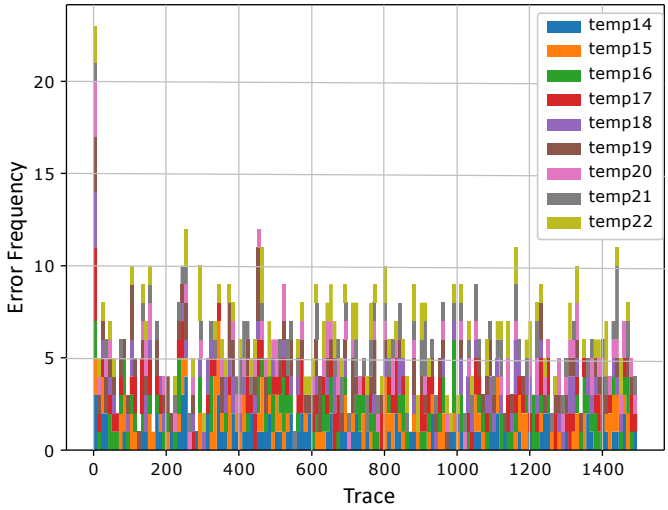


Figure 3.17: Error Histogram of Correlation Thermal Attack

Table 3.8: Results of cnn on the unprotected implementation of RSA

1	2	3	4	5	6	7	8	9	Vertical
93.7%	93.9%	93%	93.6%	95.1%	91%	94.1%	94.2%	92.7%	100%

Table 3.9: Results of PCTA on the (Montgommery implementation of RSA with $c = N - 1$)

1	2	3	4	5	Vertical
257	97	96	47	108	1024
25%	9.4%	9.3%	4.5%	10.5%	100%

frequency of correctly predicted intermediate sub-key values can be seen in Figure 3.20. Due to the usage of five traces, the maximum frequency is five. The figure shows that the confidence of the attack was very high in most cases, and that with only 5 traces the whole key could be recovered. We recommend to use PCTA with more traces to increase the predicted key's confidence level.

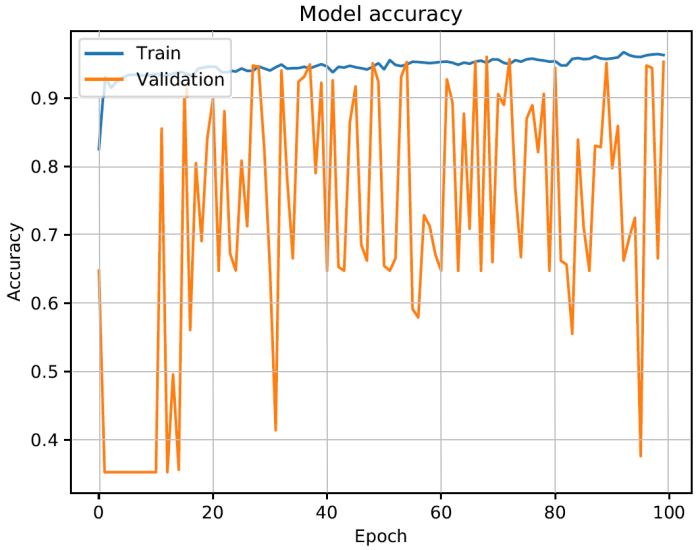


Figure 3.18: Training and Validation Curves of the CNN

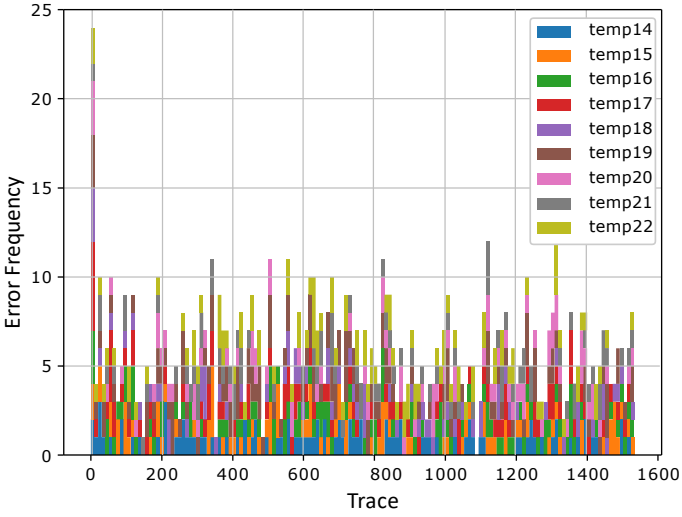


Figure 3.19: Error Histogram of DL-based Thermal Attack

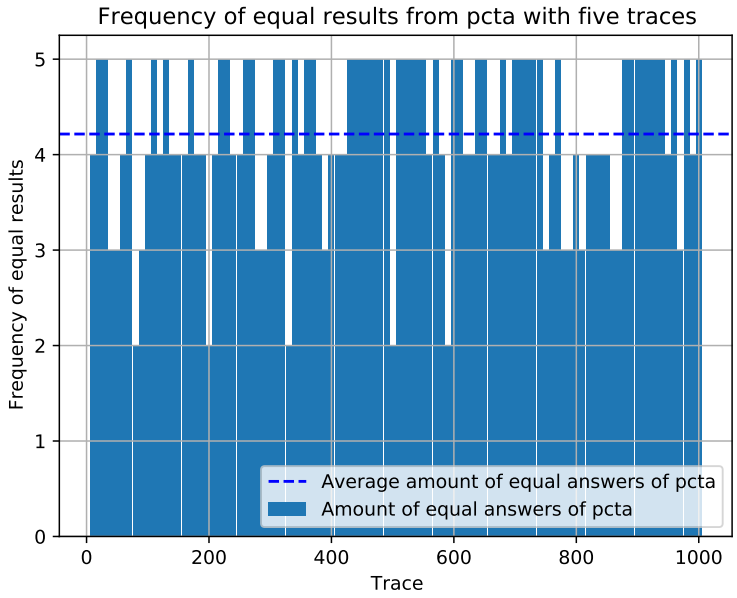


Figure 3.20: Frequency of Intermediate Results With Five Traces

3.5. DISCUSSION AND CONCLUSION

In the field of cybersecurity, a great deal of experimentation has been carried out with a wide variety of side channel attacks in order to determine which ones are the most effective. A wide variety of techniques, including thermal emission and time variation, as well as power consumption, have been investigated. Power-based attacks stand out among these because of the effectiveness they possess. They are distinguished from thermal methods by the fact that they do not require expensive equipment or specialized knowledge, and they also demonstrate a lower sensitivity to noise. Additionally, power attacks make it easier to zero in on the target of the attack, which is a task that is significantly more difficult to accomplish when dealing with timing side channels. The straightforward nature of power attacks, which only require the measurement of power, contributes to the fact that they are feasible and appealing to potential attackers.

Furthermore, the versatility of power side channel attacks extends to their capacity to compromise other channels, such as timing and electromagnetic, which broadens the potential impact of these attacks. Due to the fact that they possess this capability, they are especially pertinent in the process of developing techniques for assessing side channels. By concentrating on power as the primary mode of leakage, researchers have the potential to discover a greater variety of vulnerabilities while simultaneously reducing the complexity of their work. Through the utilization of this all-encompassing approach, our comprehension of side channel vulnerabilities is improved, thereby paving the way for the implementation of more robust security measures and mitigation strategies in

response to the ever-evolving cyber threats.

4

COUNTERMEASURES

This outlines the four countermeasures that were developed during the course of the present study. Section 4.1 introduces a neural network-driven version of the Advanced Encryption Standard (AES) algorithm with the objective of obfuscating the adversary. Section 4.2 introduces a countermeasure approach for addressing the issue of balancing the power consumption in asymmetric algorithms such as RSA and ECC. Section 4.3 introduces a set of algorithms that integrate randomization and balancing techniques in a lightweight manner. Section 4.4 introduces a lightweight implementation of the DOM-based Advanced Encryption Standard (AES).

This chapter is partially published on [19] [36] [41] [40] and also partially submitted to **Cryptography** an international, scientific, peer-reviewed, open access journal on cryptography.

4.1. S-NET: A COUNTERMEASURE BASED ON CONFUSION

This section explains the *confusion* countermeasure, the idea behind S-NET and finally, the methodology to design it.

4.1.1. CONFUSION: INVALIDATING THE LEAKAGE MODEL

In side channel analysis (SCA), an attacker correlates the power consumption with a leakage model assuming a linear relation between them. In other words, a higher power consumption results in a larger hamming weight/distance as illustrated in the left part of Figure 4.1. Hence, the different hamming weights/distances are traceable in the power traces. Note that the countermeasures based on randomization and blinding try to make this harder, but are typically not able to completely hide this linear relation when statistical analysis are performed. The reason for this is that these countermeasures only try to modify the power consumption, as shown in the left part of Figure 4.2. On the other hand, it would be much more difficult for attackers to analyze power traces when the relation between the hamming weight/distance is nonlinear with the actual power consumption as the right part of Figure 4.1 shows. In such a scenario, based on the message-key combination, different hamming weights/distances might have the same power consumption and message-key combinations with the same hamming weights and/or distance might have a different power consumption; hence, attacks based on hamming weight and/or distance are confusing and not effective. The reason for this is that such a countermeasure confuses the leakage in relation to the power consumption. Therefore, this countermeasure targets the leakage model as illustrated in the right part of Figure 4.2.

Note that the implementation of S-NET inherits the non-linearity from the stochastic properties of neural networks. Generally any mathematical function that tries to break the linear power-leakage behaviour can be categorized as a countermeasure based on confusion.

4.1.2. MOTIVATION BEHIND S-NET

Besides their stochastic properties, neural networks also have other benefits. Neural networks can be considered to a certain degree as black boxes as it is unclear how their internals precisely work. This property makes neural network based implementations difficult to be characterized. Hence, finding a good leakage model against it is extremely hard.

4.1.3. DESIGN METHODOLOGY

Figure 4.3 shows the concept of S-NET. S-NET implements the SBOX operation using a neural network without affecting the remaining AES operations. The size and weights of the neural network can be achieved by iterating over three steps, namely design, training, and optimization until a satisfying solution is reached. Thereafter, in the final and fourth step, the neural network is integrated with the other parts of AES. Each step is described in detail next.

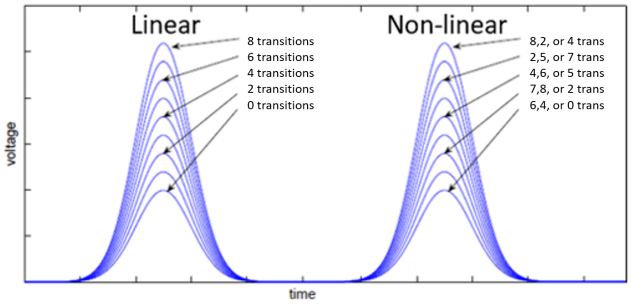


Figure 4.1: Linear power-leakage correlation (modified from [132]).

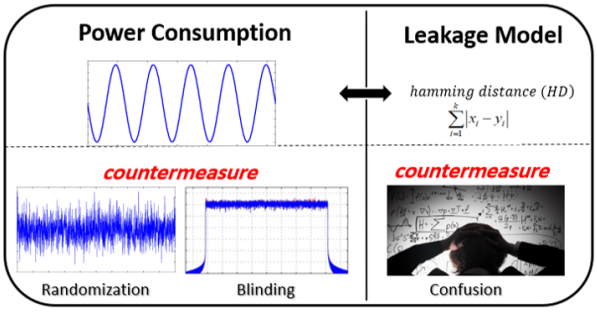


Figure 4.2: Visual explanation of the confusion concept.

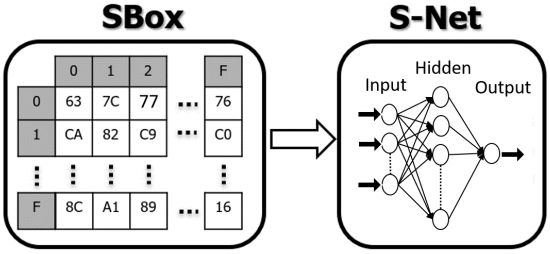


Figure 4.3: SBOX representation in S-NET

1. DESIGNING S-NET:

This step describes the methodology used to define the sizes of the input, output and hidden layers of S-NET.

The SBOX is typically represented by the look-up-table (LUT) shown in left side of Figure 4.3. The LUT contains 256 elements arranged in a table with 16 rows and 16 columns. The row index is specified by the first 4 input bits and the column index by the latter 4

input bits. Since a neural network is not a table, S-NET is designed differently. The input layer of S-NET is fixed to 8 neurons, each representing a single bit of the input, respectively. To improve the resilience against attacks, only a single neuron in the output layer has been used that generates the output byte of the SBOX. The size of the hidden layer, i.e., its width and depth, depends on how easy it is for the neural network to learn the content of the LUT. We have tried different widths and depths to find the optimal solution in term of computation and memory efficiency. We observed that the cheapest solution from a computational and memory point of view consists of using a single hidden layer for two reasons: 1) as the inputs are binary, no multiplications are required in the hidden layer, and 2) by reducing the depth to a single layer, data can be represented using less number of bits. Note that the range of intermediate values increases for a larger depth.

2. TRAINING S-NET:

This step describes the training process and how the weights and biases of S-NET are determined. Usually the data set consists of three subsets during the training of a neural network. One subset is used for the training of the network, one for the validation of the network, and one for evaluating the performance after the training is completed. However, in case of S-NET, only a single data set is used for training. The validation and evaluation are not needed as S-NET must be 100% functional, i.e., it must generate correct outputs for all 256 SBOX inputs.

3. OPTIMIZING S-NET:

This step describes the optimization techniques used to increase the performance and reduce the overhead of S-NET.

The computational complexity and memory overhead of neural networks make them undesirable solutions for both hardware and software applications. Therefore, to reduce the cost of the proposed solution, multiple optimization techniques are applied before, during, and after the training process. These techniques are highlighted next.

Integer weights: It is well understood that integer operations have a significant performance benefit in comparison with floating point operations. Therefore, the weights of the neural network are rounded to the nearest integers after the training phase. After this step, all the inputs of the SBOX are reevaluated to guarantee correct operation.

Constrain weights: The neural network typically produces a wide range of values for the weight set and hence floating point numbers are used by default during training. An implementation of a neural network in hardware and software would be more optimal if the weight set is restricted to a limited number of bits. In S-NET, we fixed the sizes of the weights to 16 bit integers, thereby speeding up the operations and lowering the memory overhead, especially when customized hardware operations are used.

Reduce multiplications: Multiplications are one of the most expensive operations in the neural network. For this reason, S-NET is designed to have a single hidden layer where no multiplications are needed as the input neurons are represented by a single bit. In the output layer, the number of multiplication is reduced by setting a threshold for the weight. Any weight value below this threshold is skipped. Hence, it results in a lower computational overhead.

Use simple activation functions: Each neuron contains an activation function. The input to this activation is equal to the sum of the product of the inputs and weights of the neuron plus the bias. Many functions have been used as activation function such as tanh, sigmoid, Rectified Linear Unit (ReLU), etc. The computational complexity of these functions varies. In our design we intentionally chose Relu for the hidden layer and no activation function in the final layer to achieve simplicity in both software and hardware implementations.

4. INTEGRATING S-NET

in this final step, the designed S-NET component is integrated into the AES implementation by replacing the conventional SBOX.

4.1.4. EXPERIMENT SETUP

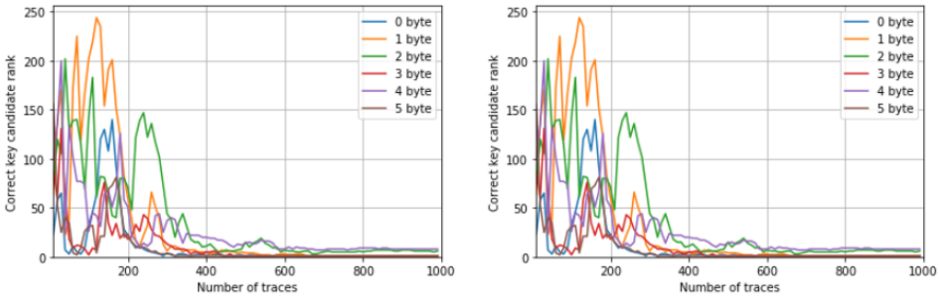
To validate the proposed concept of the countermeasure, we compare the security of an unprotected and protected software implementation of AES128, where the protected implementation uses S-NET. The software implementations run on the Chipwhisperer board from NewAE Technology Inc [94]. It is a development board that comes with the Atmel XMEGA microcontroller as target device. We used the unprotected open source AES128 implementation that comes with the board as our reference for the unprotected AES128 implementation. The power consumption is measured with an ADC that is integrated in the development board. Finally, the development board is connected to a computer to control the execution and storage of the power traces.

4.1.5. RESULTS ANALYSIS

To analyze the security of both the unprotected and protected AES implementations, two analysis methods are applied. They are referred to in literature as evaluation-style and conformance-style testing.

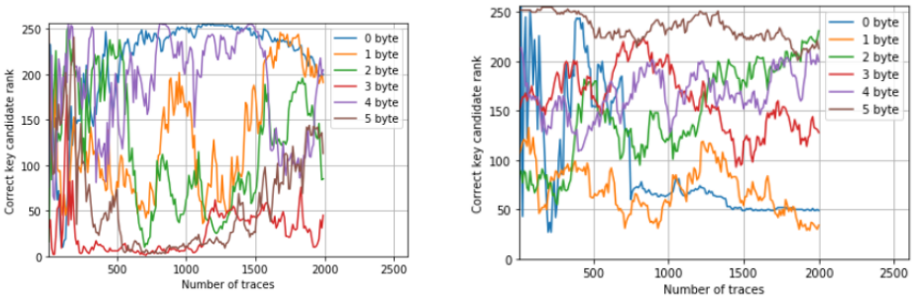
First, in evaluation-style testing traces are examined based on real attacks scenarios, preferably by advanced state-of-the-art attacks. They reveal whether the implementations are resilient against these attacks or not. Here, we limit ourselves to the most famous power attacks; they are: differential power analysis (DPA), and correlation power analysis (CPA). Second, in conformance-style testing the traces are checked to meet certain leakage requirements, without considering attacks. Examples of such analysis are TVLA [73] and signal-to-noise ratio (SNR) analysis [74]. Due to space limitations, we only limit ourselves to SNR analysis. The results of both analysis methods are provided next.

Evaluation-style testing: Two popular attacks (i.e. DPA and CPA) are performed on the recorded traces of the unprotected and protected implementations.The traces are generated based on fixed keys. For each attack, we evaluate the rank of the correct sub-key values (i.e., 8 bits of the 128-bit key). A rank of zero means that the attacker is able to retrieve the correct sub-key, while a rank of 255 represents the lowest confidence of guessing the right sub-key.



(a) DPA (b) CPA

Figure 4.4: Ranking analysis results of unprotected SBOX implementation



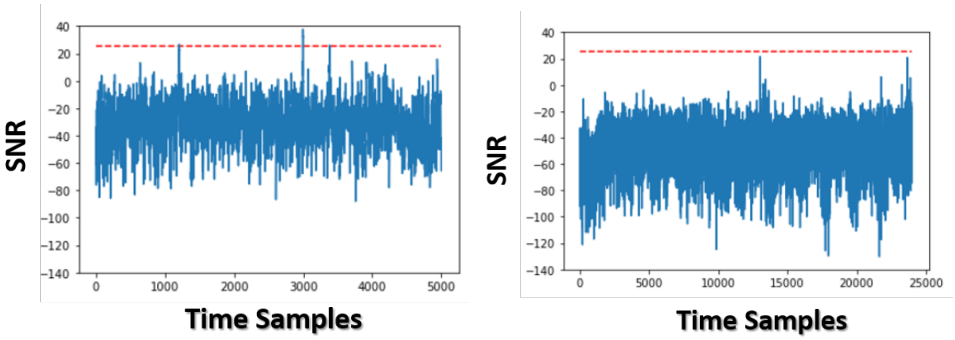
(a) DPA (b) CPA

Figure 4.5: Ranking analysis results of S-NET implementation

Figure 4.4 shows the rank analysis of the first 6 bytes for both attacks for the unprotected implementation. The figure clearly shows, as expected, that the sub-key can be retrieved successfully when approximately 400 traces are used; this applies for both attacks. In contrast, the two attacks were unsuccessful for the protected S-NET implementation as shown in Figure 4.5. The rank of the correct key behaves chaotically and never reaches zero and hence the correct sub-key could not be retrieved. The analysis have been done using only a single weight set for S-NET.

Conformance-style testing: Figures 4.6a and 4.6b show the SNR analysis of the unprotected and protected implementation, respectively. The traces for the analysis are generated based on random keys. The maximum SNR value of both figures differs. For the unprotected case, a high SNR value of 37.6 is observed around sample 3000 which is higher than the considered threshold value (which equals 25 [133]); hence, information leakes. However, for the protected case, the highest observed SNR value is 21.5 around sample 14000, which is below the minimum threshold value. Hence, it is hard to extract the secret key.

The results based on both evaluation-style and conformance-style testing clearly show



(a) Unprotected SBOX implementation (b) S-NET implementation

Figure 4.6: SNR analysis results

Table 4.1: Number of correctly predicted sub-keys

	Unprotected		Protected (S-NET)	
Leakage Model & Attack	DPA	CPA	DPA	CPA
HW(AddRoundKey)	0	0	0	0
HW/HD(SubByte)	16	16	0	0
HW/HD(LastRound)	16	16	0	0

that S-NET is secure against CPA and DPA power attacks. This can also be seen in Table 4.1. In the protected case, we were not able to recover any of the sub-key values. However, for the unprotected case, all the 16 sub-keys were successfully retrieved for attacks based on SubByte and LastRound, for both DPA and CPA using both hamming distance (HD) and hamming weight (HW).

4.2. MULTI-BIT BLINDING: AN ASYMMETRIC COUNTERMEASURE

This section presents the proposed countermeasure method. First, we motivate the reason behind the countermeasure and thereafter detail its design and implementation.

4.2.1. MOTIVATION

There are four main leakage sources from the code that give attackers the ability to retrieve the secret key of software implementations of asymmetric crypto-algorithms like RSA. They are (i) distinguishable operations [9], (ii) branch prediction [134], (iii) operands manipulations [53] and (iv) address manipulations [53]. With respect to *distinguishable operations*, it is easy to recover the key once the main operations are identified in Algorithm 1. When a square operation is followed by a multiplication the secret

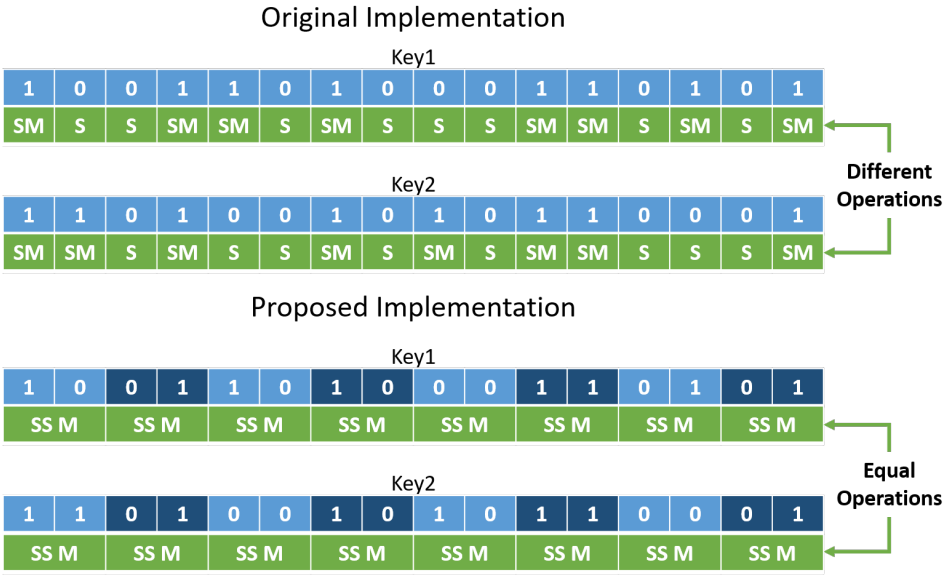


Figure 4.7: Motivation Behind Mulit-Bit Blinding

is bit one otherwise the secret bit is zero. To break this correlation, many countermeasures focused on hiding the operation itself (e.g., using Montgomery multiplication [37]) or adding extra dummy operation such as square and multiply always [135]. However, both such techniques have been broken already [39, 135]. *Branch prediction* leaks information as the branch target buffer is only updated when a branch is taken. An attacker could e.g. use this information to identify whether the if-statement in Algorithm 1 was taken or not and hence whether the key bit is one or not. In order to prevent this, some implementations used a dummy variable to store the discarded values [53]. Although the usage of Montgomery multiplication, dummy operations and dummy variables solved the leakage problems due to *distinguishable operations* and *branch prediction*, they introduced a new vulnerability related to the *operands* and/or *address manipulation* [53]. In case for example a dummy variable is used to store the output of a dummy multiplication when the key bit is zero, the following square operation will share the same operand and since both are executed using Montgomery multiplication, an attacker can compare the power behaviour of both operations to identify the dummy operations. This type of vulnerability is known as operands manipulation. Similarly, when the key bit is zero the address of the dummy variable will be accessed, which differs from the original variable address which results in a different power consumption that can be exploited by an attacker. This vulnerability is known as address manipulation.

One way to address the issues of *distinguishable operations* and *operands manipulation* is by always executing the same operations independent of the key bit without introducing dummy operations. This concept is illustrated in Figure 4.7. The top part shows a naive implementation for two random keys; when the key bit is zero only a square opera-

tion is performed and when the key bit is one a square and multiplication are performed. This results in different patterns which can be identified in the traces by the attacker. Instead, in the lower part of Figure 4.7 two bits are exploited simultaneously, and independent of their value always perform two square operations followed by a multiplication. Hence, all keys end up in the same constant sequence of performed operations. We call this strategy *Multi-bit Blinding* and Equations 1-4 shows how the operands are derived from the four possible two-bit key values.

$$00 \Rightarrow \quad (R_0^2)^2 \Rightarrow \quad R_0^2 \times R_0^2 \tag{4.1}$$

$$01 \Rightarrow \quad (R_0^2)^2 \times R_1 \Rightarrow \quad (R_0^2)^2 \times R_1 \tag{4.2}$$

$$10 \Rightarrow \quad (R_0^2 \times R_1)^2 \Rightarrow \quad (R_0^2)^2 \times R_1^2 \tag{4.3}$$

$$11 \Rightarrow \quad (R_0^2 \times R_1)^2 \times R_1 \Rightarrow \quad (R_0^2)^2 \times R_1^3 \tag{4.4}$$

In the equations, R_0 represent the running state variable and R_1 the ciphertext message as can be also seen in Algorithm 1. For example, Equation 4.4 describes the case where the two-bit key equals 11. Normally, the operation sequence would be square, multiply, square, multiply. The sequence can be rewritten to the sequence containing square, square, and multiply. In all four cases the first square operation uses the same operand R_0 while the second square and the multiplication thereafter use different operands. Note that R_1^2 and R_1^3 can be pre-computed once at the beginning of the algorithm and hence are seen as static variables. This approach has several advantages in making attacks less successful. The constant repeating sequence of operations (i.e., square, square, multiplication) independent from the key requires that the attacker needs to understand which values (i.e., operands) are used during the operations. As the value of R_0 is constantly changing, it is difficult to identify what the two-bit value of the key is based on analyses of power traces. Note that no dummy operations are used, which means that the result of the multiplication operation is used as operand in the following square operation. Hence, it avoids operand manipulation vulnerability.

To cope with *branch prediction* and *address manipulation*, a second strategy is applied referred to as *variable assignment optimization*. In this strategy we replace the conditional statement with a variable assignment using logic operations. However, to avoid *address manipulations*, instead of using a dummy variable, the variable size was doubled. Depending on the value of the two key bits, an operand is either stored in the original or extended part of the variable. This guarantees that the same memory locations are accessed in both cases (i.e., when bit is zero or one). Hence, it avoids address manipulation vulnerability. Next we provide more information on multi-bit blinding and variable assignment optimization.

4.2.2. MULTI-BIT BLINDING

One of the advantages of the proposed method is its simplicity as shown in Algorithm 10. The first part of the algorithm pre-computes the values of R_1^2 and R_1^3. The second part consists of the for loop which is used to walk over the key bits. Note that a step size of two is used as two bits of the key are simultaneously used. *Step 1* computes the square

Algorithm 10 RSA with Multi-bit Blinding

INPUT (c, d, n, r_1) **where** c presents the ciphertext, d the private key which can be represented by its binary represenation as $d_0 \cdots d_{l-1}$ where $k_j \epsilon \{0,1\}$, n the modulus, and r_1 the random mask.

OUTPUT(R_0); **where** $R_0 = c^d \, mod(n)$

$d = d + r_1(n-1)$

$R_0 = 1$

$R_1 = c$

$R_2 = R_1^2$

$R_3 = R_1 \times R_2$

for $j \leftarrow n$ **to** 0 **;** $j \leftarrow j-2$ **do**

Step 1 - Square

$t_1 \leftarrow R_0$

$R_0 \leftarrow R_0^2 \bmod (n)$

$t_2 \leftarrow R_0$

Step 2 - Select operands (Switch)

switch $k[j, j-1]$ **do**

case 00 : $t_1 = t_1; t_2 = t_2$

case 01 : $t_1 = R_0; t_2 = R_1$

case 10 : $t_1 = R_0; t_2 = R_2$

case 11 : $t_1 = R_0; t_2 = R_3$

end switch

Step 3 - Square

$t_1 \leftarrow t_1^2 \bmod (n)$

Step 4 - Multiply

$R_0 \leftarrow t_1 \times t_2 \bmod (n)$

end for

return R_0

operation and initializes the lower halves of t_1 and t_2 with R_0 and R_0^2, respectively. It assumes by default that the key bit values are 00 which are updated based on the actual key. *Step 2* inspects the two key bits using a switch statement that contains the four cases of eqs. (4.1) to (4.4). The variables t_1 and t_2 are used to specify which operands are going to be selected for the square and multiply operation of *Step 3-4*.

4.2.3. VARIABLE ASSIGNMENT OPTIMIZATION

In this optimization, we changed the switch statement (see Algorithm 10 Step 2) and created variable assignments using logic statements without the need of branches. To realize this, we doubled the size of the variables t_1 and t_2 and applied the steps described in Algorithm 11. In order for this to work with Steps 1, 3 and 4, only the lower halves of t_1 and t_2 are used there.

The algorithm use the two key bit values k[j] and k[j-1] to create the following logical assignments: $e_1 = 0$ if the second key bit is one, $e_2 = 0$ if the first key bit is one, and $e_3 = 0$ if both key bits are one. These e variables will be used next to update the values of t_1 and t_2. If the particular values of e are one, the variables are assigned to the upper halves,

Algorithm 11 Variables Assignments (Optimized Step 2)

procedure
$e_1 \leftarrow 1 - k[j-1]$
$e_2 \leftarrow 1 - k[j]$
$e_3 \leftarrow 1 - (k[j] \ \& \ k[j-1])$
$t_1[(0 + e_2 \times size) : (size + e_2 \times size)] \leftarrow R_0$
$t_2[(0 + e_2 \times size) : (size + e_2 \times size)] \leftarrow R_2$
$t_1[(0 + e_1 \times size) : (size + e_1 \times size)] \leftarrow R_0$
$t_2[(0 + e_1 \times size) : (size + e_1 \times size)] \leftarrow R_1$
$t_1[(0 + e_3 \times size) : (size + e_3 \times size)] \leftarrow R_0$
$t_2[(0 + e_3 \times size) : (size + e_3 \times size)] \leftarrow R_3$
end procedure

which means that they will not be used later.

4.2.4. EXPERIMENT SETUP

To validate the proposed countermeasure, we implemented two RSA *multi-bit blinding* software implementations: one with naive *square & multiply* operations and one using *Montgomery multiplication* in C language. In both implementations we added the blinding scheme suggested by Paul Kocher [18] as an additional security layer against vertical attacks; this blinding scheme derives a random key from the base key. Power traces have been collected for both implementations by running the implementations on the Chipwhisperer board from NewAE Technology Inc [94]. Chipwhisperer is a development board comes with an Atmel XMEGA micro-controller that is used as target device. The open source RSA implementation that comes with the board has been used as a baseline and modified to suite the proposed countermeasure. The power consumption on the board is measured with an integrated ADC which is controlled from a computer for the purpose of collecting power traces. Our experiments consider a threat model where the attacker has access to the power traces and ciphertext of the RSA decryptions. Additionally the attacker has access to a similar device in a profiled attack scenario.

4.2.5. SECURITY ANALYSIS

Figure 4.8 shows as an example of a power trace of one loop iteration of the code consisting of four steps presented in Algorithm 10 using the optimization of Algorithm 11. The four steps are identified in the power trace by the numbers with circles on top of the figure and their associated operations. For the security analysis, we ignored the first step as it is identical for all possibilities (i.e., any two-bit key value). Consequently, our profiled and non-profiled attacks are only applied on the remaining three steps.

The result of the non-profiled clustering attack on the naive implementation of the countermeasure, which is depicted in Figure 4.9, shows that the maximum accuracy that can be reached is slightly above 50%. This means that an attacker can only guess 50% of the two bit key values correctly, thus not possible to attack in this manner. We repeated the same experiment using the Montgomery multiplication version of the proposed countermeasure. Similar results were obtained (see Figure 4.10) as the previous implementation and also here it can be concluded that the attack was unsuccessful.

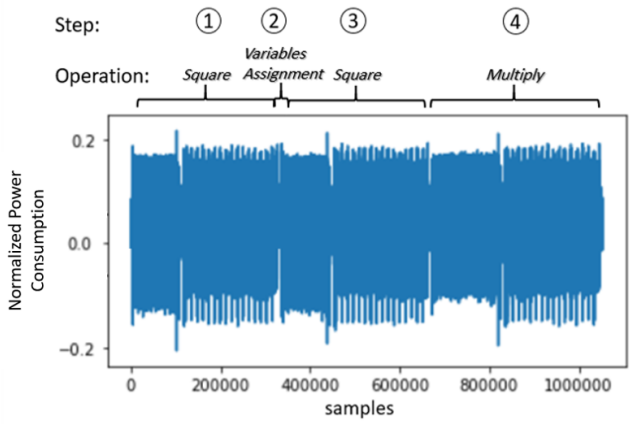
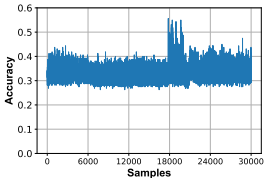
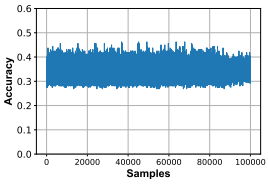


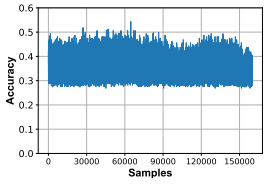
Figure 4.8: Execution Behaviour of 2-Bits of Multi-Bit Blinding



(a) Step 2: Assignment of Variables

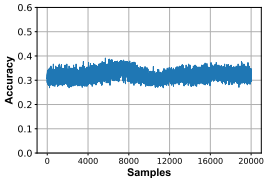


(b) Step 3: Naive Square

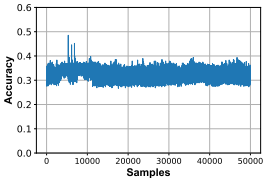


(c) Step 4: Naive Multiplication

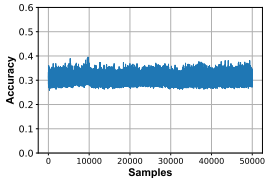
Figure 4.9: Non-Profiled Accuracy Analysis Using Naive Square and multiply Operations



(a) Step 2: Assignment of Variables



(b) Step 3: Montgomery Square



(c) Step 4: Montgomery Multiplication

Figure 4.10: Non-Profiled Accuracy Analysis Using Montgomery multiplication

Figure 4.9a shows a slight increase on the attack accuracy between samples 17500 and 20000 in the variables assignment operation. To ensure that these samples cannot be exploited by an attack we also performed a profiled attack using a convolutional neural network (CNN) [38]. Such attacks are more powerful as they can deal with misalignment in traces. We used 70% of the collected traces for training the CNN and the remaining 30% for validating the network. The attack accuracy results of both implementations

(i.e., naive square & multiply and the Montgomery ones) were similar, as shown in Figure 4.11. In both scenarios, we were able to achieve a 98% accuracy for the training but failed however to achieve a reasonable accuracy in the validating phase. The attack accuracy is similar to the non-profiled case and hence too low for a successful attack.

Our last security evaluation consists of a normalized inter-class variance test (NICV) [136] to ensure that there was no leakage as a result of correlation between executed operations. Due to the limited space this was only applied to the Montgomery based implementation as it is the most secure solution. The results do not show any spikes along the whole trace which clearly indicates the absence of leakage (see Figure 4.12).

4.2.6. PERFORMANCE ANALYSIS

Equation 4.5 is used to estimates the execution time (E) of each implementation. In the equation, N represents the total number of used keys, n_1^i and n_0^i the number of ones and zeros in key i, respectively. Op_1 and Op_0 represent the execution time to process a one and zero key bit, respectively.

$$E = \frac{1}{N}\sum_{i=1}^{N} n_1^i \times Op_1 + n_0^i \times Op_0 \tag{4.5}$$

The execution time is evaluated for 5 different implementations as can be seen in Table 4.2. They are: naive *Square & Multiply* of Algorithm 1, the same implementation using *Montgomery multiplication*, *Square & Multiply Always*, *Multi-bit blinding based on naive Square & Multiply*, and *Multi-bit blinding based on Montgomery Multiplication*. All implementations use the blinding countermeasure proposed by Paul [18]. In the table, S, M, and M_M, represents Square, Multiplication, and Montgomery multiplication, respectively. The performance analysis are performed based on a single key generated by the OpenSLL python library [121] which has been used to derive 100 keys (i.e., $N = 100$)) using the blinding method proposed by Paul [18]. The performance results of each implementation is summarized in Table 4.2. To calculate the overhead (O) of the different implementations, the equation $O = \frac{(E_i - E_{base})}{E_{base}} \times 100$ is used. The execution time E_i of a particular implementation is compared to the baseline E_{base} implementation; as baseline we selected *Square & Multiply*. The table also show the execution time needed to complete the involved operations when a key bit is zero and one (3rd and 5th column in the table). From the table we conclude that the *Proposed Square & Multiply* and *Proposed Montgomery Multiplication* perform better in terms of performance.

4.3. BALANCED DUAL-MASK COUNTERMEASURE

This section discusses the proposed countermeasure approach. First, we motivate the rationale behind it and then discuss its design and implementation.

4.3.1. MOTIVATION

When it comes to securing crypto algorithms from power attacks, most countermeasures revolve around one of two techniques: 1) randomizing the power behavior or 2)

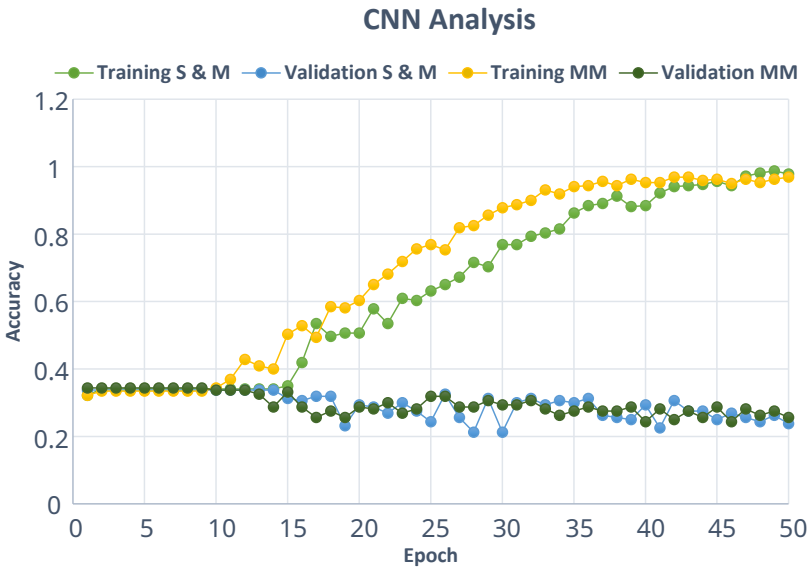


Figure 4.11: Profiled Analysis of the Proposed Scheme

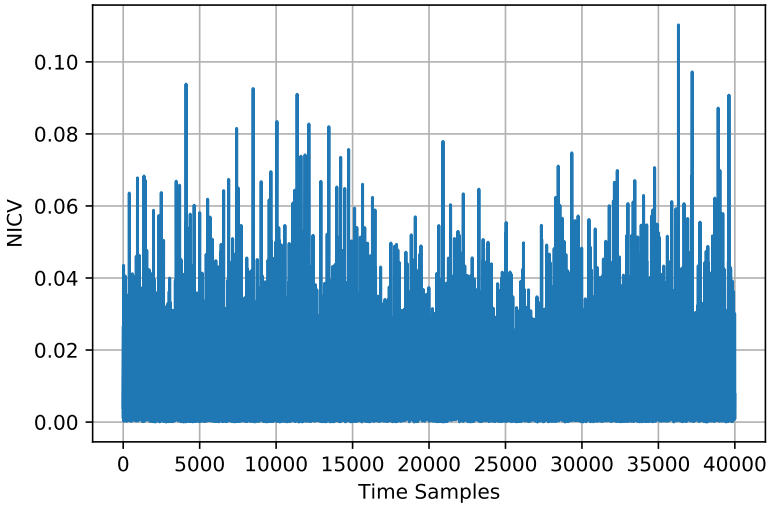


Figure 4.12: Normalized Inter-Class Variance (NICV) Results

balancing the power behavior for every key and plaintext/ciphertext pair. In our study, we considered one popular approach from each technique. For the power randomiza-

Table 4.2: Performance analysis of different implementations scheme

	Bit Zero		Bit One			
Implementation	Operations	Time	Operations	Time	Execution time (E)	Overhead (%)
Square & Multiply	S	330000	(S + M)	724000	1090373200	0%
Montgomery Multiplication	MM	372150	2(MM)	744300	1156471011	6%
Square & Multiply Always	(S + M)	724000	(S + M)	724000	1499563280	37.5%
Proposed Square & Multiply	1/2(S + S + M)	527000	1/2(S + S + M)	527000	1091480240	0.1%
Proposed Montgomery Multiplication	3/2 (MM)	558225	3/2(MM)	558225	1156245860	6%

tion approach, we use masking. Masking [137] introduces multiple randomized shares called the mask. For the power-balance approach, we apply the method in [138] where the output of the sensitive function that leaks the most (i.e., SBox) always results in the same number of ones. Hence, the leakage model will always be the same. Unfortunately, neither method was successful in securing the SubCell function as we will see in the up coming sections. Therefore, a new more robust countermeasure is needed.

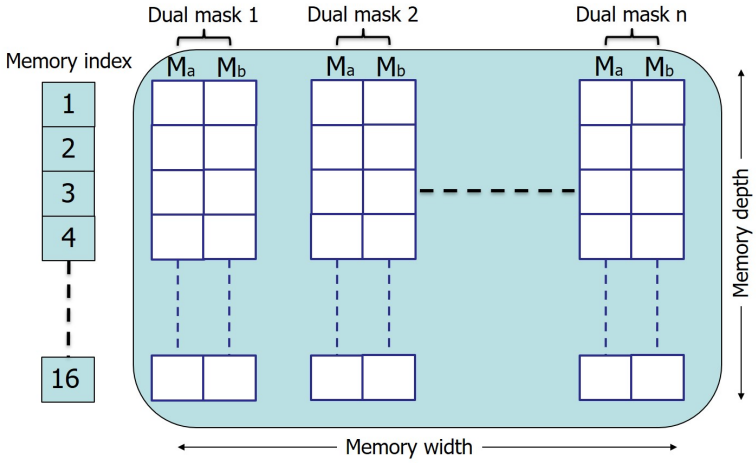
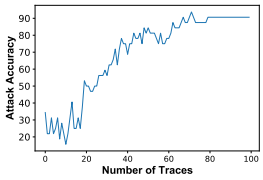


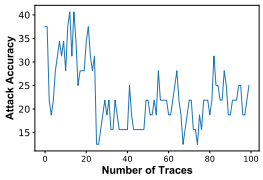
Figure 4.13: Balanced Dual Masks Scheme

4.3.2. DESIGN AND IMPLEMENTATION

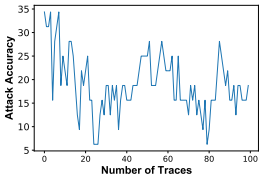
Both power-balancing and masking countermeasures failed to protect the SubCells function (see Section 4.3.4). To overcome their limitations, we propose a balanced dual-masks scheme. In this technique, as illustrated in Figure 4.13, we apply two masks to each SubCells index; both masks together contain 4-bits of actual data and 4-bits of dummy data. This dummy data can reside (partly) in either of the masks. On top of that, instead of using a single set of dual masks we can integrate n different sets. Therefore, in the figure, M_a and M_b of dual mask 1 are not equal to M_a and M_b of dual masks 2. During run-time, only one of the outputs related to the n sets will be used and the dummy bits will be filtered out.



(a) CPA on SubCells



(b) CPA on PermBits



(c) CPA on AddRoundKey

Figure 4.14: Accuracy Analysis of Non-profiled Attacks

4.3.3. EXPERIMENT SETUP

To validate the proposed attack scheme, the publicly available open-source software implementation of the GIFT 128-bit cipher [107] was used. The GIFT 128-bit program is written in C code. The power traces have been collected by running the implementation on the Chipwhisperer board from NewAE Technology Inc [94]. Chipwhisperer is a development board that comes with an Atmel XMEGA micro-controller that is used as a target device. It has been used in many attacks such as ECC [139]. The power traces were captured by an Analogue-to-Digital Converter with a sample rate of 105 MS/s. Both the target chip and the measurements setup are connected to the computer using a USB interface, to execute the program and transfer the recorded traces. The proposed attack was implemented in Python using the Keras library. This is an open-source software library that can be used to create, train, and run artificial neural networks.

4.3.4. SECURITY ANALYSIS OF NAIVE IMPLEMENTATION

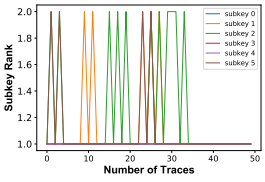
To validate the GIFT cipher against power attacks, three functions were selected as targets: 1) the *SubCells* function during second and third rounds; 2) the *PermBits* of the second round; and 3) the *AddRoundKey* in first round. They have been used in both non-profiled (i.e., CPA) and profiled attacks(i.e., deep learning power attack). Their results are discussed next.

NON-PROFILED ATTACKS

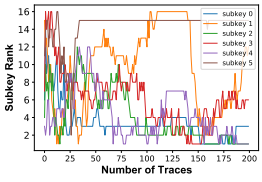
The attacks were evaluated initially with a single trace and iteratively reevaluated by adding each time a single trace until 100 traces have been used. Figure 4.14 show the accuracy analysis of CPA attacks for the three attacked functions, respectively. A higher accuracy value means that more sub-keys were correctly guessed; e.g., a 100% accuracy means that all sub-keys were correctly guessed. We observe that CPA attack was successful in recovering the majority of sub-keys values for SubCells while targeting other functions was unsuccessful. The more traces are used, the closer the guessed sub-key is from the correct sub-key. The results indicate that the software implementation of GIFT is attackable using non-profiled techniques. Although a few sub-keys have not been attacked successfully, it could be possible to attack the entire key when more traces are added.

Algorithm 12 Extract Key bits

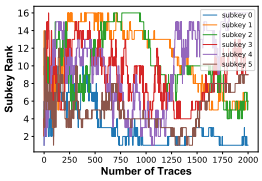
procedure KET_EXTRACT($Traces_{set}, pt_{array}$)
  pt = output of Permutation XORed with constant
  $P_k[0,15]$ = key probability
  $predict$ = is the trained model the sub-key
  for each sub-key **do**
    $P_k[0,15] = 0$
    for each trace in trace-set **do**
      $X_{0,15} = predict(trace)$
      for k=0 to 15 **do**
        $y = lkf(SBOX[pt[sub-key] \bigoplus k])$
        $P_k[k] = P_k[k] + X[y]$
      end for
    end for
    $guess_{sub-key} = max(P_k)$
  end for
end procedure



(a) Subcells of Second Round

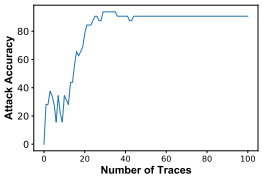


(b) PermBits

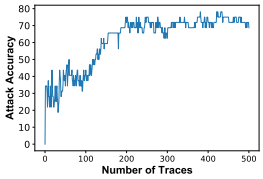


(c) AddRoundKey

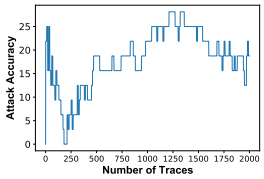
Figure 4.15: Rank Analysis of Profiled Attacks



(a) Power balancing



(b) Power Randomizing



(c) Balanced Dual Masks

Figure 4.16: Countermeasures Analysis using Non-profiled Attacks

PROFILED ATTACKS

The performed profile attacks are described in Section 2.2. The results for *SubCells* in terms of rank analysis is shown in Figure 4.15a; the lower the rank, the better the guessed sub-key.. All sub-keys reached a rank of 1 (i.e., they were fully recovered) using only 50

traces. Similarly, the results for *PermuBits* and *AddRoundKey* are shown in Figures 4.15b and 4.15c. The key ranking results show random behaviour which means that it is difficult to retrieve the correct sub-key value.

4.3.5. SECURITY ANALYSIS OF PROPOSED IMPLEMENTATION

First, we separately evaluated the balancing and masking countermeasures presented in [138] and [137], respectively. Using CPA, we were able to achieve a high accuracy with only a few traces for both countermeasures as shown in Figures 4.16a and 4.16b. Next, we evaluated the security analysis using one single dual mask only (i.e., only Dual mask 1) to validate the minimum security level of the proposed approach. Note that only the SubCells function was targeted as both AddRoundKey and PermBits functions were unattackable in the naive implementation. In the non-profiled attack (i.e., CPA) the approach was secure as the maximum accuracy the attack could reach was 25% as shown in Figure 4.16c. However, using the profiled attack (i.e., deep learning) the attack was not fully secure as some of the sub-keys were recovered as can be seen in Figure 4.17. To solve this issue, we increased the number of dual masks to two, i.e., we used different masks for profiling and attack phases where n=2 in both cases. The results shows that the sub-keys are secure as shown in Figure 4.18. In order for an attacker to create a successful template, he needs to consider all 16^n combinations for the different masks. This becomes quickly infeasible for n=8.

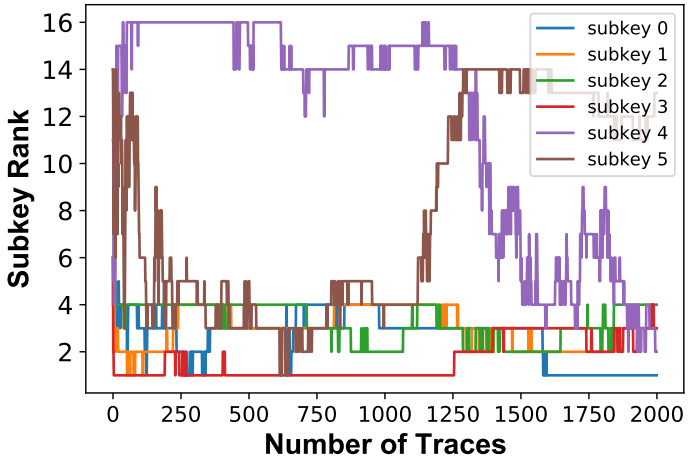


Figure 4.17: Single Dual Masks Analysis

4.3.6. AREA OVERHEAD AND PERFORMANCE ANALYSIS

Our proposed technique only increases the width of the SBox table. Therefore, there is no area overhead unless the number of dual masks exceeds the word size (i.e., 4 dual masks

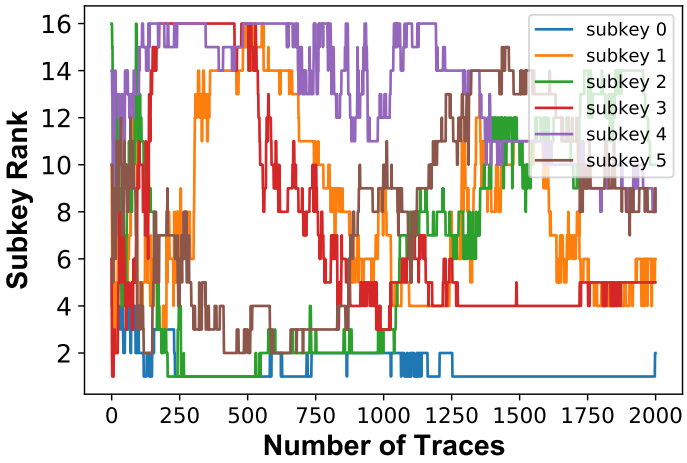


Figure 4.18: Double Dual Masks Analysis

in 32-bit wide memories and 8 dual masks in 64-bit wide memories). The performance overhead is measured by the additional number of instructions added to the baseline execution. Since these extra instructions are only required to multiplex the dual masks and select one of them. Hence, the increase in execution time is negligible.

4.4. LIGHTWEIGHT AES AND DOM EXTENSION

This section presents our proposed implementation approach for AES and its protected version using DOM. We start by motivating our approach, followed by a detailed explanation of its design and implementation.

4.4.1. MOTIVATION

Previous research primarily focused on implementing AES on either an 8-bit [140–146] or 32-bit [147, 148] data-path to reduce area and energy consumption. However, these studies only report the area of the encryption module and neglect the decryption part. In reality, the eleven 128-bit registers required for the key expansion in the decryption module contribute significantly to the overall core area. In the decryption module, all round keys must be computed first before the decryption can start. Shortening the data-path from 128-bit to a lower-bit width has a much lower improvement on the area when the decryption module is not ignored. Therefore, in our design we focus on different data-paths in the presence of the decryption unit and compare their performance in terms of throughput, area, and power. Secondly, in actual applications, keys do not change frequently. Hence, we perform the key expansion once and store the results in the registers. As long as the key remains the same, we can skip the key expansion step, resulting in a significant power and latency reduction. In addition, we reorder the sequences of

Addroundkey and *MixColumns* in the round function which results in further area and performance improvements.

4.4.2. DESIGN AND IMPLEMENTATION OF PROPOSED LIGHTWEIGHT AES

Our proposed AES designs verify whether the key changes at the start of every encryption/decryption execution. In case a key change is detected, we perform the key expansion module and leave the keys inside the key registers. Otherwise, we directly execute the round modules (i.e., *AddRoundKey*, *SubBytes*, *MixColumns*, and *ShiftRows*). This reduces the execution time of the decryption part by eleven cycles. To further optimize the design area, resource sharing is employed. Initially, we limit the number of registers to store the state to a single 128-bit register that is shared in all the round modules of both encryption and decryption. Next, we combine the encryption and decryption modules to decrease the overall area. The proposed scheme is depicted in Fig. 4.19, where *cnt* represents the round index and *key[cnt]* denotes the key that needs to be XORed with the State array. The boxes containing the word "shared" represent blocks that are shared between the encryption and decryption. An in-depth explanations of the shared modules will be provided next, including *Shared SBOX, Shared ShiftRows*, and *Shared MixColumns*.

SHARED SBOX

Akashi et al. [149] proposed a new composite field to optimize the structure of the SBOX, resulting in a significant reduction of the area compared to using a Look-up table (LUT). Thereafter, several researchers [44, 150, 151] optimized the SBOX based on the structure provided in [149]. These papers used an SBOX which is shared by both the encryption and decryption modules to reduce area. To the best of our knowledge, the SBOX design described in [151] has the lowest area. Compared with previous designs, they shared resources in three modules: preprocess, postprocess, and scalar square. The preprocess module performs the isomorphic mapping and inverse affine transformation for the decryption and the isomorphic mapping only for the encryption. The postprocess module executes the affine transformation and the inverse isomorphic mapping for the encryption and inverse isomorphic mapping only for the decryption. The scalar square performs a square and multiplication with constant $\lambda = \{1,1,0,0\}$, which leads to three XOR reductions [44]. Our new SBOX is based on the SBOX proposed in [151]; it is shown in Fig. 4.20. It contains an optimized multiplier and a modified inverter. In addition, it

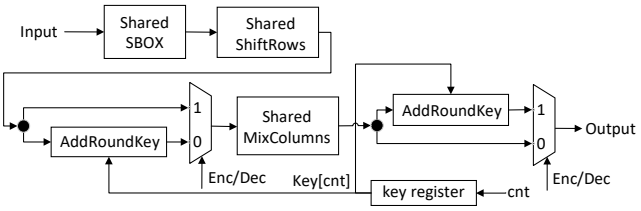


Figure 4.19: Proposed Round Function for AES Encryption and Decryption

combines the operations of the last two multipliers proposed in [149]. Each optimization is described next into more details.

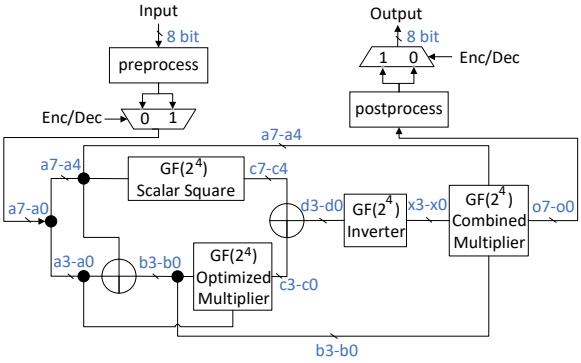


Figure 4.20: Proposed Shared SBOX

- $GF(2^4)$ **Optimized multiplier:** Our optimized $GF(2^4)$ multiplier is based on the work in [151]. That multiplier consists of 18 XOR and 12 AND gates and its critical path consists of 4 XOR and 1 AND gate. We simplified the $GF(2^4)$ multiplier based on Equation (4.6), where $\{a_3, a_2, a_1, a_0\}$ and $\{b_3, b_2, b_1, b_0\}$ denote the two 4-bit inputs (see also left bottom of Fig. 4.20), $\{c_3, c_2, c_1, c_0\}$ denote the 4-bit output, and $\{m_4, m_3, m_2, m_1, m_0\}$ are intermediate variables defined as: $m_4 = m_0 \oplus m_1$, $m_3 = a_0 \oplus a_1$, $m_2 = a_3 \oplus a_2$, $m_1 = a_2 \oplus a_0$, and $m_0 = a_3 \oplus a_1$. Although our $GF(2^4)$ optimized multiplier utilizes 4 more AND gates compared to [151], it requires 1 XOR gate less and more importantly has a shorter critical path (1 AND gate and 3 XOR gates). Surprisingly, after synthesis it turns out that the area of this implementation is also better after synthesis. We believe that compiler is able to extract more common resources with this implementation.

$$
\begin{aligned}
c_3 &= [(a_3 \oplus a_1)\&(b_3 \oplus b_1) \oplus (a_3 \oplus a_1)\&(b_2 \oplus b_0) \oplus (a_2 \oplus \\
&\quad a_0)\&(b_3 \oplus b_1)] \oplus [(a_1\&b_1) \oplus (a_1\&b_0) \oplus (a_0\&b_1)] \\
&= (b_0\&a_3) \oplus (b_1\&(a_2 \oplus a_3)) \oplus (b_2\&(a_1 \oplus a_3)) \oplus \\
&\quad (b_3\&(a_0 \oplus a_1 \oplus a_2 \oplus a_3)) \\
&= (b_0\&a_3) \oplus (b_1\&m_2) \oplus (b_2\&m_0) \oplus (b_3\&m_4); \\
c_2 &= [(a_3 \oplus a_1)\&(b_3 \oplus b_1) \oplus (a_2 \oplus a_0)\&(b_2 \oplus b_0)] \oplus (a_1\&b_1) \\
&\quad \oplus (a_0\&b_0) \\
&= (b_0\&a_2) \oplus (b_1\&a_3) \oplus (b_2\&(a_0 \oplus a_2)) \oplus (b_3\&(a_1 \oplus a_3)) \\
&= (b_0\&a_2) \oplus (b_1\&a_3) \oplus (b_2\&m_1) \oplus (b_3\&m_0); \\
c_1 &= [(a_3\&b_3) \oplus (a_3\&b_2) \oplus (a_2\&b_3)] \oplus [(a_3\&b_3) \oplus (a_2\&b_2)] \\
&\quad \oplus [(a_1\&b_1) \oplus (a_1\&b_0) \oplus (a_0\&b_1)] \\
&= (b_0\&a_1) \oplus (b_1\&(a_0 \oplus a_1)) \oplus (b_2\&(a_2 \oplus a_3)) \oplus (b_3\&a_2) \\
&= (b_0\&a_1) \oplus (b_1\&m_3) \oplus (b_2\&m_2) \oplus (b_3\&a_2); \\
c_0 &= [(a_3\&b_3) \oplus (a_3\&b_2) \oplus (a_2\&b_3)] \oplus [(a_1\&b_1) \oplus (a_0\&b_0)] \\
&= (b_0\&a_0) \oplus (b_1\&a_1) \oplus (b_2\&a_3) \oplus (b_3\&(a_2 \oplus a_3)) \\
&= (b_0\&a_0) \oplus (b_1\&a_1) \oplus (b_2\&a_3) \oplus (b_3\&m_2).
\end{aligned}
\tag{4.6}
$$

- $GF(2^4)$ **Inverter:** To decrease the area of the inverter and make the design easier to secure (by performing less non-linear operations), we further optimized the inverter based on the structure that proposed in [44]. The improved design is shown in Fig. 4.21. The $GF(2^2)$ *Scalar Square* performs a $GF(2^2)$ square operation and a scalar multiplication with the constant $\varphi = \{1,0\}$. Equation (4.7) illustrates the *Scalar Square* calculation of [44] (left) and our combined design (right), where d_3, d_2 are the inputs of *Square* module, e_3, e_2 denote intermediate results between *Square* and *Scalar* module, and f_3, f_2 represent the outputs of *Scalar* module (see Fig. 4.21). As can be seen, our design requires 2 XOR operations less.

$$
[20]\begin{cases} e_3 = d_3 \\ e_2 = d_3 \oplus d_2 \\ f_3 = e_3 \oplus e_2 \\ f_2 = e_3 \end{cases}
\qquad
Combined\begin{cases} f_3 = d_2 \\ f_2 = d_3 \end{cases}
\tag{4.7}
$$

The last step of the inverter consist of two $GF(2^2)$ multipliers. Equation (4.8) shows the design proposed in [44]. Our optimizations are provided after the second "=" sign by factoring out $X = h_1 \oplus h_0$ commonly between both multiplications. Our combined $GF(2^2)$ multiplier achieves a reduction of 1 XOR gate and 2 AND gates, compared with the design in [44].

$$
\begin{aligned}
x_3 &= (d_3\&h_1) \oplus (d_3\&h_0) \oplus (d_2\&h_1) = (d_3\&X) \oplus (d_2\&h_1); \\
x_2 &= (d_3\&h_1) \oplus (d_2\&h_0); \\
x_1 &= (e_1\&h_1) \oplus (e_1\&h_0) \oplus (f_0\&h_1) = (e_1\&X) \oplus (f_0\&h_1); \\
x_0 &= (e_1\&h_1) \oplus (e_0\&h_0).
\end{aligned}
\tag{4.8}
$$

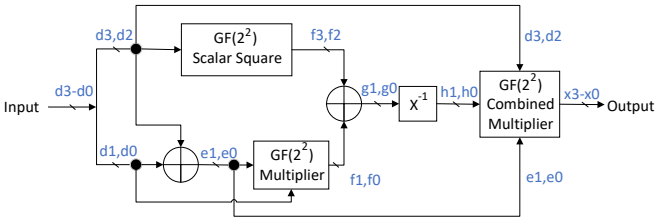


Figure 4.21: Proposed $GF(2^4)$ Inverter

- $GF(2^4)$ **Combined Multiplier:** The last two multipliers used in the SBOX presented in [151] were treated as two separate multipliers. However, Ahmad [152] proposed that these two multipliers can be merged together, resulting a significant reduction in area as can be seen at the most right part in Fig. 4.20. However, it is not clear from the paper how this shared multiplier works. For clarity, we combined the multipliers ourselves and provided a detailed structure of it in Fig. 4.22.

SHARED SHIFTROWS

Davis and John observed that the first and third shift operations in *ShiftRows* and *InvShiftRows* can be shared [148], as both produce the same results for the decryption and encryption. This can be seen in Fig. 4.23. However, the other two rows (i.e., row two and four) have different behavior and multiplexers are needed to select between them.

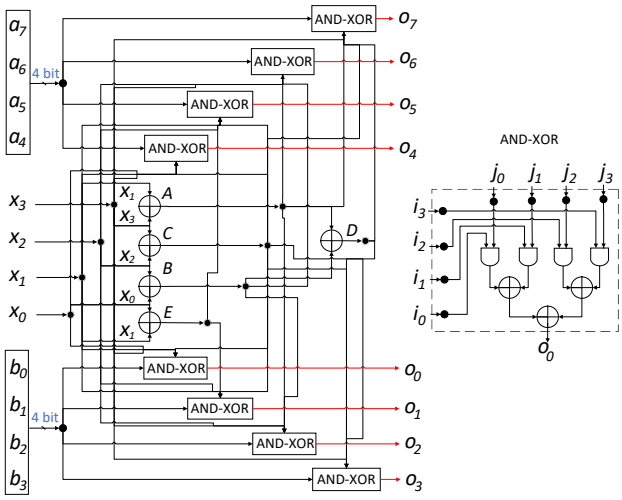


Figure 4.22: $GF(2^4)$ Combined Multiplier

$$\begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,1} & S_{1,2} & S_{1,3} & S_{1,0} \\ S_{2,2} & S_{2,3} & S_{2,0} & S_{2,1} \\ S_{3,3} & S_{3,0} & S_{3,1} & S_{3,2} \end{bmatrix} \qquad \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,3} & S_{1,0} & S_{1,1} & S_{1,2} \\ S_{2,2} & S_{2,3} & S_{2,0} & S_{2,1} \\ S_{3,1} & S_{3,2} & S_{3,3} & S_{3,0} \end{bmatrix}$$

(a) ShiftRows (b) InvShiftRows

Figure 4.23: Shift Transformation of ShiftRows and InvShiftRows

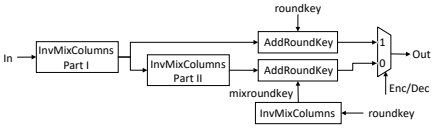
4

SHARED MIXCOLUMNS

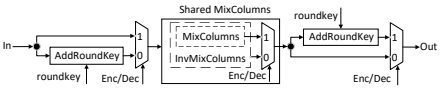
We optimize the *Shared MixColumns* based on the proposed design in paper [44], which shares resources between *MixColumns* and *InvMixColumns*. The design in [44] however requires an additional *InvMixColumns* calculation to rectify the roundkey (see Fig. 4.24a). In contrast, Fig. 4.24b shows that our proposed *Shared MixColumns* combines *MixColumns* and *InvMixColumns*, and reorganizes the sequence of *Addroundkey* and *Shared MixColumns* to avoid performing this additional *InvMixColumns* calculation. This lead to further area improvements.

4.4.3. DESIGN AND IMPLEMENTATION OF PROPOSED LIGHTWEIGHT DOM

DOM was proposed in [21] to protect AES implementations against SCAs. The authors introduced two types of SBOXes: a five-stage SBOX and an eight-stage SBOX. The five-stage SBOX represents an optimized version of the eight-stage SBOX, resulting in a savings of three cycles per round. Hence, overall it is 33 cycles (3 cycles × 11 rounds) faster, which is a significant performance improvement. This improvement comes with only a minor increase in the overall area, from 2.6k Gates to 2.8k Gates, as documented in [21]. Therefore, we chose to start from the five-stage SBOX and integrate it into our optimized design. Note that a 1^{st}-order DOM can be easily scaled into a higher order DOM without redesigning components [21]. Therefore, without loss of generality, we focus on the optimization of the 1^{st}-order DOM. As our proposed low-area design considers both encryption and decryption with shared resources (see Fig. 4.19), the lightweight DOM AES was implemented in the same manner, diverging from the original design that concentrated only on encryption [21]. Fig. 4.25 shows the main part of our lightweight DOM



(a) MixColumns/InvMixColums [44]



(b) Proposed Shared MixColumns

Figure 4.24: Diagram of MixColumns [44] and Proposed Shared Mixcolumns

design, where A_{in}, B_{in} represent the input shares, and A_{out}, B_{out} correspond to the output shares.

As discussed in the background, the independence of d+1 shares within linear modules can be ensured by employing d+1 identical modules. However, the non-linear SBOX module requires to be carefully designed. Our design is based on the lightweight DOM SBOX proposed in [21]. In comparison to that design, our approach shares the resources between encryption and decryption parts using the preprocess and postprocess functions. In addition, we optimize the DOM-indep and DOM-dep multipliers based on our simplified and shared multipliers to further reduce the area. Fig. 4.26 depicts our design of the 1^{st}-order five stages lightweight DOM SBOX, where A_{sin} and B_{sin} denote the input shares, and A_{sout} and B_{sout} denote the output shares. In the figure, $Z_0, Z_1, ..., Z_6$ and $A_{z0}, B_{z0}, ..., A_{z3}, B_{z3}$ are fresh random values of the simplified DOM-indep and DOM-dep multipliers, respectively. The flip-flops with dotted boxes are optional registers that are only necessary in pipelining scenarios. For example, when the data-path is less than 128 bits, the SBOX needs to be reused multiple times within one round, causing the input to change before the round is completed. In this case, the dotted flip-flops are necessary to ensure the design's functional correctness.

Fig. 4.26 also highlights the parts that we improved in red, i.e., the DOM multipliers. Compared to the original DOM multipliers (see [21]), we replaced the normal multipliers with our simplified multipliers (see (4.6)) and shared multipliers (see Fig. 4.22), resulting in a reduction in power and area. In addition, multiplexers are used to select between inputs for the encryption and decryption units.

Fig. 4.27 illustrates our changes made to the 1^{st}-order Dom-dep multiplier [21]; we refer to it as simplified DOM-dep multiplier. In the figure, A_a, B_a, A_b, B_b are the inputs, while A_q and B_q correspond to the outputs. A_z, B_z, and Z_0 are random numbers that used to ensure the independence of shares. In contrast to the design proposed in [21], our proposed simplified DOM-dep multiplier utilizes a simplified version of the DOM-indep multiplier and merges the right two multipliers, resulting in significant area reduction. The simplified DOM-indep multiplier is based on the simplified multiplier shown in Equation (4.6). Equation (4.9) illustrates the expression of our DOM-dep multiplier,

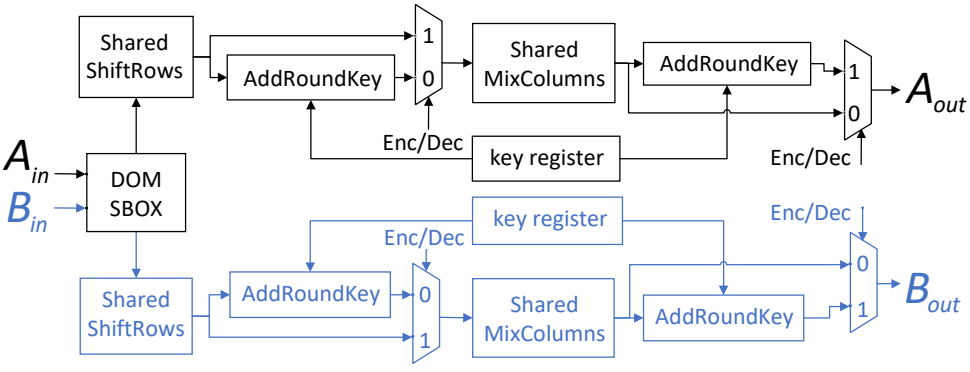


Figure 4.25: Proposed Design for DOM Encryption and Decryption

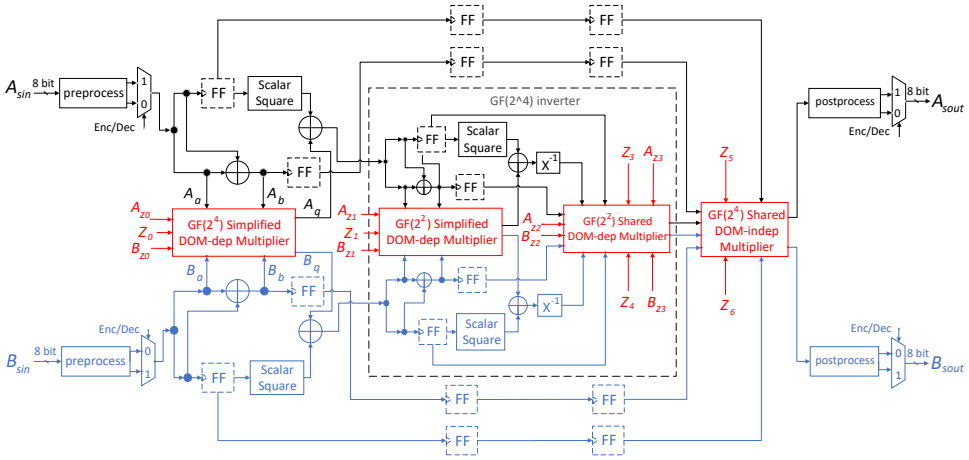
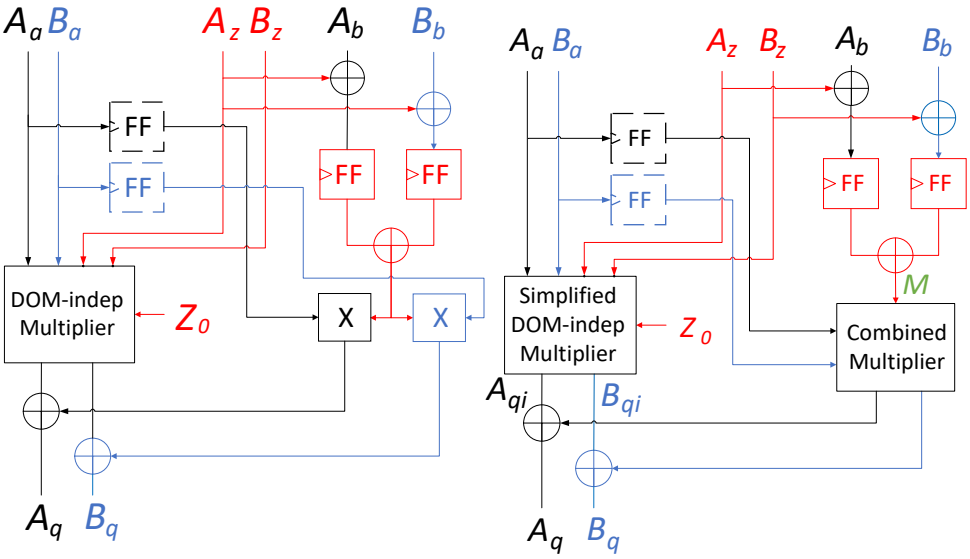


Figure 4.26: Structure of the 1^{st}-order five-stage DOM SBOX Module (modified based on [21])

where M=$(A_b \oplus B_b) \oplus (A_z \oplus B_z)$. We denote that a= $A_a \oplus B_a$, b= $A_b \oplus B_b$, z= $A_z \oplus B_z$, and q= $A_q \oplus B_q$. In the equation, A_{qi} and B_{qi} are the outputs of simplified DOM-indep multipliers. The combined multiplier (see Fig. 4.22) is utilized for the calculation of $(A_a * M \oplus B_a * M)$ to further reduce area.



(a) DOM-dep [21]

(b) Simplified DOM-dep

Figure 4.27: DOM-dep Multiplier [21] and Our Simplified DOM-dep Multiplier

$$
\begin{aligned}
& a * b \\
&= a * (b \oplus z) \oplus a * z \\
&= (A_a \oplus B_a)\left(A_b \oplus B_b \oplus A_z \oplus B_z\right) \oplus (A_a \oplus B_a)(A_z \oplus B_z) \\
&= (A_a * M \oplus B_a * M) \oplus (A_{qi} \oplus B_{qi}) \\
&= (A_a * M \oplus A_{qi}) \oplus (B_a * M \oplus B_{qi}) \\
&= A_q \oplus B_q = q
\end{aligned}
\tag{4.9}
$$

The shared DOM-indep/DOM-dep multiplier modules in Fig. 4.26 are implemented with the simpilified DOM-indep/DOM-dep multipliers (see Fig. 4.27b). We share the common resources between these two multipliers to further reduce the area.

4.4.4. SETUP

For the majority of IoT applications, the maximum payload size for each packet is between 1600 bytes (e.g., NarrowBand-IoT [153]) and 256 megabytes (e.g., Message Queue Telemetry Transport(MQTT) [154]). Taking the lower limit into consideration, we assume that the key will stay the same during the communication session of at least one hundred encrypting/decrypting operations. For that reason, we assumed a fixed key for 100 encryption and decryption operations.

To compare with the start-of-the-art, we reimplemented the state-of-the-art AES design proposed by Davis and Jones [148] and compared it with ours. All designs are synthesized using TSMC CMOS 180 nm technology. The total area and power consumption of each design are evaluated Using Synopsys Design and Power Compiler. We took the same approach with respect to the DOM design originally proposed in [21].

4.4.5. AES PERFORMANCE EVALUATION

In this section we compare our AES SBOX design proposed in Section III.B with the SBOX proposed in [151]. Note that this paper limited itself to only an SBOX implementation. The synthesis results show that the area of our proposed SBOX design is 8.2% lower than the design in [151]. The actual area numbers are 2558 vs 2788 μm^2, respectively.

We compare our complete non-DOM AES design proposed in Section III.B with the design proposed in [148].Tables 4.3 and 4.4 show the results for both designs synthesized at 50 MHz and maximum frequency, respectively. . From the first table, we can see that our 32-bit data-path implementation needs 20% less area than the design proposed in in [148]. Actually, our 128-bit data-path design is comparable in size to their 32-bit data-path design, while being 5x faster. When we look at Table 4.4 we see similar trends. Comparing both 32-bit data-path designs, our design has 14% less area, needs 26% less cycles and can run at 18% higher frequency.

Figs. 4.28a and 4.28b depict the performance per area and performance per power of our proposed AES design and the state-of-the-art design in [148]. The figures show that among our proposed designs, the 128-bit design achieves the highest score in terms of performance per area and performance per power. Our proposed 128-bit, 64-bit, and 32-bit designs surpass the state-of-the-art [148] in terms of performance per area by a factor of 4.90x, 3.02x, and 1.69x, respectively, when operating at 50MHz and by a factor of 4.55x,

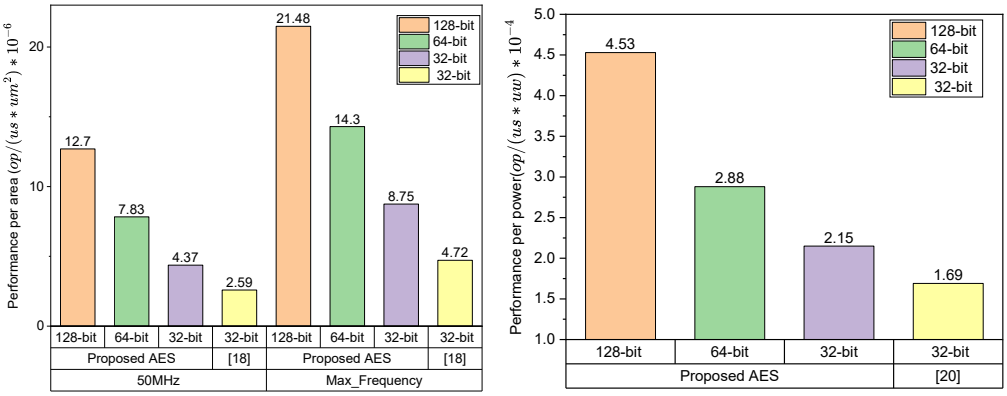
Table 4.3: AES Performance Analysis at 50MHz

Design	Data-path	Freq. (MHz)	Area (μm^2)	Area Ratio	Cycle	Cycle Ratio
[148]	32	50	174156	1	11100	1
Proposed AES	32	50	139415	0.8	8211	0.74
	64	50	151677	0.87	4211	0.38
	128	50	178674	1.03	2211	0.2

Table 4.4: AES Performance Analysis at the Maximum Frequency

Design	Data-path	Freq. (MHz)	Freq. Ratio	Area (μm^2)	Area Ratio	Cycle	Cycle Ratio
[148]	32	103.1	1	196857	1	11100	1
Proposed AES	32	121.9	1.18	169794	0.86	8211	0.74
	64	117.6	1.14	195355	0.99	4211	0.38
	128	112.4	1.09	236553	1.2	2211	0.2

3.03x, and 1.85x, respectively, when operating at the maximum frequency. They also outperform the state-of-the-art design in terms of performance per power by a factor of 2.68x, 1.70x, and 1.27x, respectively. Note that in Fig. 4.28b, only the performance per power for the 50 MHz implementation is displayed, as there were minimal differences observed when compared to the designs operating at their maximum frequencies.



(a) Performance per Area

(b) Performance per Power

Figure 4.28: Comparison vs AES Design in [148]

4.4.6. DOM PERFORMANCE EVALUATION

Table 4.5 shows a comparison of the area between our proposed 1^{st}-order DOM SBOX and the original 1^{st}-order SBOX proposed in [21]. Compared to their design, our eight-stage and five-stage 1^{st}-order DOM SBOX designs achieve an area reduction of 9.9% and 6.9%, respectively.

Table 4.5: Area comparison of DOM SBOX

Design	SBOX Type	Area (μm^2)	Area Ratio
[21] DOM SBOX	eight-stage	19682	1
	five-stage	21196	1.077
Proposed DOM SBOX	eight-stage	17735	0.901
	five-stage	19741	1.003

Although the 128-bit data-path design has the best performance, we have implemented also 64-bit and 32-bit data-path versions. Unfortunately, the authors in [21] focused only on the SBOX and have not evaluated the complete AES design. Nevertheless, to comprehensively assess the influence of these designs on overall performance, our proposed DOM SBOX has been incorporated into all AES configurations, encompassing the 128-bit, 64-bit, and 32-bit versions. Tables 4.6 and 4.7 present their area and latency results for an operating frequency of 50 MHz and their maximum operating frequency, respectively. As we mentioned in the Section 4.4.3, we chose the five-stage SBOX in our designs because it offers a substantial performance improvement with only minor sacrifices in terms of area when compared to the eight-stage SBOX. Consequently, the key expansion process takes 51 (5*10+1) cycles, while the encryption and decryption operations in the 128-bit, 64-bit, and 32-bit data-path designs require 51 (5*10+1), 61 (6*10+1), and 81 (8*10+1) cycles, respectively. As a result, it takes 10251 (51*2*100+51), 12251 (61*2*100+51), and 16251 (81*2*100+51) cycles for these designs to perform 100 encryption/decryption operations. Tables 4.6 and 4.7 demonstrate that the 32-bit design exhibits a better area, whereas the 128-bit design has a higher performance.

Table 4.6: DOM Performance Analysis at 50Mhz

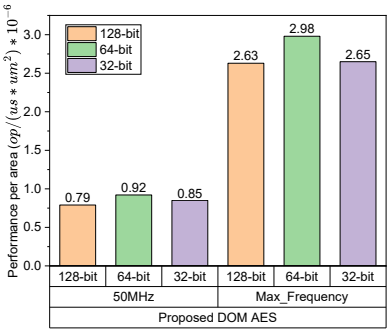
Data-path	Frequency (MHz)	Area (μm^2)	Area Ratio	Cycle	Cycle Ratio
128-bit	50	615172	1	10251	1
64-bit	50	445269	0.72	12251	1.2
32-bit	50	361582	0.59	16251	1.59

Table 4.7: DOM Performance Analysis at the Maximum Frequency

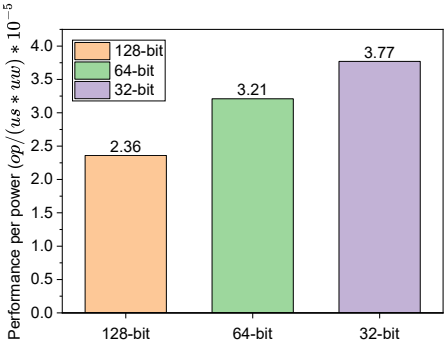
Data-path	Frequency (MHz)	Frequency ratio	Area (μm^2)	Area Ratio	Cycle	Cycle Ratio
128-bit	188.7	1.79	698780	1	10251	1
64-bit	190.8	1.81	522844	0.75	12251	1.2
32-bit	192.3	1.83	446528	0.64	16251	1.59

The DOM SBOX contains a great number of registers and operation, resulting in a high power consumption and area. However, lower data-path designs can significantly reduce area and power consumption by utilizing fewer DOM SBOXes. Fig. 4.29a shows the performance per area comparison of our proposed DOM designs. According to the

figure, the 64-bit design performs better at both 50 MHz and the maximum frequency. Fig. 4.29b illustrates a comparison of the performance per power for our proposed DOM designs, where the 32-bit design outperforms the others.



(a) Performance per Area



(b) Performance per Power

Figure 4.29: Analysis of our 1^{st}-Order DOM AES Designs

5

PRE-SILICON ASSESSMENT METHODS

This chapter focuses on the examination of the pre-silicon leakage assessment approaches. Aiming to specifically develop a mechanism that analyzes and enhances micro-electronic chips against power attacks during the design phase. The exploration begins with a thorough examination of the current landscape of pre-silicon leakage evaluation methodologies, offering a complete survey of the prevailing state-of-the-art approaches. Subsequently, we proceed to Section 5.2, wherein we present a pioneering and inventive methodology that is based on the utilization of Generative Neural Networks (GAN). This method provides an advancement in the pursuit of enhanced security in chip design, and we will explore its mechanics and ramifications. In the concluding section, Section 5.3, we provide a reflective conclusion that summarizes our investigation and emphasizes the possible implications and developments that this technique offers for the field of pre-silicon leakage evaluation in the context of power attacks.

This chapter is partially published on 2023 26th Euromicro Conference on Digital System Design (DSD)[42].

5.1. STATE OF THE ART

The idea of Pre-silicon leakage assessment is not a novel one. There are currently multiple options to evaluate countermeasure designs prior to manufacturing. These options can be divided into three categories: formal verification [75], CAD tools [72] and functional simulation tools [155]. The aim of formal verification-based solutions is to mathematically analyze the leakage of an implementation. Formal verification examples can be found in [75, 76]. In [75] the authors use formal verification to verify hardware masking countermeasures. In [76], the authors present an SMT-solver (SMT stands for satisfiability modulo theories) for software masking countermeasures. Unfortunately, such solutions focus on analyzing randomness created by masks. As a result, they only operate on one type of countermeasure, i.e., masking countermeasures. CAD-based solutions, on the other hand, tend to produce the power behavior of the targeted implementation. An example of a CAD-based solution is provided by Sadhukhan et al. [156] where they examine the leakage using both simulated and hypothetical power traces. Another example suggested by Nahiyan et al [157] is that they reduce the number of power traces needed to evaluate the leakage by improving the signal-to-noise ratio (SNR) algorithm. Unfortunately, creating simulated power traces is a time-consuming operation, and reducing the number of simulated traces cannot validate the protection against real attacks such as CPA, which typically require a large number of traces. Additionally, we see in industry that secure IPs are evaluated with a minimal of 10 million power traces [158]. Therefore, a CAD-based assessment solution would be an interesting solution only if it can generate millions of power traces in a timely manner. Unfortunately, the CAD-based tools are known to be extremely slow and generating as many as 10 million power traces using them is not viable. For example, the solution proposed by Sadhukhan *et al.* [156] takes 5.47 seconds to generate a single trace. Generating as many as 10 million power traces using this technique will take around 633 days. This, of course, is not practical.

To obviate the need of using CAD-based tools for power trace generation, functional simulation tools like RTL-PSC [155] have been introduced. Wherein, the functional simulation of a design is used to carry out pre-silicon leakage assessment. This technique does not generate power trace as it is solely dependent on the switching activity file. This dependence on the switching activity makes this methodology not fit for modelling the design's technological behaviour(e.g., CMOS), which is useful for various countermeasures and is not visible using simply the switching activity. The switching activity also gives a rather ideal picture about the power consumption since it does not model any non-idealities like timing violation. Hence, there is need for a tool that can not only carry out pre-silicon leakage assessment in a reasonable amount of time, but can also accommodate the device's technological behaviour and non-idealities correctly so as to model its leakage robustly.

5.2. GAN-BASED LEAKAGE ASSESSMENT APPROACH

This section provides a description of the proposed approach that is based on Generative Adversarial Networks (GANs). To begin, some background information about GAN is presented. Next, the work that is associated with the several applications that GAN helps enhance is presented here. Finally, the framework that was proposed is broken down,

and the validation results are shown.

5.2.1. GENERATIVE ADVERSARIAL NETWORKS (GANS)

GANs are used to generate new data sets with similar characteristics as the training set, i.e., to approximate the training set's distribution. They consist of two main components, i.e., the generator G and discriminator D (see Figure 5.1. The generator's objective is to generate samples with a similar distribution to the actual dataset distribution. The discriminator's objective is to differentiate between real and fake traces, namely x and $G(z)$. Hence, the training process of a GAN takes place in an adversarial setting wherein the discriminator and the generator play a minimax game. The loss function L can be expressed as follows:

$$\begin{aligned} \min_G \max_D L(G,D) &= \mathbb{E}_{x\sim p(x)}[\log D(x)] \\ &+ \mathbb{E}_{z\sim p(z)}[\log(1-D(G(z)))] \end{aligned} \tag{5.1}$$

This loss function corresponds to the original GAN architecture as proposed by Goodfellow et al. [159]. However, as we want to condition the GAN on a categorical label corresponding to each encryption, the label y (which for example can correspond to the hamming weight/distance of the SBOX output) is also a part of the loss function. This conditional GAN loss function was originally proposed by Mirza and Osindero [160] and can be expressed as:

$$\begin{aligned} \min_G \max_D L(D,G) = & \quad \mathbb{E}_{\boldsymbol{x}\sim \mathrm{p}_{\mathrm{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}\mid \boldsymbol{y})] \\ + & \quad \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z}\mid \boldsymbol{y})))] \end{aligned} \tag{5.2}$$

The non-cooperative game is what makes training GANs hard and unstable. To address this problem, a lot of research has been performed to find better loss functions and normalization techniques. In this work, in addition to the loss function presented in Equation 5.2, we experimented with the Least Squares GAN (LSGAN) [161] and the Wasserstein GAN Gradient penalty (WGAN-GP) loss function [162]. The LSGAN aims at increasing the quality of the generated samples by improving the discriminator's output by not only looking at the binary output decision but also the quality of the generated traces. Note that in the GAN proposed by Mirza and Osindero, the discriminator's purpose is to distinguish between the real and fake samples as shown in Equation 5.2, which is realized using a binary cross entropy loss. In LSGAN, however, we are not only concerned with the binary classification but also with how close or how far the fake traces are from the real ones. This can be seen in the loss function presented in Equation 5.3.

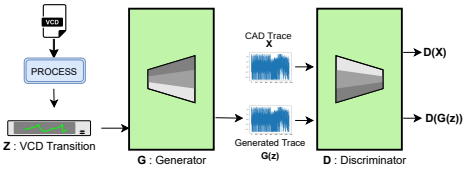


Figure 5.1: Generative Adversarial Network

$$\begin{aligned}\min_D L_{\text{LSGAN}}(D) &= \frac{1}{2}\mathrm{E}_{x\sim p_{\text{dota}}(x)}\left[(D(x)-b)^2\right] \\ &\quad + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_x(z)}\left[(D(G(z))-a)^2\right] \\ \min_G L_{\text{LSGAN}}(G) &= \frac{1}{2}\mathrm{E}_{z\sim p_x(z)}\left[(D(G(z))-c)^2\right]\end{aligned} \tag{5.3}$$

5

During the training of the discriminator, we choose the value of b as 1 to signify real traces and a as 0 to signify generated traces. As the objective of the generator is to create fake traces that look real, we set the value of c during the training of the generator to 1 in order to attempt to fool the discriminator.

The WGAN-GP comes from the family of Wasserstein GANs (WGAN). The objective of WGANs is to minimize the earth mover (EM) distance. The EM distance represents the level of dissimilarity between the distributions of the generated and real traces. Minimizing the EM distance leads to smoother gradients even when the generator outputs unsatisfactory traces. The WGAN's discriminator tries to model a function that approximates the EM distance and not just distinguishing the real samples from the generated ones.

$$L = \underbrace{\mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}[D(\tilde{\boldsymbol{x}})] - \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathbb{P}_r}[D(\boldsymbol{x})]}_{\text{Original critic loss}} + \underbrace{\lambda \mathop{\mathbb{E}}_{\hat{\boldsymbol{x}}\sim\mathbb{P}_{\hat{\boldsymbol{x}}}}\left[\left(\|\nabla_{\hat{\boldsymbol{x}}}D(\hat{\boldsymbol{x}})\|_2 - 1\right)^2\right]}_{\text{gradient penalty}} \tag{5.4}$$

Calculating the EM distance is an intractable problem and the Kantorovich-Rubinstein duality [163] can be used to make the problem simpler. The Kantorovich-Rubinstein duality is used to transform the EM distance minimization problem in order to find a least upper bound. The transformed loss function is required to satisfy K-Lipschitz continuity. This continuity limits how fast a function can change. In the original WGAN paper [164], the Lipschitz constraint is enforced by weight clipping. However, the weight clipping method is extremely sensitive to the clipping value hyperparameter and quite often reduces the network's ability to model complex functions. Instead, WGAN-GP adds a gradient penalty term to enforce the K-Lipschitz continuity as shown in Equation 5.4.

5.2.2. RELATED WORK

Generative deep models have already been applied to numerous applications like images [165], audio [166], video [167] and medical data like ECG [168]. In addition to the

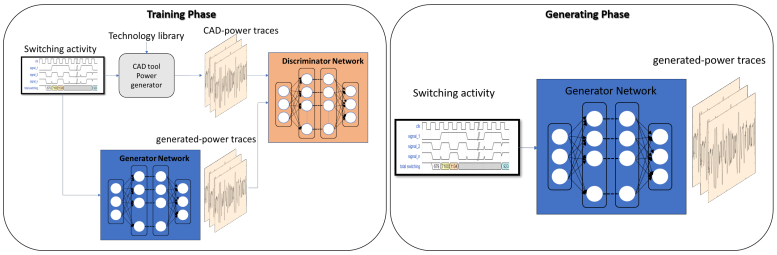


Figure 5.2: Framework Methodology

Generative Adversarial Networks, there are also other methods that could be used to generate fake traces such as *Flow* [169] and *Auto-regressive* [170] based models. *Flow* models use a sequence of invertible transformations to learn the exact data distribution. As exciting as the possibility of exact likelihood computation might seem, *Flow* models usually have manifold times more trainable parameters and require an order of magnitude more processing (in GPU-based platforms) for the training as compared to progressive GAN models [171]. On the other hand, *Auto-regressive* models decompose the likelihood into a product of conditional distributions. However, since the prediction at every timestamp is dependent on all previous predictions, these models are implicitly slow. Hence, for this work, we prefer GANs over other deep generative models as GANs can be trained relatively faster, have a reasonable number of trainable parameters, and have been empirically proven to produce really good quality results. Additionally, taking the power leakage behavior into account, a GAN-based model is ideal since it is quick at inference time and thus can produce power traces swiftly. The existing Electronic Design Automation(EDA) tool-based approaches to generate artificial power traces are extremely slow to generate a large amount of traces. Hence, employing GANs for this task can potentially lead to an immense speed-up.

Generative adversarial networks have recently been introduced in the side channel analysis domain. In 2020, Wang et al. [172] proposed the usage of conditional GANs to enlarge the size of the profiling dataset for carrying out profiled side channel attacks. They use a traditional CGAN architecture with dense layers and train it using the Jensen-Shannon Divergence approach as proposed by Mirza and Osindero [160]. In the author's own words, their study was aimed as a proof of concept and not a robust methodology. Hence their proposed methodology has a few limitations. First, dense layers are best suited for shorter trace lengths, as the number of trainable parameters grows substantially as the input/output size of the GAN's layers increases. Second, they condition the GAN on only a few labels, namely the least significant bit and Hamming weight of the SBOX output. Finally, they only use the Jenson-Shannon Divergence loss and do not experiment with other loss functions. Since the inception of GANs, many architectural changes and loss functions have been proposed that help in alleviating this problem as well as enhancing training stability.

5.2.3. PROPOSED FRAMEWORK

This section presents our proposed framework. We introduce the methodology and subsequently explain each step in detail.

METHODOLOGY

Our proposed framework can be used to evaluate the efficacy and efficiency of countermeasures against side-channel attacks without the need to procure actual power traces for an ASIC design. The methodology to create this framework consists of two major phases as shown in Figure 5.2. In the first phase, the *Training Phase*, the generative adversial neural network is being trained for the targeted circuit using the switching activity from simulation and their corresponding CAD-based power traces. Once the GAN model reaches a desired accuracy, the second phase begins. In this phase, the *Generating Phase*, the trained neural network is used to generate the desired number of power traces to evaluate the security of the design. Here the GAN generates power traces solely from switching activity. In the following subsections, we describe each phase in more detail.

TRAINING PHASE

The first step in training the neural network to generate reliable power traces is to define the GAN's input and classification labels. As input data, we use the switching activity of the targeted circuit. One of the most common representations of the switching activity is the value change dump (VCD) file. VCD files can be generated during RTL or netlist simulations. To quantify the training accuracy, we need labels that represent the actual power traces. Hence, we use gate-level power simulations generated from a CAD tool. The CAD tool uses the switching activity and technology library to generate gate-level power traces. Next we configure the GAN network to be able to generate accurate power traces. Unfortunately, using the switching activity for the labels prevents us from using embedding layers in the GAN's architecture, as embedding layers expect integer numbers as input. The reason for this is that they are implemented as simple look-up tables. Hence, we remove the embedding layer as well as the noise vector and instead just use the VCD transition as a directed input to the GAN (see also Figure 5.1). A similar architectural choice was made by Kumar et al. in their MelGAN architecture [173], wherein they observe little perceptual difference in the generated waveforms when additional noise is fed to the generator. Mathieu et al. [174] and Isola et al. [175] demonstrated the capability of the noise vector's redundancy when using highly informative conditioning. Finally, we train the structured GAN until we reach a desired accuracy level.

GENERATING PHASE

During this phase, the generator component of the trained GAN model (see Figure 5.1) is utilized to generate the power traces that can be used to evaluate the design or countermeasure. It generates these traces using the switching activity which can be obtained from RTL or gate-level simulations. Note that the amount of power traces that need to be generated can be significantly higher than those used in the training phase. Generally, more than 100k traces could be required for the evaluation. Several data sets can be

constructed based on the evaluation method; examples are data sets with a random key and random plaintext, fixed key and fixed plaintext, and fixed key and random plaintext.

EVALUATE FRAMEWORK

The evaluation of the framework is performed through a generalization test. We evaluate if our model can provide reliable power traces. In this step, not only the plaintext varies but also the key value. In each scenario, the quality of the generated power traces is verified through leakage assessment techniques known as evaluation-test (i.e., Correlation Power Attack). Finally, both CAD-based and generated power traces are compared. In addition, for various different implementations, traces can also be generated and evaluated. In this context, the framework can be used to perform a design space exploration to find the most secure solution for a certain algorithm by quickly generating and evaluating traces for VCD files belonging to a different design. We limit our tests to three implementations only, which are: unprotected AES implementation, AES implementation with masking and AES implementation with blinding.

When the framework is completed, designers can generate as many traces as needed. For example, security components used in the industry (i.e., Intellectual Property blocks) require for their vulnerability assessment 10 million, 100 million or 1 billion power traces [158].

FRAMEWORK OPTIMIZATION

Each time the design changes, the generator model must be retrained to be able to generate reliable power traces. Training the generator is the most time-consuming phase of the proposed framework. Luckily, *transfer learning* can be used to reduce the training time. Transfer learning enables retraining of the neural network with much less effort, including the amount of training data required to achieve high accuracy [176]. There are two methods of applying transfer learning, namely feature extraction and fine tuning. In feature extraction, some of the layers in the trained networks are frozen and used for feature extraction, while others are retrained based on the new training data. In fine tuning, the weights of the trained network are used as initialization for the new network. Using the fine tuning technique in our framework, we were able to reduce the required number of traces from 10000 to 1000 to train the GAN for each newly developed countermeasure.

5.2.4. EXPERIMENTAL RESULTS

This section presents the experiments setup, performed experiments and evaluates the obtained results.

SETUP

The different AES designs are simulated in Questasim [177], which is additionally used to generate the VCD files that contain the switching activity. Their corresponding power traces are generated by Synopsys SpyGlass; using the RTL-code of the design and the technology library for the target ASIC design, SpyGlass can generate power traces at gate-level. The VCD files and power traces are used to train the GAN model. The GAN model is implemented with the PyTorch [178] deep learning library and the remaining analysis are performed in Python as well. All the experiments including the training of the GAN

are performed on an Intel i7-10750H CPU running at 2.60GHz and the Nvidia GeForce RTX 2070 GPU.

To verify the accuracy of the framework, we test our findings using three implementations with different keys/plaintext combinations to ensure that our approach works regardless of the input to the target design. As weight initialization (known as transfer learning) for both the Generator and Discriminator, a data set is used containing 20k traces of the unprotected AES implementation based on random keys and plaintext values. Note this is only done once per technology library. Next for the evaluation of the target design, we start by training the generator with only 1k traces. Subsequently, the Generator is used to generate traces based on a VCD belonging to the trained target (e.g., unprotected AES, protected masked SBOX implementation or protected AES implementation based on balancing). The traces are validated against corresponding traces obtained from SpyGlass. Note that the GAN is only trained with random inputs (i.e., for both plaintext and keys).

EVALUATION METRICS AND PERFORMED EXPERIMENTS

In this subsection, we first present the metrics used to evaluate our results. Thereafter we describe to which experiments these metrics have been applied.

Evaluation-style Metric: In evaluation-style testing, power traces are tested using actual side-channel attack scenarios. They show whether the implementations are resistant to these attacks or not. The attacks can be performed in a profiled or unprofiled manner. Examples of profiled side-channel attacks are template-based [52] and deep learning attacks [38]. Examples of unprofiled side-channel attacks are Differential power analysis [9] and correlation power analysis [47]. In this paper, we limit our analysis to CPA as it is one of the most popular unprofiled techniques. Subkeys with highest correlation are most likely the correct key guesses. The results are represented using rank analysis of the correct, also referred to as partial guessing entropy.

Trace equivalency: To compare the similarity between the SpyGlass (referred to as CAD traces) and the GAN traces (referred to as generated traces), certain signal processing metrics like *dynamic time warping* and *power spectral density* can be used. *Dynamic time warping (DTW)* [179] finds an optimal alignment between two unmatched temporal sequences. This optimal alignment or the 'warping path' maps the two sequences such that the distance between them is minimized. The minimum distance can be used as a measure for the similarity between any such two sequences. Similarly, even the *Power spectral density (PSD)* of two signals can be used as a measure for the similarity between them. *Power spectral density (PSD)* is the measure of power distributed across different frequency components that compose a signal [180]. For measured and generated traces to be visually similar, the *DTW* distance should be low in value (the lowest it can be is zero) and the *PSD* should be very similar.

FRAMEWORK EVALUATION

As described previously, we compare the attackability of the generated traces with the CAD-based power traces. To make the comparison fair, we test the Generator's generalization ability on VCD files corresponding to the same AES implementation but with a different key. In addition, different AES implementations with masking and blinding

countermeasures are tested as well. Note that the experiment was performed using the GAN architecture shown in Figure 5.3. Hyperparameter tuning for GANs is much more complex than hyperparameter tuning for other machine learning models since the two-model architecture of GANs does not easily fit into the popular hyperparameter search APIs. Our initial architecture was inspired by a popular GAN implementation [181] and then we randomly searched for optimal hyperparameters. For this work: the batch-size is 100, and the kernel size for the convolutional layers is 8. The idea of using a slightly larger kernel size was inspired from [182], where the author demonstrates that deep learning based power side-channel attacks using a convolutional neural network (CNN) with larger kernel sizes perform much better than CNNs with small kernel sizes. As for the loss function, the visual appearance of the traces as well as the leakage behavior showed minor variations for different loss functions. This minor impact of loss functions on the generated traces is in line with [183], where the authors state that the quality of the GAN generated samples are not substantially dependent on the loss functions.

```
DiscriminatorModel(
  (layers): Sequential(
    (0): Conv1d(1, 48, kernel_size=(8,), stride=(2,))
    (1): LeakyReLU(negative_slope=2.0)
    (2): MaxPool1d(kernel_size=2, stride=2)
    (3): Conv1d(48, 24, kernel_size=(8,), stride=(2,))
    (4): LeakyReLU(negative_slope=2.0)
    (5): MaxPool1d(kernel_size=2, stride=2)
    (6): Conv1d(24, 12, kernel_size=(8,), stride=(2,))
    (7): LeakyReLU(negative_slope=2.0)
    (8): Conv1d(12, 6, kernel_size=(8,), stride=(2,))
    (9): LeakyReLU(negative_slope=2.0)
    (10): MaxPool1d(kernel_size=2, stride=2)
    (11): Conv1d(6, 3, kernel_size=(8,), stride=(2,))
    (12): LeakyReLU(negative_slope=2.0)
    (13): MaxPool1d(kernel_size=2, stride=2)
    (14): Conv1d(3, 1, kernel_size=(8,), stride=(2,))
    (15): LeakyReLU(negative_slope=2.0)
    (16): MaxPool1d(kernel_size=2, stride=2)
  )
)
GeneratorModel(
  (layers): Sequential(
    (0): ConvTranspose1d(1, 48, kernel_size=(8,), stride=(2,))
    (1): LeakyReLU(negative_slope=2.0)
    (2): MaxPool1d(kernel_size=2, stride=2)
    (3): ConvTranspose1d(48, 24, kernel_size=(8,), stride=(2,))
    (4): LeakyReLU(negative_slope=2.0)
    (5): MaxPool1d(kernel_size=2, stride=2)
    (6): ConvTranspose1d(24, 12, kernel_size=(8,), stride=(1,))
    (7): LeakyReLU(negative_slope=2.0)
    (8): ConvTranspose1d(12, 6, kernel_size=(8,), stride=(1,))
    (9): LeakyReLU(negative_slope=2.0)
    (10): ConvTranspose1d(6, 1, kernel_size=(8,), stride=(1,))
    (11): Sigmoid()
  )
)
```

Figure 5.3: GAN architecture

Evaluation-style Metric: The generated traces behave similarly to the measured traces when we look at CPA ranking analysis. For example, in the case of the unprotected implementation, the number of attackable bytes was 16, which equals the 16 attackable bytes when CAD traces were used. Similarly, in the mask-protected implementation, the number of the attackable bytes was zero for the GAN generated traces, which matches the results obtained from the CAD traces. The same results have been observed for the balance-protected implementation. We further go into the depths of the ranking analysis by examining the rank behavior of the correct key for each of the three implementations as shown in Figures 5.4, 5.5, and 5.6. The figures show that for different implementations the rank analysis trends are similar both for generated and CAD traces. This shows that the generative model can be used with a wide range of VCD inputs. We believe that the GAN model is able to extract the relevant signals for the leakage effectively from the VCD and filter out the unneeded signals. As a result, the generated traces can be used for

evaluating countermeasures.

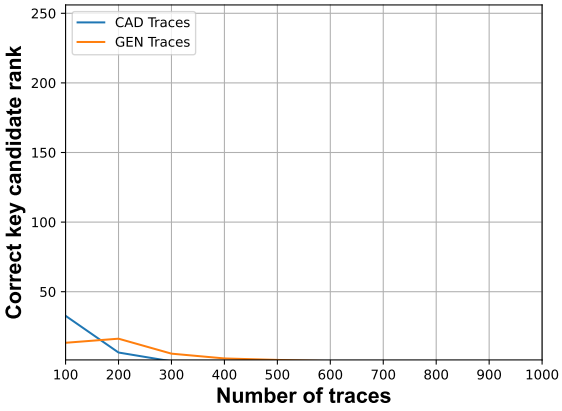


Figure 5.4: CPA Results of Unprotected AES Implementation

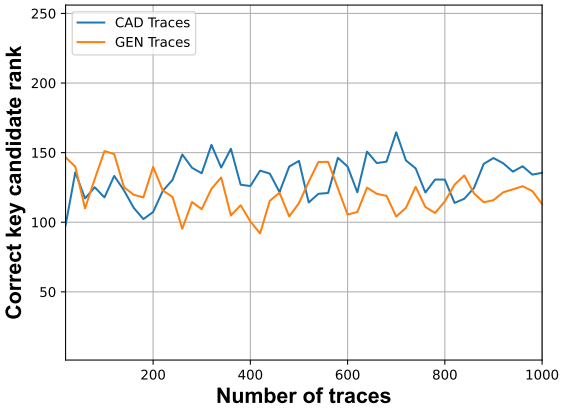


Figure 5.5: CPA Results of Masked AES Implementation

Trace equivalency: The distance between GAN and CAD traces was calculated using FastDTW [184] and we used Euclidean distance as the distance measure for DTW. We obtained values around 0.5 and noticed that the GAN provide good results when the DTW value is between 2.0 and 0.3 . Both these scores test to the similarity of the CAD and generated traces. The Power spectral density (PSD) of CAD and generated traces were also almost identical, as shown in Figure 5.7a. Note however that the distance between the PSD of CAD and VCD traces is much larger (see Figure 5.7b). This shows that using traces obtained from VCD only is not as accurate as generated from the GAN, even

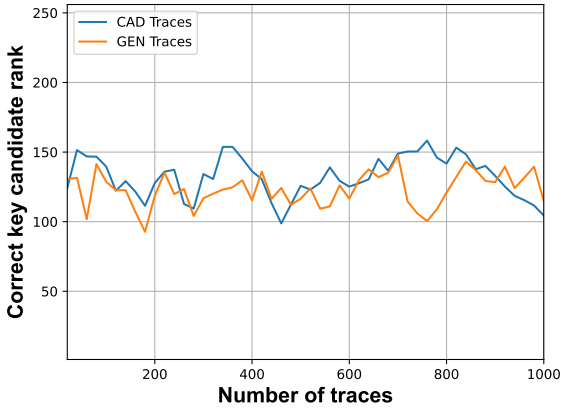
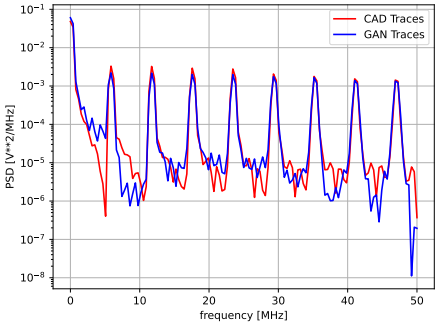
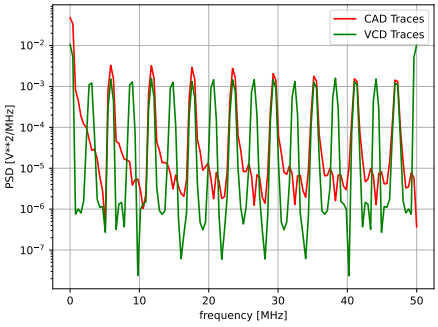


Figure 5.6: CPA Results of Balanced AES Implementation

though the GAN uses the VCD as input.



(a) PSD of CAD Versus GAN Traces



(b) PSD of CAD Versus VCD Traces

5.2.5. COMPARISON TO STATE OF THE ART

Because we are only interested in hardware implementations, we have excluded from the comparison any trace generation methods that are based on formal verification, any methods that target software implementations at the instruction level, and any analytical-based approaches such as [185]. Methods that did not provide a discussion of the amount of time necessary to gather the traces were also excluded [186]. Instead, we compare our method solely to state-of-the-art pre-silicon techniques. We compare our approach to one method that is based on the layout level and three other methods that are based on the gate level, one of which is an approach based on artificial intelligence. For the purpose of this comparison, two criteria were used: first, the quality of the traces, which was

evaluated depending on the level at which the traces were created (i.e., the layout, gate, or RTL level), and second, the speed at which those traces were generated. We presumed that traces produced at lower levels had a higher quality; nevertheless, they require more time to generate [187]. In terms of quality, we were successful in generating traces at the gate level, which is better than the RTL level and allowed us to obtain comparable results in terms of attackability. When it comes to speed, our proposed approach exceeded the state-of-the-art by 120 times as shown in Table 5.1.

We are under the impression that our approach is likewise capable of generating layout-based traces at a quicker rate than the conventional way. However, because we need to train the GAN using several thousands of layout-based traces and because the generation of those traces takes a significant amount of time, this may be considered a shortcoming of the present technique. Still has the potential to be quicker than the standard approach (i.e., CAD tool) when the number of traces needed is in the hundreds of thousands or more.

Table 5.1: Comparison with State of the Art

Pre-silicon	Prepartion/Training time	Time (10k traces)	Trace Type
Gate-level (GAN)	35 minutes	$\sim 30 seconds$	GAN generated
Gate-level (Transfer-learning)	15 minutes	$\sim 30 seconds$	GAN generated
Gate-level PRIMAL [188]	3 hours	-	CNN generated
RTL-Level (RTL-PSC) [155]	N/A	1 hours	Simulated
Gate-Level (PATCH) [189]	N/A	10 hours	Simulated
Layout (RTL-PSC) [155]	N/A	days	Simulated

5.3. CONCLUSION

This study proposes a framework that is able to quickly generate traces that are very comparable to CAD-based traces. Our generative models were not only able to generate visually indistinguishable power traces from the training set but were also able to learn the characteristics of the VCD transition array. According to our experiments, only a few thousand CAD traces are required to train GAN models using transfer learning in order to produce high-quality power traces by simply generating them from the switching activity. As a result, we significantly improved the performance. The evaluation study was carried out on various hardware implementations of the AES algorithm. However, our framework is designed so that it can be extended to other cryptographic algorithms. However, since the trace length increases significantly for asymmetric algorithms like RSA or even software implementation of AES, a progressively growing convolutional architecture [190] like that proposed by Harrold et al. [191] must be used.

6

CONCLUSION

This chapter provides a comprehensive review that outlines the successes and milestones achieved throughout the course of this dissertation. Additionally, we shine a light on potential areas for future research. In Section 6.1, the essential results gained from the substantial research given in this thesis are encapsulated. In the subsequent section, Section 6.2, an in-depth analysis is conducted to provide potential avenues for future research. These suggestions highlight possible advancements and areas of investigation that can further expand upon the groundwork established in this study.

6.1. SUMMARY

- **Chapter 1: Introduction** This chapter presents an exposition on the imperative nature of security and secure solutions in countering side channel attacks. It begins with identifying the hazard or risk of hardware vulnerabilities, which is evident from the 2019 DELL report indicating that 63% of organizations have experienced data breaches as a result of hardware vulnerabilities. Additionally, the chapter elaborated on how side channels, which are undetectable, could potentially be the silent end of all hardware attacks and how much simpler they will become as technology develops. Following a discussion of hardware vulnerabilities, the chapter delineated the opportunities and challenges within the domain of side channel analysis. It restricted its discussion to three specific areas: protection measures utilizing countermeasures, validation measures employing post-silicon techniques, and prevention measures utilizing pre-silicon techniques. The chapter then addressed the research topics, which center on the response to the central question, "Which side channel attack protection strategies are the most effective?" In order to address this inquiry, the chapter primarily concentrated on three facets: identifying the side channel that attracts attackers the most; devising strategies to fortify the countermeasures; and determining the most effective method for validating these countermeasures during the design phase. The chapter then elaborated on the principal contributions: 1) Analysis and Investigation of Processing Techniques on Power Side Channels; 2) Analysis and Investigation of Thermal Side Channels; 3) Analysis and Investigation of Time Side Channel Attacks; 4) Development of Symmetric and Asymmetric Based Countermeasures; 5) Development of Lightweight Based Countermeasures; and 7) An Artificial Intelligence-Based Leakage Assessment Technique.

- **Chapter 2: Background** In this chapter, we will discuss the essential background information required for this thesis. The chapter begins by providing an explanation of the cryptographic algorithms that were researched for the thesis. These algorithms include Advanced Encryption Standard (AES), which is an asymmetric algorithm; RSA, which is a method that is lightweight; and GIFT. Next, the chapter discusses side channel attacks, which it organizes into two categories: class profile attacks, which enable an adversary to analyze the device target prior to the attack, and class non-profile attacks, which are attacks that are carried out without an earlier analysis of the device target. Both categories are explained in detail. Following that, the chapter presents a classification and literature review of several countermeasures that can protect against power attacks. Finally, the chapter illustrates the present leakage assessment's styles.

- **Chapter 3: Side Channel Analysis** This chapter presents a comprehensive analysis of three distinct side channel attacks that exploit various forms of information leakage, specifically power usage, temporal variations, and thermal emissions. The initial section provides an explanation for the specific selection of these three leakages. The selection is determined by factors such as powerfulness, simplicity, and remote accessibility. Power-based attacks are often regarded as the most powerful and simple method; however, they typically necessitate physical access to the targeted device. Time-based attacks come next in terms of powerfulness and ease of use, while they surpass them in terms of accessibility. Thermal emission attacks are ultimately

selected due to their recognition as a potential hazard, as the monitored equipment utilizes this channel to prevent overheating. The subsequent chapter provides a comprehensive explanation of the attack methodologies employed in every type of attack. Power attacks encompassed a range of attack techniques, including both profiled and non-profiled methods. These attacks were executed against various implementations, covering both symmetric and asymmetric systems. The time-based attacks were conducted utilizing a profiled technique and targeted a lightweight implementation. The thermal attacks were ultimately executed using non-profiled methods and targeted asymmetric entities.

- **Chapter 4: Countermeasures** This chapter presents a comprehensive analysis of four countermeasures that have been developed. The countermeasures implemented cover a diverse array of algorithms, including AES, RSA, and GIFT, and were applicable to various sorts of applications, including lightweight ones. The initial countermeasure devised involves the utilization of a neural network-based Advanced Encryption Standard (AES) implementation. The primary objective of this countermeasure is to obfuscate the attacker by introducing unpredictable variations in power consumption. The second countermeasure that has been devised for asymmetric algorithms. The objective of this countermeasure is to mitigate the leakage by equitably distributing power consumption across all operations. The development of the third algorithm was undertaken with the purpose of providing a lightweight countermeasure specifically designed for symmetric algorithms. The concept behind these countermeasures is the integration of both randomization and balancing techniques. This is achieved by replicating two different instances of the operation, such as the SBOX, with the purpose of ensuring that the outcomes of these two operations exhibit balanced power behaviors. The fourth countermeasure entails the utilization of an efficient implementation of a state-of-the-art countermeasure known as DOM. The countermeasure employed in this study incorporates optimization approaches such as key expansion bypassing, resource sharing, and meticulous module optimizations.

- **Chapter 5: Pre-silicon Leakage Assessment Methods** The third and last application domain that we studied is Bioinformatics. The knowledge of bioinformatics is used in a wide range of applications. Food industries are leveraging this knowledge for food profiling which is an essential step in any food monitoring system. Food profilers work on massive data structures and incur considerable data movement for a real-time monitoring system. We translated the problem of food profiling into hyperdimensional computing representation, which makes it easy to be implemented using CIM. Based on that, we proposed our accelerator. We synthesized the required hardware for our accelerator using UMC's 65nm library by considering an accurate PCM model. Our evaluations demonstrate that our CIM-based implementation achieves a (1) throughput improvement of 192× and 724× and (2) memory reduction of 36× and 33× compared to two state-of-the-art profilers (and).

6.2. OUTLOOK

In this section, we offer a number of insightful recommendations with the goal of enhancing and expanding upon the topics that were discussed in this thesis. In the following discussion, we will give these proposals, which were derived from our in-depth examination and comprehension of the topic.

- **Post quantum Countermeasures development and assessment:** Traditional cryptography, which is still in use today, is based on the difficulty of solving mathematical problems such as the integer factorization issue and the discrete logarithm problem. Both of these difficulties are examples of challenges that can be used to encrypt and decrypt messages. However, these problems are believed to be solvable by quantum computers in polynomial time. Post-quantum cryptography is a field of cryptography that aims to develop algorithms that are resistant to attacks by quantum computers. Lattice problems, code-based problems, and multivariate problems are some examples of the kinds of mathematical problems that form the foundation of post-quantum algorithms. Similar to classical cryptography, post-quantum cryptography can also be vulnerable to power-side channel attacks. The HQC code-based KEM implementation was the target of one of the first power side-channel attacks on a post-quantum algorithm [3]. Power attacks have also successfully attacked the implementation of other algorithms, including multivariate-based [4] and lattice-based [5] algorithms. In the course of our research, we did not incorporate post-quntum as a component of the analysis. Therefore, it is necessary to build a model for assessing leakage of post-quantum countermeasure implementation in a way that is both more quick, efficient, and accurate to prevent possible attacks.

- **Leakage assessment for other channels:** In Chapter 4, our research delved into the exploration of three distinct side channels. We identified these leakage channels as having significant implications for the security of micro-electronic devices. Despite this, our subsequent chapters predominantly concentrated on a solitary leakage channel, primarily due to its elevated potential for posing hardware-level vulnerabilities. Nevertheless, it remains our conviction that the examination of other such channels should be pursued with equal diligence. Therefore, as part of our future work, we intend to extend our evaluation to these additional channels, aiming to develop pre-silicon assessment techniques specifically tailored to them. This comprehensive approach will further enhance our understanding of the security landscape in the realm of micro-electronic devices.

- **Post-silicon leakage assessment method** In the realm of semiconductor device development, the examination of pre-silicon and post-silicon leakage is a crucial and ever-evolving area of study. Its overall goal is to establish strong defenses against power-based assaults, so assuring the integrity and safety of these vital components of modern technology. Although considerable progress has been made in understanding and correcting pre-silicon leakage, there is still a discernible attention gap between pre-silicon and post-silicon assessment methodologies.

  A significant obstacle is presented by the limited amount of research that has been done to investigate the development of post-silicon techniques. Because of this short-

coming, the sector must continue to rely on conventional techniques of assessment, which frequently appear to be prohibitively expensive and time-consuming. As a consequence of this, several chip manufacturers are contemplating skipping or significantly reducing the amount of post-silicon leakage evaluation they perform in an effort to cut production costs and speed up time-to-market. This development, on the other hand, is certainly risky because it may result in the existence of vulnerabilities that were not previously present in semiconductor devices.

Individuals who rely on these chips in a variety of applications, ranging from critical infrastructure to mobile devices like smartphones and tablets, could have their privacy and security compromised as a result of these hidden flaws. The protection of users' data privacy and security has become of the utmost importance as the spread of technology continues to touch every facet of our life.

Because of this, our dedication to finding a solution to this problem should be unyielding. It is imperative that we spend not only in research and development but also in the process of making post-silicon leakage evaluation tools more readily available and competitively priced for chip makers. Doing so will allow us to lay a firm foundation for a future in which chip devices will be better protected from power-based threats, which will eventually ensure that the privacy and security of users will remain uncompromised and unaltered. This coordinated effort will spur innovation, boost trust, and move us into a future in which technology can flourish without putting the very people it serves in risk.

BIBLIOGRAPHY

[1] A. Shinde. *Introduction to Cyber Security: Guide to the World of Cyber Security*. Notion Press, 2021. ISBN: 9781637816431. URL: https://books.google.nl/books?id=VLEcEAAAQBAJ.

[2] F. B. of Investigation (FBI). *Internet Crime Report 2021*. 2021. URL: https://www.ic3.gov/Media/PDF/AnnualReport/2021_IC3Report.pdf (visited on 10/02/2023).

[3] W. E. Forum. *Why we need global rules to crack down on cybercrime*. 2023. URL: https://www.weforum.org/agenda/2023/01/global-rules-crack-down-cybercrime/ (visited on 10/02/2023).

[4] D. Technologies. *The Reality of Hardware-level Security: Companies Need It and They Need It Now*. 2019. URL: https://www.dell.com/en-us/blog/reality-of-hardware-level-security-companies-need-it-now/ (visited on 10/02/2023).

[5] S. Atefi, A. Sivagnanam, A. Ayman, J. Grossklags, and A. Laszka. "The Benefits of Vulnerability Discovery and Bug Bounty Programs: Case Studies of Chromium and Firefox". In: *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*. Ed. by Y. Ding, J. Tang, J. F. Sequeda, L. Aroyo, C. Castillo, and G. Houben. ACM, 2023, pp. 2209–2219. DOI: 10.1145/3543507.3583352. URL: https://doi.org/10.1145/3543507.3583352.

[6] M. Potkonjak, G. Qu, F. Koushanfar, and C. Chang. "20 Years of research on intellectual property protection". In: *IEEE International Symposium on Circuits and Systems, ISCAS 2017, Baltimore, MD, USA, May 28-31, 2017*. IEEE, 2017, pp. 1–4. DOI: 10.1109/ISCAS.2017.8050602. URL: https://doi.org/10.1109/ISCAS.2017.8050602.

[7] J. Francq and F. Frick. "Introduction to hardware trojan detection methods". In: *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, DATE 2015, Grenoble, France, March 9-13, 2015*. Ed. by W. Nebel and D. Atienza. ACM, 2015, pp. 770–775. URL: http://dl.acm.org/citation.cfm?id=2755929.

[8] Y. Zhou and D. Feng. "Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testing". In: *IACR Cryptol. ePrint Arch.* (2005), p. 388. URL: http://eprint.iacr.org/2005/388.

[9] P. C. Kocher, J. Jaffe, and B. Jun. “Differential Power Analysis”. In: *Advances in Cryptology - CRYPTO ’99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*. Ed. by M. J. Wiener. Vol. 1666. Lecture Notes in Computer Science. Springer, 1999, pp. 388–397. DOI: 10.1007/3-540-48405-1_25. URL: https://doi.org/10.1007/3-540-48405-1%5C_25.

[10] A. Prout, W. Arcand, D. Bestor, B. Bergeron, C. Byun, V. Gadepally, M. Houle, M. Hubbell, M. Jones, A. Klein, P. Michaleas, L. Milechin, J. Mullen, A. Rosa, S. Samsi, C. Yee, A. Reuther, and J. Kepner. “Measuring the Impact of Spectre and Meltdown”. In: *CoRR* abs/1807.08703 (2018). arXiv: 1807.08703. URL: http://arxiv.org/abs/1807.08703.

[11] Satista. *Global vulnerable devices due to Meltdown and Spectre 2018*. 2018. URL: https://www.statista.com/statistics/800230/worldwide-meltdown-spectre-vulnerable-devices-by-type/ (visited on 10/02/2023).

[12] J. Daemen and V. Rijmen. “The Block Cipher Rijndael”. In: *Smart Card Research and Applications, This International Conference, CARDIS ’98, Louvain-la-Neuve, Belgium, September 14-16, 1998, Proceedings*. Ed. by J. Quisquater and B. Schneier. Vol. 1820. Lecture Notes in Computer Science. Springer, 1998, pp. 277–284. DOI: 10.1007/10721064_26. URL: https://doi.org/10.1007/10721064%5C_26.

[13] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim, and Y. Todo. “GIFT: A Small Present - Towards Reaching the Limit of Lightweight Encryption”. In: *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*. Ed. by W. Fischer and N. Homma. Vol. 10529. Lecture Notes in Computer Science. Springer, 2017, pp. 321–345. DOI: 10.1007/978-3-319-66787-4_16. URL: https://doi.org/10.1007/978-3-319-66787-4%5C_16.

[14] R. L. Rivest, A. Shamir, and L. M. Adleman. “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”. In: *Commun. ACM* 21.2 (1978), pp. 120–126. DOI: 10.1145/359340.359342. URL: https://doi.org/10.1145/359340.359342.

[15] V. S. Miller. “Use of Elliptic Curves in Cryptography”. In: *Advances in Cryptology - CRYPTO ’85, Santa Barbara, California, USA, August 18-22, 1985, Proceedings*. Ed. by H. C. Williams. Vol. 218. Lecture Notes in Computer Science. Springer, 1985, pp. 417–426. DOI: 10.1007/3-540-39799-X_31. URL: https://doi.org/10.1007/3-540-39799-X%5C_31.

[16] Z. A. Al-Odat, A. Abbas, and S. U. Khan. “Randomness Analyses of the Secure Hash Algorithms, SHA-1, SHA-2 and Modified SHA”. In: *International Conference on Frontiers of Information Technology, FIT 2019, Islamabad, Pakistan, December 16-18, 2019*. IEEE, 2019, pp. 316–321. DOI: 10.1109/FIT47737.2019.00066. URL: https://doi.org/10.1109/FIT47737.2019.00066.

[17] "MD5 Hash Function". In: *Encyclopedia of Cryptography and Security, 2nd Ed.* Ed. by H. C. A. van Tilborg and S. Jajodia. Springer, 2011, p. 771. DOI: 10.1007/978-1-4419-5906-5_1197. URL: https://doi.org/10.1007/978-1-4419-5906-5%5C_1197.

[18] P. C. Kocher. "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems". In: *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*. Ed. by N. Koblitz. Vol. 1109. Lecture Notes in Computer Science. Springer, 1996, pp. 104–113. DOI: 10.1007/3-540-68697-5_9. URL: https://doi.org/10.1007/3-540-68697-5%5C_9.

[19] A. Aljuffri, P. Venkatachalam, C. Reinbrecht, S. Hamdioui, and M. Taouil. "S-NET: A Confusion Based Countermeasure Against Power Attacks for SBOX". In: *Embedded Computer Systems: Architectures, Modeling, and Simulation - 20th International Conference, SAMOS 2020, Samos, Greece, July 5-9, 2020, Proceedings*. Ed. by A. Orailoglu, M. Jung, and M. Reichenbach. Vol. 12471. Lecture Notes in Computer Science. Springer, 2020, pp. 295–307. DOI: 10.1007/978-3-030-60939-9_20. URL: https://doi.org/10.1007/978-3-030-60939-9%5C_20.

[20] A. Gornik, A. Moradi, J. Oehm, and C. Paar. "A Hardware-Based Countermeasure to Reduce Side-Channel Leakage: Design, Implementation, and Evaluation". In: *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 34.8 (2015), pp. 1308–1319. DOI: 10.1109/TCAD.2015.2423274. URL: https://doi.org/10.1109/TCAD.2015.2423274.

[21] H. Groß, S. Mangard, and T. Korak. "Domain-Oriented Masking: Compact Masked Hardware Implementations with Arbitrary Protection Order". In: *IACR Cryptol. ePrint Arch.* (2016), p. 486. URL: http://eprint.iacr.org/2016/486.

[22] A. Aljuffri, C. Reinbrecht, S. Hamdioui, and M. Taouil. "Impact of Data Pre-Processing Techniques on Deep Learning Based Power Attacks". In: *16th International Conference on Design & Technology of Integrated Systems in Nanoscale Era, DTIS 2021, Montpellier, France, June 28-30, 2021*. IEEE, 2021, pp. 1–6. DOI: 10.1109/DTIS53253.2021.9505051. URL: https://doi.org/10.1109/DTIS53253.2021.9505051.

[23] Z. Martinasek, J. Hajny, and L. Malina. "Optimization of Power Analysis Using Neural Network". In: *Smart Card Research and Advanced Applications - 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised Selected Papers*. Ed. by A. Francillon and P. Rohatgi. Vol. 8419. Lecture Notes in Computer Science. Springer, 2013, pp. 94–107. DOI: 10.1007/978-3-319-08302-5_7. URL: https://doi.org/10.1007/978-3-319-08302-5%5C_7.

[24] E. Cagli, C. Dumas, and E. Prouff. "Convolutional Neural Networks with Data Augmentation Against Jitter-Based Countermeasures - Profiling Attacks Without Pre-processing". In: *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*. Ed. by W. Fischer and N. Homma. Vol. 10529. Lecture Notes in Computer

Science. Springer, 2017, pp. 45–68. DOI: 10.1007/978-3-319-66787-4_3. URL: https://doi.org/10.1007/978-3-319-66787-4%5C_3.

[25] N. Debande, Y. Souissi, M. A. Elaabid, S. Guilley, and J. Danger. "Wavelet transform based pre-processing for side channel analysis". In: *45th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 2012, Workshops Proceedings, Vancouver, BC, Canada, December 1-5, 2012*. IEEE Computer Society, 2012, pp. 32–38. DOI: 10.1109/MICROW.2012.15. URL: https://doi.org/10.1109/MICROW.2012.15.

[26] P. Saravanan, P. Kalpana, V. Prcethisri, and V. Sneha. "Power analysis attack using neural networks with wavelet transform as pre-processor". In: *18th International Symposium on VLSI Design and Test, VDAT 2014, Coimbatore, India, July 16-18, 2014*. IEEE, 2014, pp. 1–6. DOI: 10.1109/ISVDAT.2014.6881059. URL: https://doi.org/10.1109/ISVDAT.2014.6881059.

[27] N. Noreen, S. Palaniappan, A. Qayyum, I. Ahmad, M. Imran, and M. Shoaib. "A Deep Learning Model Based on Concatenation Approach for the Diagnosis of Brain Tumor". In: *IEEE Access* 8 (2020), pp. 55135–55144. DOI: 10.1109/ACCESS.2020.2978629. URL: https://doi.org/10.1109/ACCESS.2020.2978629.

[28] L. Wu and S. Picek. "Remove Some Noise: On Pre-processing of Side-channel Measurements with Autoencoders". In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2020.4 (2020), pp. 389–415. DOI: 10.13154/tches.v2020.i4.389-415. URL: https://doi.org/10.13154/tches.v2020.i4.389-415.

[29] L. Macıas-Garcıa, J. M. Luna-Romera, J. Garcıa-Gutiérrez, M. Martınez-Ballesteros, J. C. R. Santos, and R. González-Cámpora. "A study of the suitability of autoencoders for preprocessing data in breast cancer experimentation". In: *J. Biomed. Informatics* 72 (2017), pp. 33–44. DOI: 10.1016/j.jbi.2017.06.020. URL: https://doi.org/10.1016/j.jbi.2017.06.020.

[30] C. Reinbrecht, A. Aljuffri, S. Hamdioui, M. Taouil, and J. Sepúlveda. "GRINCH: A Cache Attack against GIFT Lightweight Cipher". In: *Design, Automation & Test in Europe Conference & Exhibition, DATE 2021, Grenoble, France, February 1-5, 2021*. IEEE, 2021, pp. 549–554. DOI: 10.23919/DATE51398.2021.9474201. URL: https://doi.org/10.23919/DATE51398.2021.9474201.

[31] D. J. Bernstein. *Cache Timing Attacks on AES*. Apr. 2005. URL: https://cr.yp.to/antiforgery/cachetiming-20050414.pdf (visited on 12/31/2016).

[32] D. A. Osvik, A. Shamir, and E. Tromer. "Cache Attacks and Countermeasures: The Case of AES". In: *Topics in Cryptology - CT-RSA 2006, The Cryptographers' Track at the RSA Conference 2006, San Jose, CA, USA, February 13-17, 2006, Proceedings*. Ed. by D. Pointcheval. Vol. 3860. Lecture Notes in Computer Science. Springer, 2006, pp. 1–20. DOI: 10.1007/11605805_1. URL: https://doi.org/10.1007/11605805%5C_1.

[33] O. Aciiçmez and Ç. K. Koç. "Trace-Driven Cache Attacks on AES (Short Paper)". In: *Information and Communications Security, 8th International Conference, ICICS 2006, Raleigh, NC, USA, December 4-7, 2006, Proceedings*. Ed. by P. Ning, S. Qing, and N. Li. Vol. 4307. Lecture Notes in Computer Science. Springer, 2006, pp. 112–121. DOI: 10.1007/11935308_9. URL: https://doi.org/10.1007/11935308%5C_9.

[34] A. Aljuffri, M. Zwalua, C. R. W. Reinbrecht, S. Hamdioui, and M. Taouil. "Applying Thermal Side-Channel Attacks on Asymmetric Cryptography". In: *IEEE Trans. Very Large Scale Integr. Syst.* 29.11 (2021), pp. 1930–1942. DOI: 10.1109/TVLSI.2021.3111407. URL: https://doi.org/10.1109/TVLSI.2021.3111407.

[35] R. J. Masti, D. Rai, A. Ranganathan, C. Müller, L. Thiele, and S. Capkun. "Thermal Covert Channels on Multi-core Platforms". In: *24th USENIX Security Symposium, USENIX Security 15, Washington, D.C., USA, August 12-14, 2015*. Ed. by J. Jung and T. Holz. USENIX Association, 2015, pp. 865–880. URL: https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/masti.

[36] A. Aljuffri, C. Reinbrecht, S. Hamdioui, and M. Taouil. "Multi-Bit Blinding: A Countermeasure for RSA Against Side Channel Attacks". In: *39th IEEE VLSI Test Symposium, VTS 2021, San Diego, CA, USA, April 25-28, 2021*. IEEE, 2021, pp. 1–6. DOI: 10.1109/VTS50974.2021.9441035. URL: https://doi.org/10.1109/VTS50974.2021.9441035.

[37] M. Joye and S. Yen. "The Montgomery Powering Ladder". In: *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*. Ed. by B. S. K. Jr., Ç. K. Koç, and C. Paar. Vol. 2523. Lecture Notes in Computer Science. Springer, 2002, pp. 291–302. DOI: 10.1007/3-540-36400-5_22. URL: https://doi.org/10.1007/3-540-36400-5%5C_22.

[38] H. Maghrebi, T. Portigliatti, and E. Prouff. "Breaking Cryptographic Implementations Using Deep Learning Techniques". In: *Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings*. Ed. by C. Carlet, M. A. Hasan, and V. Saraswat. Vol. 10076. Lecture Notes in Computer Science. Springer, 2016, pp. 3–26. DOI: 10.1007/978-3-319-49445-6_1. URL: https://doi.org/10.1007/978-3-319-49445-6%5C_1.

[39] G. Perin and L. Chmielewski. "A Semi-Parametric Approach for Side-Channel Attacks on Protected RSA Implementations". In: *Smart Card Research and Advanced Applications - 14th International Conference, CARDIS 2015, Bochum, Germany, November 4-6, 2015. Revised Selected Papers*. Ed. by N. Homma and M. Medwed. Vol. 9514. Lecture Notes in Computer Science. Springer, 2015, pp. 34–53. DOI: 10.1007/978-3-319-31271-2_3. URL: https://doi.org/10.1007/978-3-319-31271-2%5C_3.

[40] A. Aljuffri, R. Huang, S. Hamdioui, K. Ma, and M. Taouil. “Securing an Efficient Lightweight AES Accelerator”. In: *2023 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. 2023.

[41] A. Aljuffri, C. Reinbrecht, S. Hamdioui, M. Taouil, and J. Sepúlveda. “Balanced Dual-Mask Protection Scheme for GIFT Cipher Against Power Attacks”. In: *2022 IEEE 40th VLSI Test Symposium (VTS)*. 2022, pp. 1–6. DOI: 10.1109/VTS52500.2021.9794230.

[42] A. Aljuffri, M. Saxena, C. R. W. Reinbrecht, S. Hamdioui, and M. Taouil. “A Pre-Silicon Power Leakage Assessment Based on Generative Adversarial Networks”. In: *2023 26th Euromicro Conference on Digital System Design (DSD)*. 2023.

[43] M. Taouil, A. Aljuffri, and S. Hamdioui. “Power Side Channel Attacks: Where Are We Standing?” In: *16th International Conference on Design & Technology of Integrated Systems in Nanoscale Era, DTIS 2021, Montpellier, France, June 28-30, 2021*. IEEE, 2021, pp. 1–6. DOI: 10.1109/DTIS53253.2021.9505075. URL: https://doi.org/10.1109/DTIS53253.2021.9505075.

[44] X. Zhang and K. K. Parhi. “High-speed VLSI architectures for the AES algorithm”. In: *IEEE Trans. Very Large Scale Integr. Syst.* 12.9 (2004), pp. 957–967. DOI: 10.1109/TVLSI.2004.832943. URL: https://doi.org/10.1109/TVLSI.2004.832943.

[45] W. Diffie and M. E. Hellman. “New directions in cryptography”. In: *IEEE Trans. Inf. Theory* 22.6 (1976), pp. 644–654. DOI: 10.1109/TIT.1976.1055638. URL: https://doi.org/10.1109/TIT.1976.1055638.

[46] Collectif. “Integer factorization and discrete logarithm problems”. en. In: Les cours du CIRM 1 (2014). talk:2. DOI: 10.5802/ccirm.21. URL: http://www.numdam.org/articles/10.5802/ccirm.21/.

[47] E. Brier, C. Clavier, and F. Olivier. “Correlation Power Analysis with a Leakage Model”. In: *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*. Ed. by M. Joye and J. Quisquater. Vol. 3156. Lecture Notes in Computer Science. Springer, 2004, pp. 16–29. DOI: 10.1007/978-3-540-28632-5_2. URL: https://doi.org/10.1007/978-3-540-28632-5%5C_2.

[48] K. Schramm, G. Leander, P. Felke, and C. Paar. “A Collision-Attack on AES: Combining Side Channel- and Differential-Attack”. In: *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*. Ed. by M. Joye and J. Quisquater. Vol. 3156. Lecture Notes in Computer Science. Springer, 2004, pp. 163–175. DOI: 10.1007/978-3-540-28632-5_12. URL: https://doi.org/10.1007/978-3-540-28632-5%5C_12.

[49] T. Akishita and T. Takagi. “Zero-Value Point Attacks on Elliptic Curve Cryptosystem”. In: *Information Security, 6th International Conference, ISC 2003, Bristol, UK, October 1-3, 2003, Proceedings*. Ed. by C. Boyd and W. Mao. Vol. 2851. Lecture Notes in Computer Science. Springer, 2003, pp. 218–233. DOI: 10.1007/10958513_17. URL: https://doi.org/10.1007/10958513%5C_17.

[50] J. Heyszl, A. Ibing, S. Mangard, F. D. Santis, and G. Sigl. “Clustering Algorithms for Non-profiled Single-Execution Attacks on Exponentiations”. In: *Smart Card Research and Advanced Applications - 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised Selected Papers*. Ed. by A. Francillon and P. Rohatgi. Vol. 8419. Lecture Notes in Computer Science. Springer, 2013, pp. 79–93. DOI: 10.1007/978-3-319-08302-5_6. URL: https://doi.org/10.1007/978-3-319-08302-5%5C_6.

[51] J. D. Golic and C. Tymen. “Multiplicative Masking and Power Analysis of AES”. In: *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*. Ed. by B. S. K. Jr., Ç. K. Koç, and C. Paar. Vol. 2523. Lecture Notes in Computer Science. Springer, 2002, pp. 198–212. DOI: 10.1007/3-540-36400-5_16. URL: https://doi.org/10.1007/3-540-36400-5%5C_16.

[52] S. Chari, J. R. Rao, and P. Rohatgi. “Template Attacks”. In: *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*. Ed. by B. S. K. Jr., Ç. K. Koç, and C. Paar. Vol. 2523. Lecture Notes in Computer Science. Springer, 2002, pp. 13–28. DOI: 10.1007/3-540-36400-5_3. URL: https://doi.org/10.1007/3-540-36400-5%5C_3.

[53] M. Carbone, V. Conin, M. Cornelie, F. Dassance, G. Dufresne, C. Dumas, E. Prouff, and A. Venelli. “Deep Learning to Evaluate Secure RSA Implementations”. In: *IACR Cryptol. ePrint Arch.* (2019), p. 54. URL: https://eprint.iacr.org/2019/054.

[54] L. Lerman, G. Bontempi, and O. Markowitch. “Power analysis attack: an approach based on machine learning”. In: *Int. J. Appl. Cryptogr.* 3.2 (2014), pp. 97–115. DOI: 10.1504/IJACT.2014.062722. URL: https://doi.org/10.1504/IJACT.2014.062722.

[55] G. Hospodar, B. Gierlichs, E. D. Mulder, I. Verbauwhede, and J. Vandewalle. “Machine learning in side-channel analysis: a first study”. In: *J. Cryptogr. Eng.* 1.4 (2011), pp. 293–302. DOI: 10.1007/s13389-011-0023-x. URL: https://doi.org/10.1007/s13389-011-0023-x.

[56] M. Masoumi, P. Habibi, and M. Jadidi. “Efficient implementation of masked AES on Side-Channel Attack Standard Evaluation Board”. In: *2015 International Conference on Information Society (i-Society)*. 2015, pp. 151–156. DOI: 10.1109/i-Society.2015.7366878.

[57] J. A. Ambrose, R. G. Ragel, and S. Parameswaran. “A smart random code injection to mask power analysis based side channel attacks”. In: *Proceedings of the 5th International Conference on Hardware/Software Codesign and System Synthesis, CODES+ISSS 2007, Salzburg, Austria, September 30 - October 3, 2007*. Ed. by S. Ha, K. Choi, N. D. Dutt, and J. Teich. ACM, 2007, pp. 51–56. DOI: 10.1145/1289816.1289832. URL: https://doi.org/10.1145/1289816.1289832.

[58] J. Coron and I. Kizhvatov. “Analysis and Improvement of the Random Delay Countermeasure of CHES 2009”. In: *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*. Ed. by S. Mangard and F. Standaert. Vol. 6225. Lecture Notes in Computer Science. Springer, 2010, pp. 95–109. DOI: 10.1007/978-3-642-15031-9_7. URL: https://doi.org/10.1007/978-3-642-15031-9%5C_7.

[59] H. Maghrebi, E. Prouff, S. Guilley, and J. Danger. “A First-Order Leak-Free Masking Countermeasure”. In: *Topics in Cryptology - CT-RSA 2012 - The Cryptographers’ Track at the RSA Conference 2012, San Francisco, CA, USA, February 27 - March 2, 2012. Proceedings*. Ed. by O. Dunkelman. Vol. 7178. Lecture Notes in Computer Science. Springer, 2012, pp. 156–170. DOI: 10.1007/978-3-642-27954-6_10. URL: https://doi.org/10.1007/978-3-642-27954-6%5C_10.

[60] Y. Lu, M. O’Neill, and J. V. McCanny. “Evaluation of Random Delay Insertion against DPA on FPGAs”. In: *ACM Trans. Reconfigurable Technol. Syst.* 4.1 (2010), 11:1–11:20. DOI: 10.1145/1857927.1857938. URL: https://doi.org/10.1145/1857927.1857938.

[61] T. Popp and S. Mangard. “Masked Dual-Rail Pre-charge Logic: DPA-Resistance Without Routing Constraints”. In: *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*. Ed. by J. R. Rao and B. Sunar. Vol. 3659. Lecture Notes in Computer Science. Springer, 2005, pp. 172–186. DOI: 10.1007/11545262_13. URL: https://doi.org/10.1007/11545262%5C_13.

[62] L. Zhang, L. Vega, and M. B. Taylor. “Power Side Channels in Security ICs: Hardware Countermeasures”. In: *CoRR* abs/1605.00681 (2016). arXiv: 1605.00681. URL: http://arxiv.org/abs/1605.00681.

[63] H. Nozaki, M. Motoyama, A. Shimbo, and S. Kawamura. “Implementation of RSA Algorithm Based on RNS Montgomery Multiplication”. In: *Cryptographic Hardware and Embedded Systems - CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings*. Ed. by Ç. K. Koç, D. Naccache, and C. Paar. Vol. 2162. Lecture Notes in Computer Science. Springer, 2001, pp. 364–376. DOI: 10.1007/3-540-44709-1_30. URL: https://doi.org/10.1007/3-540-44709-1%5C_30.

[64] Z. Chen and P. Schaumont. “Virtual Secure Circuit: Porting Dual-Rail Pre-charge Technique into Software on Multicore”. In: *IACR Cryptol. ePrint Arch.* (2010), p. 272. URL: http://eprint.iacr.org/2010/272.

[65] M. Doulcier-Verdier, J. Dutertre, J. J. A. Fournier, J. Rigaud, B. Robisson, and A. Tria. “A side-channel and fault-attack resistant AES circuit working on duplicated complemented values”. In: *IEEE International Solid-State Circuits Conference, ISSCC 2011, Digest of Technical Papers, San Francisco, CA, USA, 20-24 February, 2011*. IEEE, 2011, pp. 274–276. DOI: 10.1109/ISSCC.2011.5746316. URL: https://doi.org/10.1109/ISSCC.2011.5746316.

[66] C. Wang, M. Yan, Y. Cai, Q. Zhou, and J. Yang. “Power Profile Equalizer: A Lightweight Countermeasure against Side-Channel Attack”. In: *2017 IEEE International Conference on Computer Design, ICCD 2017, Boston, MA, USA, November 5-8, 2017*. IEEE Computer Society, 2017, pp. 305–312. DOI: 10.1109/ICCD.2017.54. URL: https://doi.org/10.1109/ICCD.2017.54.

[67] J.-L. Danger, S. Guilley, S. Bhasin, and M. Nassar. “Overview of Dual rail with Precharge logic styles to thwart implementation-level attacks on hardware cryptoprocessors”. In: *2009 3rd International Conference on Signals, Circuits and Systems (SCS)*. 2009, pp. 1–8. DOI: 10.1109/ICSCS.2009.5412599.

[68] H. Thapliyal, T. S. S. Varun, and S. D. Kumar. “Adiabatic Computing Based Low-Power and DPA-Resistant Lightweight Cryptography for IoT Devices”. In: *2017 IEEE Computer Society Annual Symposium on VLSI, ISVLSI 2017, Bochum, Germany, July 3-5, 2017*. IEEE Computer Society, 2017, pp. 621–626. DOI: 10.1109/ISVLSI.2017.115. URL: https://doi.org/10.1109/ISVLSI.2017.115.

[69] M. Masoumi. “Novel Hybrid CMOS/Memristor Implementation of the AES Algorithm Robust Against Differential Power Analysis Attack”. In: *IEEE Trans. Circuits Syst. II Express Briefs* 67-II.7 (2020), pp. 1314–1318. DOI: 10.1109/TCSII.2019.2932337. URL: https://doi.org/10.1109/TCSII.2019.2932337.

[70] Rambus. *DPA Workstation Platform*. URL: https://www.rambus.com/security/dpa-countermeasures/dpa-workstation-platform/ (visited on 05/08/2021).

[71] Riscure. *Inspector Side Channel Analysis*. URL: https://www.riscure.com/security-tools/inspector-sca (visited on 05/08/2021).

[72] A. Nahiyan, M. (Tony) He, J. Park, and M. Tehranipoor. “CAD for Side-Channel Leakage Assessment”. In: *Emerging Topics in Hardware Security*. Ed. by M. Tehranipoor. Cham: Springer International Publishing, 2021, pp. 171–197. ISBN: 978-3-030-64448-2. DOI: 10.1007/978-3-030-64448-2_7. URL: https://doi.org/10.1007/978-3-030-64448-2_7.

[73] G. Becker *et al. Test Vector Leakage Assessment (TVLA) methodology in practice*. 2011. URL: https://pdfs.semanticscholar.org/ (visited on 09/23/2019).

[74] S. Mangard. “Hardware Countermeasures against DPA ? A Statistical Analysis of Their Effectiveness”. In: *Topics in Cryptology - CT-RSA 2004, The Cryptographers' Track at the RSA Conference 2004, San Francisco, CA, USA, February 23-27, 2004, Proceedings*. Ed. by T. Okamoto. Vol. 2964. Lecture Notes in Computer Science. Springer, 2004, pp. 222–235. DOI: 10.1007/978-3-540-24660-2_18. URL: https://doi.org/10.1007/978-3-540-24660-2%5C_18.

[75] R. Bloem, H. Groß, R. Iusupov, B. Könighofer, S. Mangard, and J. Winter. “Formal Verification of Masked Hardware Implementations in the Presence of Glitches”. In: *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*. Ed. by J. B. Nielsen and V. Rijmen. Vol. 10821. Lecture Notes in Computer Science. Springer, 2018, pp. 321–353. DOI: 10.1007/978-3-319-78375-8_11. URL: https://doi.org/10.1007/978-3-319-78375-8%5C_11.

[76] H. Eldib, C. Wang, and P. Schaumont. “Formal Verification of Software Countermeasures against Side-Channel Attacks”. In: *ACM Trans. Softw. Eng. Methodol.* 24.2 (2014), 11:1–11:24. DOI: 10.1145/2685616. URL: https://doi.org/10.1145/2685616.

[77] K. Gandolfi, C. Mourtel, and F. Olivier. “Electromagnetic Analysis: Concrete Results”. In: *Cryptographic Hardware and Embedded Systems - CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings*. Ed. by Ç. K. Koç, D. Naccache, and C. Paar. Vol. 2162. Lecture Notes in Computer Science. Springer, 2001, pp. 251–261. DOI: 10.1007/3-540-44709-1_21. URL: https://doi.org/10.1007/3-540-44709-1%5C_21.

[78] J. DaRolt, A. Das, G. D. Natale, M. Flottes, B. Rouzeyre, and I. Verbauwhede. “A New Scan Attack on RSA in Presence of Industrial Countermeasures”. In: *Constructive Side-Channel Analysis and Secure Design - Third International Workshop, COSADE 2012, Darmstadt, Germany, May 3-4, 2012. Proceedings*. Ed. by W. Schindler and S. A. Huss. Vol. 7275. Lecture Notes in Computer Science. Springer, 2012, pp. 89–104. DOI: 10.1007/978-3-642-29912-4_8. URL: https://doi.org/10.1007/978-3-642-29912-4%5C_8.

[79] A. Schlösser, D. Nedospasov, J. Krämer, S. Orlic, and J. Seifert. “Simple photonic emission analysis of AES”. In: *J. Cryptogr. Eng.* 3.1 (2013), pp. 3–15. DOI: 10.1007/S13389-013-0053-7. URL: https://doi.org/10.1007/s13389-013-0053-7.

[80] D. Genkin, A. Shamir, and E. Tromer. “RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis”. In: *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*. Ed. by J. A. Garay and R. Gennaro. Vol. 8616. Lecture Notes in Computer Science. Springer, 2014, pp. 444–461. DOI: 10.1007/978-3-662-44371-2_25. URL: https://doi.org/10.1007/978-3-662-44371-2%5C_25.

[81] J. Coron, E. Prouff, M. Rivain, and T. Roche. “Higher-Order Side Channel Security and Mask Refreshing”. In: *IACR Cryptol. ePrint Arch.* (2015), p. 359. URL: http://eprint.iacr.org/2015/359.

[82] K. Tiri and I. Verbauwhede. “A Dynamic and Differential CMOS Logic Style to Resist Power and Timing Attacks on Security IC’s”. In: *IACR Cryptol. ePrint Arch.* (2004), p. 66. URL: http://eprint.iacr.org/2004/066.

[83] F. Durvaux, M. Renauld, F. Standaert, L. van Oldeneel tot Oldenzeel, and N. Veyrat-Charvillon. “Efficient Removal of Random Delays from Embedded Software Implementations Using Hidden Markov Models”. In: *Smart Card Research and Advanced Applications - 11th International Conference, CARDIS 2012, Graz, Austria, November 28-30, 2012, Revised Selected Papers*. Ed. by S. Mangard. Vol. 7771. Lecture Notes in Computer Science. Springer, 2012, pp. 123–140. DOI: 10.1007/978-3-642-37288-9_9. URL: https://doi.org/10.1007/978-3-642-37288-9%5C_9.

[84] V. Zeman and Z. Martinasek. “Innovative Method of the Power Analysis”. In: *Radioengineering* 22 (2013), pp. 586–594. URL: https://api.semanticscholar.org/CorpusID:54614105.

[85] R. Gilmore, N. Hanley, and M. O’Neill. “Neural network based attack on a masked implementation of AES”. In: *IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2015, Washington, DC, USA, 5-7 May, 2015*. IEEE Computer Society, 2015, pp. 106–111. DOI: 10.1109/HST.2015.7140247. URL: https://doi.org/10.1109/HST.2015.7140247.

[86] H. Maghrebi. “Deep Learning based Side Channel Attacks in Practice”. In: *IACR Cryptol. ePrint Arch.* (2019), p. 578. URL: https://eprint.iacr.org/2019/578.

[87] M. Carbone, V. Conin, M. Cornelie, F. Dassance, G. Dufresne, C. Dumas, E. Prouff, and A. Venelli. “Deep Learning to Evaluate Secure RSA Implementations”. In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2019.2 (2019), pp. 132–161. DOI: 10.13154/tches.v2019.i2.132-161. URL: https://doi.org/10.13154/tches.v2019.i2.132-161.

[88] X. Glorot and Y. Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*. Ed. by Y. W. Teh and D. M. Titterington. Vol. 9. JMLR Proceedings. JMLR.org, 2010, pp. 249–256. URL: http://proceedings.mlr.press/v9/glorot10a.html.

[89] D. P. Kingma and J. Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2015. URL: http://arxiv.org/abs/1412.6980.

[90] P. Baldi and P. J. Sadowski. “Understanding Dropout”. In: *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*. Ed. by C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger. 2013, pp. 2814–2822. URL: https://proceedings.neurips.cc/paper/2013/hash/71f6278d140af599e06ad9bf1ba03cb0-Abstract.html.

[91] F. Xiao, T. Lu, M. Wu, and Q. Ai. “Maximal overlap discrete wavelet transform and deep learning for robust denoising and detection of power quality disturbance”. In: *IET Generation, Transmission & Distribution* 14.1 (2020), pp. 140–147. DOI: https://doi.org/10.1049/iet-gtd.2019.1121. eprint: https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-gtd.2019.1121. URL: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-gtd.2019.1121.

[92] S. A. Bigdeli and M. Zwicker. “Image Restoration using Autoencoding Priors”. In: *Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2018) - Volume 5: VISAPP, Funchal, Madeira, Portugal, January 27-29, 2018*. Ed. by F. H. Imai, A. Trémeau, and J. Braz. SciTePress, 2018, pp. 33–44. DOI: 10.5220/0006532100330044. URL: https://doi.org/10.5220/0006532100330044.

[93] VLSI Research Group – COMELEC Department of the Telecom ParisTech. *DPA Contest v2*. URL: http://www.dpacontest.org/v2/index.php (visited on 09/23/2019).

[94] NewAE Technology Inc. *Chipwhisperer-Lite two part board*. URL: http://store.newae.com/chipwhisperer-lite-cw1173-two-part-version/ (visited on 01/31/2020).

[95] Riscure. *The CHES 2018 Challenge*. URL: https://chesctf.riscure.com/2018/news (visited on 02/15/2020).

[96] Riscure. *Pinata board*. URL: https://www.riscure.com/product/pinata-training-target/ (visited on 01/31/2020).

[97] H. Aly and M. ElGayyar. “Attacking AES Using Bernstein’s Attack on Modern Processors”. In: *Progress in Cryptology - AFRICACRYPT 2013, 6th International Conference on Cryptology in Africa, Cairo, Egypt, June 22-24, 2013. Proceedings*. Ed. by A. M. Youssef, A. Nitaj, and A. E. Hassanien. Vol. 7918. Lecture Notes in Computer Science. Springer, 2013, pp. 127–139. DOI: 10.1007/978-3-642-38553-7_7. URL: https://doi.org/10.1007/978-3-642-38553-7%5C_7.

[98] J. Dhem and J. Quisquater. “Recent Results on Modular Multiplications for Smart Cards”. In: *Smart Card Research and Applications, This International Conference, CARDIS ’98, Louvain-la-Neuve, Belgium, September 14-16, 1998, Proceedings*. Ed. by J. Quisquater and B. Schneier. Vol. 1820. Lecture Notes in Computer Science. Springer, 1998, pp. 336–352. DOI: 10.1007/10721064_31. URL: https://doi.org/10.1007/10721064%5C_31.

[99] W. Schindler. “A Timing Attack against RSA with the Chinese Remainder Theorem”. In: *Cryptographic Hardware and Embedded Systems - CHES 2000, Second International Workshop, Worcester, MA, USA, August 17-18, 2000, Proceedings*. Ed. by Ç. K. Koç and C. Paar. Vol. 1965. Lecture Notes in Computer Science. Springer, 2000, pp. 109–124. DOI: 10.1007/3-540-44499-8_8. URL: https://doi.org/10.1007/3-540-44499-8%5C_8.

[100] D. Brumley and D. Boneh. “Remote Timing Attacks Are Practical”. In: *Proceedings of the 12th USENIX Security Symposium, Washington, D.C., USA, August 4-8, 2003*. USENIX Association, 2003. URL: https://www.usenix.org/conference/12th-usenix-security-symposium/remote-timing-attacks-are-practical.

[101] Y. Tomoeda, H. Miyake, A. Shimbo, and S. Kawamura. “An SPA-Based Extension of Schindler’s Timing Attack against RSA Using CRT”. In: *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* 88-A.1 (2005), pp. 147–153. URL: http://search.ieice.org/bin/summary.php?id=e88-a%5C_1%5C_147%5C&category=D%5C&year=2005%5C&lang=E%5C&abst=.

[102] P. Kocher, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom. "Spectre Attacks: Exploiting Speculative Execution". In: *meltdownattack.com* (2018). URL: https://spectreattack.com/spectre.pdf.

[103] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg. "Meltdown: Reading Kernel Memory from User Space". In: *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*. Ed. by W. Enck and A. P. Felt. USENIX Association, 2018, pp. 973–990. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/lipp.

[104] A. Loiseau, M. Lecomte, and J. J. A. Fournier. "Template Attacks against ECC: practical implementation against Curve25519". In: *2020 IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2020, San Jose, CA, USA, December 7-11, 2020*. IEEE, 2020, pp. 13–22. DOI: 10.1109/HOST45689.2020.9300261. URL: https://doi.org/10.1109/HOST45689.2020.9300261.

[105] A. Vajda. "Multi-core and Many-core Processor Architectures". In: *Programming Many-Core Chips*. Boston, MA: Springer US, 2011, pp. 9–43. ISBN: 978-1-4419-9739-5. DOI: 10.1007/978-1-4419-9739-5_2. URL: https://doi.org/10.1007/978-1-4419-9739-5_2.

[106] O. Aciiçmez and Ç. K. Koç. "Trace-Driven Cache Attacks on AES". In: *IACR Cryptol. ePrint Arch.* (2006), p. 138. URL: http://eprint.iacr.org/2006/138.

[107] T. Peyrin. *GIFT Block CIpher*. https://github.com/giftcipher/gift. 2019.

[108] M. Gautschi, P. D. Schiavone, A. Traber, I. Loi, A. Pullini, D. Rossi, E. Flamand, F. K. Gürkaynak, and L. Benini. "Near-Threshold RISC-V Core With DSP Extensions for Scalable IoT Endpoint Devices". In: *IEEE Trans. Very Large Scale Integr. Syst.* 25.10 (2017), pp. 2700–2713. DOI: 10.1109/TVLSI.2017.2654506. URL: https://doi.org/10.1109/TVLSI.2017.2654506.

[109] C. V. Penumuchu. *Simple Real-Time Operating System: A Kernel Inside View for a Beginner*. Trafford Publishing, 2007.

[110] Xilinx. *kintex-7*. Online. https://www.xilinx.com/products/silicon-devices/fpga/kintex-7.html. 2020.

[111] J. Brouchier, T. Kean, C. Marsh, and D. Naccache. "Temperature Attacks". In: *IEEE Secur. Priv.* 7.2 (2009), pp. 79–82. DOI: 10.1109/MSP.2009.54. URL: https://doi.org/10.1109/MSP.2009.54.

[112] D. B. Bartolini, P. Miedl, and L. Thiele. "On the capacity of thermal covert channels in multicores". In: *Proceedings of the Eleventh European Conference on Computer Systems, EuroSys 2016, London, United Kingdom, April 18-21, 2016*. Ed. by C. Cadar, P. R. Pietzuch, K. Keeton, and R. Rodrigues. ACM, 2016, 24:1–24:16. DOI: 10.1145/2901318.2901322. URL: https://doi.org/10.1145/2901318.2901322.

[113] M. Hutter and J. Schmidt. “The Temperature Side Channel and Heating Fault Attacks”. In: *IACR Cryptol. ePrint Arch.* (2014), p. 190. URL: http://eprint.iacr.org/2014/190.

[114] M. Happe, A. Agne, and C. Plessl. “Measuring and Predicting Temperature Distributions on FPGAs at Run-Time”. In: *2011 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2011, Cancun, Mexico, November 30 - December 2, 2011*. Ed. by P. M. Athanas, J. Becker, and R. Cumplido. IEEE Computer Society, 2011, pp. 55–60. DOI: 10.1109/ReConFig.2011.59. URL: https://doi.org/10.1109/ReConFig.2011.59.

[115] J. S. Lee, K. Skadron, and S. W. Chung. “Predictive Temperature-Aware DVFS”. In: *IEEE Trans. Computers* 59.1 (2010), pp. 127–133. DOI: 10.1109/TC.2009.136. URL: https://doi.org/10.1109/TC.2009.136.

[116] Digilent. *PYNQ-Z1: Python Productivity for Zynq-7000 ARM/FPGA SoC*. 2020. URL: https://store.digilentinc.com/pynq-z1-python-productivity-for-zynq-7000-arm-fpga-soc/ (visited on 11/13/2020).

[117] Xilinx. *7 Series FPGAs and Zynq-7000 SoC XADC Dual 12-Bit 1 MSPS Analog-to-Digital Converter*. 2018. URL: https://docs.xilinx.com/r/en-US/ug480_7Series_XADC (visited on 11/13/2020).

[118] B. Gierlichs, K. Lemke-Rust, and C. Paar. “Templates vs. Stochastic Methods”. In: *Cryptographic Hardware and Embedded Systems - CHES 2006, 8th International Workshop, Yokohama, Japan, October 10-13, 2006, Proceedings*. Ed. by L. Goubin and M. Matsui. Vol. 4249. Lecture Notes in Computer Science. Springer, 2006, pp. 15–29. DOI: 10.1007/11894063_2. URL: https://doi.org/10.1007/11894063%5C_2.

[119] R. Kaiser and W. Knight. “Digital signal averaging”. In: *Journal of Magnetic Resonance (1969)* 36.2 (1979), pp. 215–220. ISSN: 0022-2364. DOI: https://doi.org/10.1016/0022-2364(79)90096-9. URL: https://www.sciencedirect.com/science/article/pii/0022236479900969.

[120] C. C. Enz and G. C. Temes. “Circuit techniques for reducing the effects of op-amp imperfections: autozeroing, correlated double sampling, and chopper stabilization”. In: *Proc. IEEE* 84.11 (1996), pp. 1584–1614. DOI: 10.1109/5.542410. URL: https://doi.org/10.1109/5.542410.

[121] pyca. *cryptography package for python developer*. 2020. URL: https://cryptography.io/en/latest/hazmat/primitives/asymmetric/rsa/ (visited on 09/21/2020).

[122] E. Prouff, R. Strullu, R. Benadjila, E. Cagli, and C. Dumas. “Study of Deep Learning Techniques for Side-Channel Analysis and Introduction to ASCAD Database”. In: *IACR Cryptol. ePrint Arch.* (2018), p. 53. URL: http://eprint.iacr.org/2018/053.

[123] I. J. Goodfellow, Y. Bengio, and A. C. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016. ISBN: 978-0-262-03561-3. URL: http://www.deeplearningbook.org/.

[124] A. F. Agarap. "Deep Learning using Rectified Linear Units (ReLU)". In: *CoRR* abs/1803.08375 (2018). arXiv: 1803.08375. URL: http://arxiv.org/abs/1803.08375.

[125] S. Ioffe and C. Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. Ed. by F. R. Bach and D. M. Blei. Vol. 37. JMLR Workshop and Conference Proceedings. JMLR.org, 2015, pp. 448–456. URL: http://proceedings.mlr.press/v37/ioffe15.html.

[126] S. Liu and W. Deng. "Very deep convolutional neural network based image classification using small training sample size". In: *3rd IAPR Asian Conference on Pattern Recognition, ACPR 2015, Kuala Lumpur, Malaysia, November 3-6, 2015*. IEEE, 2015, pp. 730–734. DOI: 10.1109/ACPR.2015.7486599. URL: https://doi.org/10.1109/ACPR.2015.7486599.

[127] A. Camuto, M. Willetts, U. Simsekli, S. J. Roberts, and C. C. Holmes. "Explicit Regularisation in Gaussian Noise Injections". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. 2020. URL: https://proceedings.neurips.cc/paper/2020/hash/c16a5320fa475530d9583c34fd356ef5-Abstract.html.

[128] A. Bauer, É. Jaulmes, E. Prouff, and J. Wild. "Horizontal and Vertical Side-Channel Attacks against Secure RSA Implementations". In: *Topics in Cryptology - CT-RSA 2013 - The Cryptographers' Track at the RSA Conference 2013, San Francisco,CA, USA, February 25-March 1, 2013. Proceedings*. Ed. by E. Dawson. Vol. 7779. Lecture Notes in Computer Science. Springer, 2013, pp. 1–17. DOI: 10.1007/978-3-642-36095-4_1. URL: https://doi.org/10.1007/978-3-642-36095-4%5C_1.

[129] Z. Ding, W. Guo, L. Su, J. Wei, and H. Gu. "Further Research on N-1 Attack against Exponentiation Algorithms". In: *Information Security and Privacy - 19th Australasian Conference, ACISP 2014, Wollongong, NSW, Australia, July 7-9, 2014. Proceedings*. Ed. by W. Susilo and Y. Mu. Vol. 8544. Lecture Notes in Computer Science. Springer, 2014, pp. 162–175. DOI: 10.1007/978-3-319-08344-5_11. URL: https://doi.org/10.1007/978-3-319-08344-5%5C_11.

[130] Zeroplus. *Zeroplus - Logic Analyzer LAP-C 16032*. 2021. URL: http://www.zeroplus.com.tw/logic-analyzer%5C_en/products.php?pdn=1%5C&product%5C_id=253.

[131] C. Clavier, B. Feix, G. Gagnerot, M. Roussellet, and V. Verneuil. "Improved Collision-Correlation Power Analysis on First Order Protected AES". In: *Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings*. Ed. by B. Preneel and T. Takagi. Vol. 6917. Lecture Notes in Computer Science. Springer, 2011, pp. 49–62. DOI: 10.1007/978-3-642-23951-9_4. URL: https://doi.org/10.1007/978-3-642-23951-9%5C_4.

[132] F. Standaert. “Introduction to Side-Channel Attacks”. In: *Secure Integrated Circuits and Systems*. Ed. by I. M. R. Verbauwhede. Integrated Circuits and Systems. Springer, 2010, pp. 27–42. DOI: 10.1007/978-0-387-71829-3_2. URL: https://doi.org/10.1007/978-0-387-71829-3%5C_2.

[133] N. technology inc. *Measuring SNR of Target*. URL: https://chipwhisperer.readthedocs.io/en/latest/tutorials/pa_intro_3-openadc-cwlitearm.html/ (visited on 05/13/2020).

[134] O. Aciiçmez, Ç. K. Koç, and J. Seifert. “On the Power of Simple Branch Prediction Analysis”. In: *IACR Cryptol. ePrint Arch.* (2006), p. 351. URL: http://eprint.iacr.org/2006/351.

[135] M. F. Witteman, J. G. J. van Woudenberg, and F. Menarini. “Defeating RSA Multiply-Always and Message Blinding Countermeasures”. In: *Topics in Cryptology - CT-RSA 2011 - The Cryptographers’ Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings*. Ed. by A. Kiayias. Vol. 6558. Lecture Notes in Computer Science. Springer, 2011, pp. 77–88. DOI: 10.1007/978-3-642-19074-2_6. URL: https://doi.org/10.1007/978-3-642-19074-2%5C_6.

[136] S. Bhasin, J. Danger, S. Guilley, and Z. Najm. “NICV: Normalized Inter-Class Variance for Detection of Side-Channel Leakage”. In: *IACR Cryptol. ePrint Arch.* (2013), p. 717. URL: http://eprint.iacr.org/2013/717.

[137] H. S. Kim and S. Hong. “New Type of Collision Attack on First-Order Masked AESs”. In: *ETRI Journal* 38.2 (2016), pp. 387–396. DOI: https://doi.org/10.4218/etrij.16.0114.0854. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.4218/etrij.16.0114.0854. URL: https://onlinelibrary.wiley.com/doi/abs/10.4218/etrij.16.0114.0854.

[138] Y. Won, P. Hodgers, M. O’Neill, and D. Han. “On the Security of Balanced Encoding Countermeasures”. In: *Smart Card Research and Advanced Applications - 14th International Conference, CARDIS 2015, Bochum, Germany, November 4-6, 2015. Revised Selected Papers*. Ed. by N. Homma and M. Medwed. Vol. 9514. Lecture Notes in Computer Science. Springer, 2015, pp. 242–256. DOI: 10.1007/978-3-319-31271-2_15. URL: https://doi.org/10.1007/978-3-319-31271-2%5C_15.

[139] S. P. Karthikeyan and H. El-Razouk. “Horizontal Correlation Analysis of Elliptic Curve Diffie Hellman”. In: *3rd International Conference on Information and Computer Technologies, ICICT 2020, San Jose, CA, USA, March 9-12, 2020*. IEEE, 2020, pp. 511–519. DOI: 10.1109/ICICT50521.2020.00087. URL: https://doi.org/10.1109/ICICT50521.2020.00087.

[140] M. Lu, A. Fan, J. Xu, and W. Shan. “A Compact, Lightweight and Low-Cost 8-Bit Datapath AES Circuit for IoT Applications in 28nm CMOS”. In: *17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications / 12th IEEE International Conference On Big Data Science And Engineering, TrustCom/BigDataSE 2018, New York, NY, USA, August 1-3, 2018*. IEEE, 2018,

pp. 1464–1469. DOI: 10.1109/TrustCom/BigDataSE.2018.00204. URL: https://doi.org/10.1109/TrustCom/BigDataSE.2018.00204.

[141] S. N. Dhanuskodi, S. Allen, and D. E. Holcomb. "Efficient Register Renaming Architectures for 8-bit AES Datapath at 0.55 pJ/bit in 16-nm FinFET". In: *IEEE Trans. Very Large Scale Integr. Syst.* 28.8 (2020), pp. 1807–1820. DOI: 10.1109/TVLSI.2020.2999593. URL: https://doi.org/10.1109/TVLSI.2020.2999593.

[142] M. S. Wamser and G. Sigl. "Pushing the limits further: Sub-atomic AES". In: *2017 IFIP/IEEE International Conference on Very Large Scale Integration, VLSI-SoC 2017, Abu Dhabi, United Arab Emirates, October 23-25, 2017*. IEEE, 2017, pp. 1–6. DOI: 10.1109/VLSI-SoC.2017.8203470. URL: https://doi.org/10.1109/VLSI-SoC.2017.8203470.

[143] S. Banik, A. Bogdanov, and F. Regazzoni. "Atomic-AES: A Compact Implementation of the AES Encryption/Decryption Core". In: *Progress in Cryptology - INDOCRYPT 2016 - 17th International Conference on Cryptology in India, Kolkata, India, December 11-14, 2016, Proceedings*. Ed. by O. Dunkelman and S. K. Sanadhya. Vol. 10095. Lecture Notes in Computer Science. 2016, pp. 173–190. DOI: 10.1007/978-3-319-49890-4_10. URL: https://doi.org/10.1007/978-3-319-49890-4%5C_10.

[144] A. Moradi, A. Poschmann, S. Ling, C. Paar, and H. Wang. "Pushing the Limits: A Very Compact and a Threshold Implementation of AES". In: *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*. Ed. by K. G. Paterson. Vol. 6632. Lecture Notes in Computer Science. Springer, 2011, pp. 69–88. DOI: 10.1007/978-3-642-20465-4_6. URL: https://doi.org/10.1007/978-3-642-20465-4%5C_6.

[145] S. Mathew, S. Satpathy, V. B. Suresh, M. A. Anders, H. Kaul, A. Agarwal, S. Hsu, G. K. Chen, and R. Krishnamurthy. "340 mV-1.1 V, 289 Gbps/W, 2090-Gate NanoAES Hardware Accelerator With Area-Optimized Encrypt/Decrypt GF(2 4) 2 Polynomials in 22 nm Tri-Gate CMOS". In: *IEEE J. Solid State Circuits* 50.4 (2015), pp. 1048–1058. DOI: 10.1109/JSSC.2014.2384039. URL: https://doi.org/10.1109/JSSC.2014.2384039.

[146] J. Yu and M. Aagaard. "Benchmarking and Optimizing AES for Lightweight Cryptography on ASICs," in: *Proceedings of the Lightweight Cryptography Workshop, Gaithersburg, MD, USA*. 4–6 November, 2019.

[147] M.-H. Dao, V.-P. Hoang, V.-L. Dao, and X.-T. Tran. "An Energy Efficient AES Encryption Core for Hardware Security Implementation in IoT Systems". In: *2018 International Conference on ATC*. 2018, pp. 301–304. DOI: 10.1109/ATC.2018.8587500.

[148] C. Davis and E. John. "Shared Round Core Architecture: A Novel AES Implementation for Implantable Cardiac Devices". In: *65th IEEE International Midwest Symposium on Circuits and Systems, MWSCAS 2022, Fukuoka, Japan, August 7-10, 2022*. IEEE, 2022, pp. 1–4. DOI: 10.1109/MWSCAS54063.2022.9859276. URL: https://doi.org/10.1109/MWSCAS54063.2022.9859276.

[149] A. Satoh, S. Morioka, K. Takano, and S. Munetoh. “A Compact Rijndael Hardware Architecture with S-Box Optimization”. In: *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*. Ed. by C. Boyd. Vol. 2248. Lecture Notes in Computer Science. Springer, 2001, pp. 239–254. DOI: 10.1007/3-540-45682-1_15. URL: https://doi.org/10.1007/3-540-45682-1%5C_15.

[150] N. Ahmad and S. M. R. Hasan. “Low-power compact composite field AES S-Box/Inv S-Box design in 65 nm CMOS using Novel XOR Gate”. In: *Integr.* 46.4 (2013), pp. 333–344. DOI: 10.1016/j.vlsi.2012.06.002. URL: https://doi.org/10.1016/j.vlsi.2012.06.002.

[151] Y. Teng, W. Chin, D. Chang, P. Chen, and P. Chen. “VLSI Architecture of S-Box With High Area Efficiency Based on Composite Field Arithmetic”. In: *IEEE Access* 10 (2022), pp. 2721–2728. DOI: 10.1109/ACCESS.2021.3139040. URL: https://doi.org/10.1109/ACCESS.2021.3139040.

[152] N. Ahmad. “NEW ARCHITECTURE OF LOW AREA AES S-BOX/ INV S-BOX USING VLSI IMPLEMENTATION”. In: *Jurnal Teknologi* 78.5-9 (May 2016).

[153] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer. “A comparative study of LPWAN technologies for large-scale IoT deployment”. In: *ICT Express* 5.1 (2019), pp. 1–7. DOI: 10.1016/j.icte.2017.12.005. URL: https://doi.org/10.1016/j.icte.2017.12.005.

[154] D. Thavamani. “MQTT Messages-An Overview”. In: *International Journal of Mathematics and Computer Research* 09 (Apr. 2021). DOI: 10.47191/ijmcr/v9i4.07.

[155] M. T. He, J. Park, A. Nahiyan, A. Vassilev, Y. Jin, and M. M. Tehranipoor. “RTL-PSC: Automated Power Side-Channel Leakage Assessment at Register-Transfer Level”. In: *37th IEEE VLSI Test Symposium, VTS 2019, Monterey, CA, USA, April 23-25, 2019*. IEEE, 2019, pp. 1–6. DOI: 10.1109/VTS.2019.8758600. URL: https://doi.org/10.1109/VTS.2019.8758600.

[156] R. Sadhukhan, P. Mathew, D. B. Roy, and D. Mukhopadhyay. “Count Your Toggles: a New Leakage Model for Pre-Silicon Power Analysis of Crypto Designs”. In: *J. Electron. Test.* 35.5 (2019), pp. 605–619. DOI: 10.1007/s10836-019-05826-8. URL: https://doi.org/10.1007/s10836-019-05826-8.

[157] A. Nahiyan, J. Park, M. T. He, Y. Iskander, F. Farahmandi, D. Forte, and M. M. Tehranipoor. “SCRIPT: A CAD Framework for Power Side-channel Vulnerability Assessment Using Information Flow Tracking and Pattern Generation”. In: *ACM Trans. Design Autom. Electr. Syst.* 25.3 (2020), 26:1–26:27. DOI: 10.1145/3383445. URL: https://doi.org/10.1145/3383445.

[158] RAMBUS. *DPA Resistant Core - Rambus*. URL: https://www.rambus.com/security/dpa-countermeasures/dpa-resistant-core/.

[159] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. “Generative Adversarial Networks”. In: *CoRR* abs/1406.2661 (2014). arXiv: 1406.2661. URL: http://arxiv.org/abs/1406.2661.

[160] M. Mirza and S. Osindero. "Conditional Generative Adversarial Nets". In: *CoRR* abs/1411.1784 (2014). arXiv: 1411.1784. URL: http://arxiv.org/abs/1411.1784.

[161] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley. "Least Squares Generative Adversarial Networks". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, pp. 2813–2821. DOI: 10.1109/ICCV.2017.304. URL: https://doi.org/10.1109/ICCV.2017.304.

[162] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. "Improved Training of Wasserstein GANs". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett. 2017, pp. 5767–5777. URL: https://proceedings.neurips.cc/paper/2017/hash/892c3b1c6dccd52936e27cbd0ff683d6-Abstract.html.

[163] G. Basso. *A Hitchhikers guide to Wasserstein distances*. June 2015. URL: https://bit.ly/3lFixDe.

[164] M. Arjovsky, S. Chintala, and L. Bottou. "Wasserstein GAN". In: *CoRR* abs/1701.07875 (2017). arXiv: 1701.07875. URL: http://arxiv.org/abs/1701.07875.

[165] A. Brock, J. Donahue, and K. Simonyan. "Large Scale GAN Training for High Fidelity Natural Image Synthesis". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL: https://openreview.net/forum?id=B1xsqj09Fm.

[166] J. H. Engel, K. K. Agrawal, S. Chen, I. Gulrajani, C. Donahue, and A. Roberts. "GANSynth: Adversarial Neural Audio Synthesis". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL: https://openreview.net/forum?id=H1xQVn09FX.

[167] "HRVGAN: High Resolution Video Generation using Spatio-Temporal GAN". In: *CoRR* abs/2008.09646 (2020). Withdrawn. arXiv: 2008.09646. URL: https://arxiv.org/abs/2008.09646.

[168] K. Antczak. "A Generative Adversarial Approach To ECG Synthesis And Denoising". In: *CoRR* abs/2009.02700 (2020). arXiv: 2009.02700. URL: https://arxiv.org/abs/2009.02700.

[169] R. Prenger, R. Valle, and B. Catanzaro. "Waveglow: A Flow-based Generative Network for Speech Synthesis". In: *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*. IEEE, 2019, pp. 3617–3621. DOI: 10.1109/ICASSP.2019.8683143. URL: https://doi.org/10.1109/ICASSP.2019.8683143.

[170] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu. "WaveNet: A Generative Model for Raw Audio". In: *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13-15 September 2016*. ISCA, 2016, p. 125. URL: http://www.isca-speech.org/archive/SSW%5C_2016/abstracts/ssw9%5C_DS-4%5C_van%5C_den%5C_Oord.html.

[171] A. Odena. "Open Questions about Generative Adversarial Networks". In: *Distill* (2019). https://distill.pub/2019/gan-open-problems. DOI: 10.23915/distill.00018.

[172] P. Wang, P. Chen, Z. Luo, G. Dong, M. Zheng, N. Yu, and H. Hu. "Enhancing the Performance of Practical Profiling Side-Channel Attacks Using Conditional Generative Adversarial Networks". In: *CoRR* abs/2007.05285 (2020). arXiv: 2007.05285. URL: https://arxiv.org/abs/2007.05285.

[173] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brébisson, Y. Bengio, and A. C. Courville. "MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett. 2019, pp. 14881–14892. URL: https://proceedings.neurips.cc/paper/2019/hash/6804c9bca0a615bdb9374d00a9fcba59-Abstract.html.

[174] M. Mathieu, C. Couprie, and Y. LeCun. "Deep multi-scale video prediction beyond mean square error". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2016. URL: http://arxiv.org/abs/1511.05440.

[175] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. "Image-to-Image Translation with Conditional Adversarial Networks". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, pp. 5967–5976. DOI: 10.1109/CVPR.2017.632. URL: https://doi.org/10.1109/CVPR.2017.632.

[176] Y. Wang *et al.* "Transferring GANs: generating images from limited data". In: *CoRR* abs/1805.01677 (2018). arXiv: 1805.01677. URL: http://arxiv.org/abs/1805.01677.

[177] Siemens. *Questa Advanced Simulato*. URL: https://eda.sw.siemens.com/en-US/ic/questa/simulation/advanced-simulator/ (visited on 05/08/2021).

[178] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc,

E. B. Fox, and R. Garnett. 2019, pp. 8024–8035. URL: https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

[179] M. Müller. *Information retrieval for music and motion*. Springer, 2007. DOI: 10.1007/978-3-540-74048-3. URL: https://doi.org/10.1007/978-3-540-74048-3.

[180] K. D. Rao and M. Swamy. "Spectral Analysis of Signals". In: *Digital Signal Processing: Theory and Practice*. Singapore: Springer Singapore, 2018, pp. 721–751. ISBN: 978-981-10-8081-4. DOI: 10.1007/978-981-10-8081-4_12. URL: https://doi.org/10.1007/978-981-10-8081-4_12.

[181] A. Radford, L. Metz, and S. Chintala. "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2016. URL: http://arxiv.org/abs/1511.06434.

[182] R. Tubbing. "An Analysis of Deep Learning Based Profiled Side-channel Attacks". MA thesis. the Netherlands: Delft University of Technology, 2019.

[183] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet. "Are GANs Created Equal? A Large-Scale Study". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. 2018, pp. 698–707. URL: https://proceedings.neurips.cc/paper/2018/hash/e46de7e1bcaaced9a54f1e9Abstract.html.

[184] S. Salvador and P. Chan. "Toward accurate dynamic time warping in linear time and space". In: *Intell. Data Anal.* 11.5 (2007), pp. 561–580. URL: http://content.iospress.com/articles/intelligent-data-analysis/ida00303.

[185] A. V. Lakshmy, C. Rebeiro, and S. Bhunia. "FORTIFY: Analytical Pre-Silicon Side-Channel Characterization of Digital Designs". In: *27th Asia and South Pacific Design Automation Conference, ASP-DAC 2022, Taipei, Taiwan, January 17-20, 2022*. IEEE, 2022, pp. 660–665. DOI: 10.1109/ASP-DAC52403.2022.9712551. URL: https://doi.org/10.1109/ASP-DAC52403.2022.9712551.

[186] P. SLPSK, P. K. Vairam, C. Rebeiro, and V. Kamakoti. "Karna: A Gate-Sizing based Security Aware EDA Flow for Improved Power Side-Channel Attack Protection". In: *Proceedings of the International Conference on Computer-Aided Design, ICCAD 2019, Westminster, CO, USA, November 4-7, 2019*. Ed. by D. Z. Pan. ACM, 2019, pp. 1–8. DOI: 10.1109/ICCAD45719.2019.8942173. URL: https://doi.org/10.1109/ICCAD45719.2019.8942173.

[187] I. Buhan, L. Batina, Y. Yarom, and P. Schaumont. "SoK: Design Tools for Side-Channel-Aware Implementations". In: *ASIA CCS '22*. 2022. DOI: 10.1145/3488932.3517415. URL: https://doi.org/10.1145/3488932.3517415.

[188] Y. Zhou, H. Ren, Y. Zhang, B. Keller, B. Khailany, and Z. Zhang. “PRIMAL: Power Inference using Machine Learning”. In: *Proceedings of the 56th Annual Design Automation Conference 2019, DAC 2019, Las Vegas, NV, USA, June 02-06, 2019*. ACM, 2019, p. 39. DOI: 10.1145/3316781.3317884. URL: https://doi.org/10.1145/3316781.3317884.

[189] V. S. Bokharaie and A. Jahanian. “Power side-channel leakage assessment and locating the exact sources of leakage at the early stages of ASIC design process”. In: (2022). DOI: 10.1007/s11227-021-03927-w. URL: https://doi.org/10.1007/s11227-021-03927-w.

[190] T. Karras, T. Aila, S. Laine, and J. Lehtinen. “Progressive Growing of GANs for Improved Quality, Stability, and Variation”. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL: https://openreview.net/forum?id=Hk99zCeAb.

[191] A. Harell, R. Jones, S. Makonin, and I. V. Bajic. “PowerGAN: Synthesizing Appliance Power Signatures Using Generative Adversarial Networks”. In: *CoRR* abs/2007.13645 (2020). arXiv: 2007.13645. URL: https://arxiv.org/abs/2007.13645.

Curriculum Vitæ

Abdullah Alawi M. Aljuffri

03-08-1985 Born in Jeddah, Saudi Arabia.

Education

2018-2024 PhD. degree in Computer Engineering
Delft University of Technology
Thesis: Securing Power Side Channels by Design
Promotors: Prof. dr. ir. Said Hamdioui, Dr. ir. Mottaqiallah Taouil

2015-2018 M.Sc. degree in Computer Engineering
Delft University of Technology
Thesis: Exploring Deep Learning For Side Channels Analysis
Supervisors: Prof. Said Hamdioui

2004-2010 B.Sc. degree in Computer Engineering
King Abdulaziz University

List of Publications

12. **A. Aljuffri**, R. Huang, L. Muntenaar, G. Gaydadjiev, S. Hamdioui, K. Ma and M. Taouil: Security Evaluation of an Efficient Lightweight AES Accelerator. submitted to Cryptogr. (2024)

11. **A. Aljuffri**, R. Huang, S. Hamdioui, K. Ma and M. Taouil, "Securing an Efficient Lightweight AES Accelerator," 2023 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Exter, UK, 2023, pp. xxx, doi: xxxx

10. **A. Aljuffri**, M. Saxena, C. R. W. Reinbrecht, S. Hamdioui and M. Taouil," A Pre-Silicon Power Leakage Assessment Based on Generative Adversarial Networks," 2023 26th Euromicro Conference on Digital System Design (DSD), Durres, Albainia, 2023, pp. xxx-xxx, doi: xxxxxx

9. **A. Aljuffri**, C. Reinbrecht, S. Hamdioui, M. Taouil and J. Sepúlveda, "Balanced Dual-Mask Protection Scheme for GIFT Cipher Against Power Attacks," 2022 IEEE 40th VLSI Test Symposium (VTS), San Diego, CA, USA, 2022, pp. 1-6, doi: 10.1109/VTS52500.2021.9794230.

8. **A. Aljuffri**, M. Zwalua, C. R. W. Reinbrecht, S. Hamdioui and M. Taouil, "Applying Thermal Side-Channel Attacks on Asymmetric Cryptography," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 29, no. 11,
pp. 1930-1942, Nov. 2021, doi: 10.1109/TVLSI.2021.3111407.

7. M. Taouil, **A. Aljuffri** and S. Hamdioui, "Power Side Channel Attacks: Where Are We Standing?," 2021 16th International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS), Montpellier, France, 2021,
pp. 1-6, doi: 10.1109/DTIS53253.2021.9505075.

6. L. C. Garaffa, **A. Aljuffri**, C. Reinbrecht, S. Hamdioui, M. Taouil and J. Sepulveda, "Revealing the Secrets of Spiking Neural Networks: The Case of Izhikevich Neuron," 2021 24th Euromicro Conference on Digital System Design (DSD), Palermo, Italy, 2021, pp. 514-518, doi: 10.1109/DSD53832.2021.00083.

5. **A. Aljuffri**, C. Reinbrecht, S. Hamdioui and M. Taouil, "Multi-Bit Blinding: A Countermeasure for RSA Against Side Channel Attacks," 2021 IEEE 39th VLSI Test Symposium (VTS), San Diego, CA, USA, 2021, pp. 1-6, doi: 10.1109/VTS50974.2021.9441035.

4. **A. Aljuffri**, C. Reinbrecht, S. Hamdioui and M. Taouil, "Impact of Data Pre-Processing Techniques on Deep Learning Based Power Attacks," 2021 16th International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS), Montpellier, France, 2021,
pp. 1-6, doi: 10.1109/DTIS53253.2021.9505051.

3. C. Reinbrecht, **A. Aljuffri**, S. Hamdioui, M. Taouil and J. Sepúlveda, "GRINCH: A Cache Attack against GIFT Lightweight Cipher," 2021 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 2021,
pp. 549-554, doi: 10.23919/DATE51398.2021.9474201.

2. C. Reinbrecht, **A. Aljuffri**, S. Hamdioui, M. Taouil, B. Forlin and J. Sepulveda, "Guard-NoC: A Protection Against Side-Channel Attacks for MPSoCs," 2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Limassol, Cyprus, 2020,
pp. 536-541, doi: 10.1109/ISVLSI49217.2020.000-1.

1. **Aljuffri**, A., Venkatachalam, P., Reinbrecht, C., Hamdioui, S., Taouil, M. (2020). S-NET: A Confusion Based Countermeasure Against Power Attacks for SBOX. In: Orailoglu, A., Jung, M., Reichenbach, M. (eds) Embedded Computer Systems: Architectures, Modeling, and Simulation. SAMOS 2020. Lecture Notes in Computer Science, vol 12471. Springer, Cham. https:doi.org/10.1007/978-3-030-60939-9_20