Application of Machine Learning in the Domain of Side-Channel Attacks

Využití strojového učení při útocích postranním kanálem

Bc. Aleš Ryška

Master's thesis
2024

Tomas Bata University in Zlín
Faculty of Applied Informatics

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2023/2024

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Aleš Ryška**
Osobní číslo:	**A22561**
Studijní program:	**N0613A140022 Informační technologie**
Specializace:	**Kybernetická bezpečnost**
Forma studia:	**Kombinovaná**
Téma práce:	**Využití strojového učení v oblasti útoků postranním kanálem**
Téma práce anglicky:	**Application of Machine Learning in the Domain of Side-Channel Attacks**

Zásady pro vypracování

1. Vypracujte literární rešerši se zaměřením na současné technologické možnosti útoků postranními kanály.
2. Proveďte podrobnější popis možností a efektivity útoků v oblasti mikrokontrolérů.
3. Získejte vhodná data reprezentující průběh či výsledky útoku postranními kanály.
4. Zvolte jednu nebo více vhodných A.I. metod z oblasti strojového učení.
5. Implementujte A.I. model pro analýzu získaných dat z útoků postranními kanály.
6. Popište potenciál a efektivitu navrženého přístupu, možnosti dalšího rozvoje či nastavení a v případě více modelů také proveďte porovnání.

Forma zpracování diplomové práce: **tištěná/elektronická**
Jazyk zpracování: **Angličtina**

Seznam doporučené literatury:

1. KOCHER, Paul, et al. Introduction to differential power analysis. *Journal of Cryptographic Engineering*, 2011, 1: 5-27.
2. SKLAVOS, Nicolas; CHAVES, Ricardo; DI NATALI, Giogio a REGAZZONI, Francesco. *Hardware security and trust: design and deployment of integrated circuits in a threatened environment*. Cham, Switzerland: Springer, [2017].
3. HETTWER, Benjamin; GEHRER, Stefan; GÜNEYSU, Tim. Applications of machine learning techniques in side-channel attacks: a survey. *Journal of Cryptographic Engineering*, 2020, 10: 135-162.
4. WU, Lichao, et al. Label Correlation in Deep Learning-based Side-channel Analysis. *IEEE Transactions on Information Forensics and Security*, 2023.
5. BURSZTEIN, Elie, et al. Generic Attacks against Cryptographic Hardware through Long-Range Deep Learning. *arXiv preprint arXiv:2306.07249*, 2023.
6. BURSZTEIN, Elie; PICOD, Jean-Michel. A hacker guide to deep-learning based side channel attacks. *Defcon27*, 2019, 25, 2020.
7. GÉRON, Aurélien. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems*. Third edition. Beijing: O'Reilly, [2023]. ISBN 978-1-098-12597-4.

Vedoucí diplomové práce: **prof. Ing. Roman Šenkeřík, Ph.D.**
Ústav informatiky a umělé inteligence

Konzultant diplomové práce: **Ing. Alžběta Turečková**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **5. listopadu 2023**
Termín odevzdání diplomové práce: **13. května 2024**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
DĚKAN
L.S.

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 5. ledna 2024

THESIS AUTHOR STATEMENT

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomové práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky. Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně 10. 5. 2024

Aleš Ryška, v.r.

podpis autora

ABSTRAKT

Tato diplomová práce se zabývá integraci strojového učení při útoku postranním kanálem za účelem obnovení klíče. Byl navržen přístup založený na hlubokém učení s využitím neuronových sítí, který umožňuje extrahovat kryptografická tajemství na základě úniku informací z postranních kanálů. Metodologie zahrnuje sběr dat, extrakci funkcí a trénování modelu s optimalizací parametrů a ověřováním. Zhodnocení nástroje proběhlo na simulovaných i reálných datech.

Klíčová slova: Strojové učení, Útok postranním kanálem, obnovení klíče, hluboké učení, kryptografie, únik dat, kryptografická tajemství, trénování modelu, kyberbezpečnost, bezpečnost dat, extrakce klíče, únik informací, optimalizace, simulace

ABSTRACT

This thesis deals with the integration of machine learning in side-channel attacks for key recovery. A deep learning-based approach using neural networks is proposed to extract cryptographic secrets based on information leakage from side channels. The methodology includes data collection, feature extraction, model training with parameter optimization, and validation.

Keywords: Machine Learning, Side-Channel Analysis, Key Recovery, Deep Learning, Cryptography, Data Leakage, Cryptographic Secrets, Model Training, Cybersecurity, Data Privacy, Key Extraction, Information Leakage, Optimization, Simulation

"We all die. The goal isn't to live forever, the goal is to create something that will." Chuck Palahniuk

TABLE OF CONTENTS

INTRODUCTION

The ever-growing reliance on technology necessitates robust security measures to protect sensitive information. However, malicious actors constantly seek vulnerabilities, and side-channel attacks pose a significant threat to modern devices. These attacks exploit unintended data leakage, often through power consumption or electromagnetic emanations, to extract confidential data. While traditional methods are employed to counter such attacks, advancements in machine learning (ML) offer promising avenues for enhanced security. This thesis explores the feasibility of utilizing a research approach focusing on regular expressions for side-channel analysis, drawing upon current technological advancements. The focus is on microcontrollers, given their widespread application in various critical systems, ranging from embedded devices to industrial control units.

This research investigated to: Delineate the vulnerabilities and effectiveness of existing side-channel attacks targeting devices. This will involve a thorough examination of current attack vectors and their potential impact. Acquire and analyze representative data associated with the power consumption or other relevant side-channel emissions during the execution of the chosen attack. This data will serve as the foundation for subsequent ML-based analysis. This will involve exploring ML methods and potential for extracting valuable information. Evaluate the potential and effectiveness of the proposed approach, including its broader applicability and potential for further development.

I. THEORY

1 Evolution of Side-Channel Attacks

The Rise of Digital Electronics (1960s-1970s) The 1960s and 1970s marked a transformative era in electronics with the rise of digital technologies. It was during this time that engineers and researchers first started to observe peculiar signals emitted by these devices. These signals, initially considered as mere noise or unintended consequences, would later become the focus of extensive research.

The foundation of side-channel attacks, a formidable category of cybersecurity threats, can be traced back to the unassuming yet critical period between the 1960s and the 1990s. Next chapters delves into the early exploration of side-channel attacks when researchers and engineers began to uncover the unintentional signals emitted by electronic devices during their operation. Image 1.1 simply describes the overview about attack vectors [14, 26].

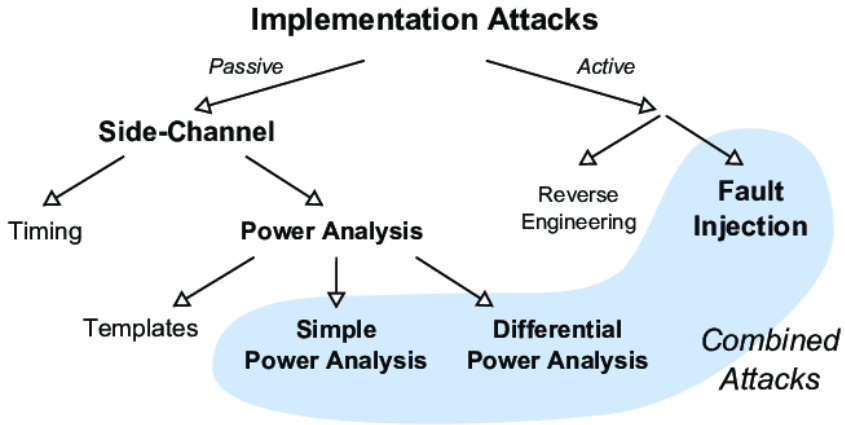

Fig. 1.1 Overview about attack vectors

1.1 Electromagnetic Radiation

Electromagnetic interference (EMI) attacks pose a significant threat to the integrity of microcontroller operations. The most common consequence is a bit-flip, where a single data bit within the microcontroller's memory is inadvertently inverted. This seemingly minor change can lead to corrupted data, erroneous instruction execution, and ultimately unexpected system behavior. In more severe scenarios, EMI might disrupt the microcontroller's clock signal, causing it to skip instructions entirely. This

disrupts critical operations and potentially bypasses security checks. In the worst-case scenario, a strong EMI attack can entirely crash the microcontroller, leading to complete system failure, this attack is shown at image 1.2 [26, 3].

Fig. 1.2 Illustration of Electromagnetic Radiation [28]

The specific consequences of an EMI attack depend on the targeted microcontroller, the nature of the attack itself, and the overall system function. However, it is crucial to recognize that even a single bit-flip can be catastrophic in security-sensitive applications. Such an event could allow attackers to extract sensitive data or compromise cryptographic keys. Fortunately, there are several strategies to mitigate the risk of EMI attacks on microcontrollers. Shielding the microcontroller with a metal casing significantly attenuates incoming electromagnetic fields. Additionally, implementing filtering circuits on the microcontroller's power supply and input/output lines helps to reduce the impact of transient voltage fluctuations. Error detection and correction (EDAC) techniques, employing error-correcting codes and redundant data storage, allow the microcontroller to detect and potentially rectify bit-flips caused by EMI. Finally, careful design practices at the microcontroller level can inherently reduce susceptibility to electromagnetic interference [3]. By implementing a combination of these mitigation strategies, system designers can significantly reduce the vulnerability of microcontrollers to EMI attacks and ensure the security and integrity of critical systems [26].

1.2 Acoustic Signals

Acoustic attacks present another avenue for disrupting microcontrollers, exploiting their susceptibility to high-powered sound waves. Unlike EMI attacks using radio waves, acoustic attacks leverage sound to cause malfunctions. There are two main ways

sound disrupts these devices: physical disruption and resonance induction. Physical disruption involves high-powered sound waves causing vibrations that misalign delicate components or create fractures, disrupting electrical pathways. Resonance induction occurs when sound waves at specific frequencies resonate with internal components, affecting electrical signal timing and potentially causing bit-flips or data processing errors. ional, caused by loud machinery or ultrasonic cleaning devices. The consequences of successful acoustic attacks mirror those of EMI attacks: bit-flips leading to corrupted data and unexpected behavior, instruction errors causing malfunctions, and even complete system crashes in severe cases. The severity depends on the microcontroller's design, sound intensity and frequency, and the environment [11]. Fortunately, there are ways to mitigate acoustic attacks. Encasing the microcontroller in a sturdy, vibration-dampening material reduces the impact of sound waves. Additionally, enclosures can be designed to shield against specific problematic frequencies. Similar to EMI attacks, error correction techniques can help identify and potentially fix bit-flips. Finally, microcontroller design practices can inherently reduce susceptibility to vibrations and sound-induced electrical noise [26, 3, 11]. By implementing a combination of these mitigation strategies, system designers can fortify microcontrollers against acoustic attacks and ensure the reliable operation of critical systems.

1.3 Early Security Implications

While the primary focus during this era was on understanding and controlling unintended signals, a select group of researchers started to recognize the nascent security implications. Discussions emerged regarding the possibility of exploiting these unintended signals for security breaches. However, these concerns were largely confined to a niche community, and there was minimal awareness in the broader field of cybersecurity.

It highlights the foundational research conducted during the 1960s to the 1990s, a period when the unintended signals emitted by electronic devices were identified and categorized. Chapter serves as the historical backdrop for the subsequent evolution of side-channel attacks into a potent threat to information security.

1.3.1 The Thing

The discovery of a covert listening device within the premises of the American Embassy in Moscow stands as a significant event in the espionage and diplomatic history. Known

colloquially as "The Thing," this device was ingeniously concealed within a wooden carving of the Great Seal of the United States, which had been presented as a gift by the Soviet Union to the American Ambassador. This episode not only highlights the ingenuity and audacity of espionage tactics during the Cold War but also underscores the perpetual game of cat and mouse between global superpowers [29].

The Discovery of The Thing The Great Seal bug, later dubbed "The Thing," was a remarkable piece of espionage technology for its time. Discovered in the late 1940s, it was hidden within a seemingly innocuous wooden plaque that adorned the wall of the U.S. Ambassador's study in the Moscow Embassy. The device's discovery was accidental, coming to light only when a routine sweep for electronic bugs detected unusual signals emanating from the Ambassador's office [29].

Technical Ingenuity The Thing (shown at images 1.3 and 1.4) was a marvel of engineering, especially considering the era of its creation. It consisted of a passive resonant cavity microphone that did not require any power source to operate. Instead, it was activated by radio waves transmitted from an external source. When these radio waves were aimed at The Thing, they were modulated by the vibrations of sound waves hitting the microphone diaphragm within the device, thereby transmitting sound back to the Soviets without the need for any internal power source [29].

Fig. 1.3 Replica of The Thing displayed at the NSA National Cryptologic Museum [29].

Fig. 1.4 The seal opened to reveal the hidden microphone [29].

Implications for Diplomatic Security The discovery of The Thing had profound implications for diplomatic security protocols. It served as a stark reminder of the vulnerabilities embassies face and the lengths to which adversaries would go to gather intelligence. In response, significant advancements were made in technical surveillance countermeasures (TSCM), leading to more sophisticated methods for securing diplomatic communications and premises. The Great Seal bug episode is a testament to the intricate dance of espionage and counterespionage that characterized the Cold War era. It exemplifies the constant evolution of surveillance technology and the ongoing need for vigilance in the protection of national security interests. The ingenuity of

The Thing and its impact on diplomatic security measures continue to be studied by security professionals and historians alike, serving as a compelling case study in the field of international relations and espionage [29].

1.4 AES Attack (2002)

In 2002, researchers demonstrated a successful side-channel attack on the Advanced Encryption Standard (AES-128). This attack showed that even widely adopted cryptographic algorithms were vulnerable to side-channel analysis.

The year 2002 marked a significant milestone in the history of cryptographic attacks when researchers unveiled a successful side-channel attack on the Advanced Encryption Standard (AES-128). For example attack on AES, at image 1.5 is a simple leakage model of this cipher [14, 2, 25].

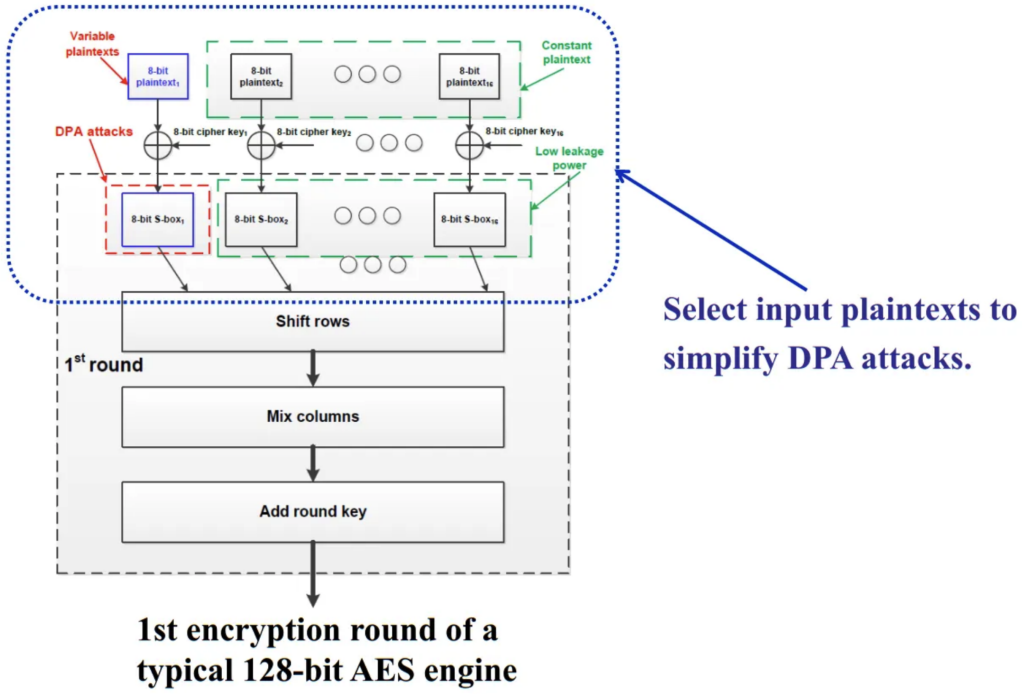

Fig. 1.5 AES Attack [30]

In this model could be seen that the power consumption of the device is directly related to the data being processed. Leakage is dependend on the moment of the encryption process. After measurement a lot of power traces, that could be used statistical analysis to find the correct key. Correlation between power traces and cipher structure could be seen at figure 1.6.

To clarify and give some abstract what is happening in the power traces, the high level

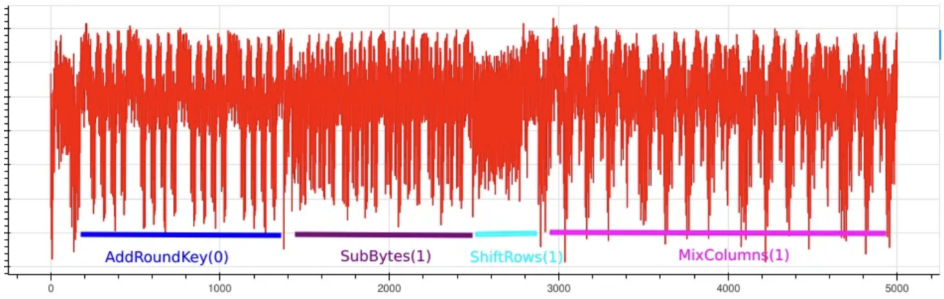

Fig. 1.6 AES Power Traces [30]

model of the AES encryption process at 1.5. One of the most important findings is that the individual steps of the AES algorithm (Add round key, Mix columns, Shift rows) could been observed as a pattern in captured power traces.

The Emergence of AES The Advanced Encryption Standard (AES) was introduced in 2001 as a replacement for the aging Data Encryption Standard (DES). AES was designed to provide a high level of security while being efficient in terms of both software and hardware implementation. Its adoption quickly spread across various industries and applications, including secure communication and data protection [2].

1.4.1 Detailed view at the attack

Team of researchers, led by Eli Biham and Adi Shamir presented a novel side-channel attack on AES. This attack was rooted in the principles of differential power analysis (DPA) and marked the first successful practical attack on the AES cryptographic algorithm. The key aspects of this attack include:

Power Analysis Similar to DPA, the AES attack in 2002 leveraged variations in power consumption during the execution of the AES algorithm. By carefully measuring these fluctuations, the researchers were able to gain insights into the internal workings of the algorithm.

Correlation Analysis The attack relied on advanced statistical techniques, particularly correlation analysis, to distinguish between the power consumption patterns associated with different AES key bytes. This analysis allowed the attackers to recover the secret key with high accuracy [14].

1.4.2 Implications and Significance

The success of the AES attack in 2002 sent shockwaves through the cryptographic community and had several far-reaching implications:

Cryptographic Standard Vulnerability The attack demonstrated that even well-regarded cryptographic standards like AES were not immune to the attacks focused on hardware. It underscored the importance of thoroughly evaluating cryptographic algorithms for potential side-channel vulnerabilities.

Need for Secure Implementations The attack highlighted the critical role of secure cryptographic implementations. While AES itself remained strong, vulnerabilities

could emerge if it was not implemented securely, emphasizing the importance of adhering to best practices.

1.4.3 Response and Countermeasures

In response to the AES attack in 2002, the cryptographic community and industry experts intensified efforts to develop countermeasures and secure implementations [25].

Enhanced Hardware Security Hardware security modules (HSMs) and Secure Elements was more important topic than before. Secure hardware design principles were further refined to protect cryptographic keys and operations from side-channel attacks.

Algorithmic Countermeasures Researchers explored techniques such as masking and blinding to mitigate the risk of power analysis attacks. The AES attack in 2002 serves as a critical milestone in the field of side-channel attacks, highlighting the need for robust cryptographic standards and secure implementations. Understanding this attack is instrumental in appreciating the ongoing efforts to secure cryptographic systems in an ever-evolving threat landscape [14].

1.5 Public Awareness and Countermeasures

As awareness of side-channel attacks grew, researchers and industry professionals began developing countermeasures to mitigate these threats. This included techniques such as blinding, masking, and threshold implementations to protect cryptographic implementations from leaking information.

As side-channel attacks gained prominence in the cybersecurity landscape, public awareness became a pivotal factor in mitigating these threats. The journey towards understanding, addressing, and ultimately neutralizing side-channel vulnerabilities has been marked by a collaborative effort involving educational initiatives, industry outreach, and the development of robust countermeasures. [21].

1.6 Raising Public Awareness

Public awareness about side-channel attacks has been a cornerstone of the defense against them. Educational initiatives have played a vital role in disseminating knowl-

edge about these attacks. Workshops, conferences, and academic programs have been established to educate professionals and students alike about the intricacies of side-channel vulnerabilities. Through these efforts, a growing number of individuals have become informed about the potential risks these attacks pose.

Industry stakeholders also took proactive measures to raise awareness. Manufacturers, developers, and vendors engaged in outreach programs to inform their user base about the potential risks associated with side-channel attacks. They emphasized the importance of secure implementations and the need for regular updates and patches to mitigate these threats.

Moreover, high-profile incidents of side-channel attacks were disclosed responsibly, contributing to increased awareness. Case studies detailing real-world scenarios served as educational tools, effectively highlighting the urgency of proactive security measures [21] [19]. One of the most interesting attack from last days was attack on Trezor (Cryptocurrency Wallet) [22].

1.7 Standardization and Best Practices

Addressing side-channel vulnerabilities required the establishment of clear guidelines and best practices for implementing cryptography securely. Standardization bodies and industry groups played a crucial role in formulating these guidelines. They covered various aspects, including algorithmic choices, key management, and secure coding practices. These standards became the foundation for secure cryptographic implementations. Secure coding standards, in particular, became imperative. These guidelines instructed developers on how to implement cryptographic algorithms in a way that minimizes the leakage of sensitive information through side channels. Certification programs were introduced to validate and attest to the security robustness of cryptographic implementations. Organizations could now seek certification to assure their stakeholders that their systems were adequately protected against side-channel threats [21] [19].

1.8 Technological Advancements

Manufacturers and designers recognized the importance of secure hardware designs. These designs aimed to minimize information leakage through side channels. Manufacturers integrated countermeasures directly into hardware components to bolster security. Innovations in cryptography also emerged as a significant defense strat-

egy. Cryptographers explored techniques that inherently resist side-channel attacks. Threshold implementations and homomorphic encryption, for example, were cryptographic paradigms designed to mitigate information leakage and enhance security. Dynamic security solutions entered the scene, capable of adapting to emerging side-channel threats. These solutions often involve continuous monitoring and adjustment of security parameters based on the evolving threat landscape. This dynamic approach provided an additional layer of protection against side-channel attacks [21].

1.9 Challenges and Ongoing Efforts

Despite the progress made, challenges persist. Small and medium-sized enterprises faced hurdles in implementing robust security measures due to resource constraints. Efforts were made to provide accessible resources and tools for secure implementation, ensuring that security was not exclusive to large organizations. Recognizing that side-channel attacks are a global concern, international collaboration efforts were initiated. Information sharing, joint research endeavors, and the establishment of global standards aimed to create a unified front against these threats. This collaborative approach seeks to address side-channel vulnerabilities comprehensively. Public awareness and countermeasures have been integral in safeguarding against side-channel attacks. The collaborative efforts of educational institutions, industry stakeholders, and standardization bodies have significantly contributed to the development of a more secure cyberspace. As technology continues to advance, maintaining a high level of public awareness and proactive countermeasures remains essential in safeguarding against the ever-evolving landscape of side-channel threats [21].

2 State of the Art: Side-Channel Analysis Research

Research into side-channel attacks continues to advance. As technology evolves, new vulnerabilities and attack techniques emerge, necessitating ongoing efforts to understand, detect, and defend against side-channel threats.

The field of side-channel attacks remains dynamic, with ongoing research efforts continuously pushing the boundaries of knowledge and understanding [21].

2.1 Overview of Current Research Landscape

Side-channel attacks represent a constantly evolving threat in the cybersecurity landscape. Researchers tirelessly pursue understanding, mitigating, and staying ahead of these attacks. This chapter explores the current state of side-channel attack research, delving into the latest trends, innovative techniques, and promising areas that will shape the future of system security [4, 21].

2.1.1 Emerging Trends in Side-Channel Attack Techniques

The research landscape is abuse with activity across several key areas. An emerging trend involves the use of advanced attack techniques such as machine learning and artificial intelligence to automate the process of extracting cryptographic keys from side-channel information, potentially making attacks more efficient and successful. Additionally, researchers are exploring novel cross-device attacks that exploit interactions between multiple devices within a system, such as using information leaked from a computer monitor's side channel to target a nearby smartphone.

2.1.2 Cryptographic Solutions

With the threat of quantum computers, researchers are diligently studying side-channel vulnerabilities in post-quantum cryptography algorithms and developing countermeasures to ensure their robustness. The security of blockchain systems is also under scrutiny, as vulnerabilities in cryptocurrency wallets and transactions could lead to significant financial implications, driving research in this area.

2.2 Hardware Vulnerabilities

Hardware vulnerabilities remain a critical research area. From microarchitectural attacks and cache timing attacks to threats like Rowhammer [4], researchers are probing hardware for exploitable weaknesses. As quantum computing advances, developing hardware security components that are resistant to quantum-based side-channel attacks is becoming increasingly crucial [4, 21].

IoT and Embedded System Challenges IoT and embedded systems introduce unique challenges due to tight energy constraints. Researchers are investigating energy-based side channels as potential avenues for attacks and devising countermeasures. Additionally, the growing concern surrounding backdoors and malicious modifications embedded within hardware components is prompting research into detection and prevention methods for these threats [4, 21].

Ethical and Legal Considerations Research in side-channel attacks extends beyond technical considerations to include ethical and legal implications. Privacy concerns surrounding the use of side-channel attacks on biometric data and personal information are being addressed, alongside the development of legal frameworks and policies to govern side-channel attacks and their potential consequences. Interdisciplinary efforts that bring together researchers from cryptography, hardware engineering, and machine learning are fostering innovative solutions. Open-source initiatives are crucial for knowledge sharing and advancing the field.

3 Power Analysis

Power analysis constitutes a critical area of study in cryptographic security, particularly concerning the assessment of vulnerabilities in cryptographic devices. This chapter focuses on Differential Power Analysis (DPA) and Correlation Power Analysis (CPA), two sophisticated techniques that exploit variations in power consumption to extract cryptographic keys from secure electronic systems. Understanding these methods is imperative for designing robust cryptographic defenses.

3.1 Correlation Power Analysis (CPA)

Correlation Power Analysis (CPA) emerges as a sophisticated side-channel attack (SCA) technique within the realm of cryptanalysis. Unlike traditional methods that focus solely on the cryptographic algorithm's vulnerabilities, CPA exploits information leakage from the physical implementation of a device during cryptographic operations. This leakage can manifest in various forms, such as power consumption, electromagnetic radiation, or timing variations. In CPA, the attacker specifically targets the relationship between the intermediate values manipulated during cryptographic operations and the power consumption of the device. The core assumption underlying CPA is that a device's power consumption fluctuates based on the number of active bits (bits with a value of 1) being processed at a given time. Cryptographic operations often involve extensive bit manipulations, and the switching activity of these bits can influence the device's power draw. This relationship between intermediate values and power consumption forms the basis for a CPA attack, which can be broken down into several key steps.

Data Acquisition: The attacker initiates the attack by collecting power traces while the targeted cryptographic device executes the same operation with different plaintexts but the same secret key. A power trace essentially represents a recording of the device's power consumption over time.

Hypothesizing Intermediate Values: The attacker then selects a hypothesized intermediate value within the cryptographic algorithm that they believe might be correlated

with the power consumption. This hypothesis could be based on the attacker's knowledge of the algorithm or educated guesses informed by the algorithm's structure.

Pre-processing the Traces: The collected power traces are pre-processed to reduce noise and enhance the signal of interest. This pre-processing might involve filtering techniques or data alignment to compensate for variations and inconsistencies between the traces.

Correlation Analysis: For each power trace, the attacker calculates a correlation coefficient between the hypothesized intermediate value and the power consumption values at corresponding time points in the trace. The correlation coefficient is a statistical measure that indicates the strength and direction of a linear relationship between two variables.

Key Recovery: Finally, the attacker analyzes the correlation coefficients obtained for all the traces. Traces where the secret key leads to a high correlation between the hypothesized intermediate value and the power consumption are considered informative. By statistically analyzing these informative traces, the attacker might be able to recover the secret key. While CPA offers advantages like requiring minimal knowledge of the specific cryptographic algorithm and being relatively easy to implement with the right tools and expertise, it also has limitations. CPA may not be effective against ciphers with low power consumption variations or masking countermeasures implemented to thwart such attacks. Additionally, a statistically significant number of power traces are often required for reliable key recovery. Despite these limitations, CPA has been demonstrated to be successful against various real-world cryptographic implementations, including hardware encryption modules and smart cards. This highlights the importance of considering side-channel vulnerabilities, like those exploited by CPA, when designing and deploying cryptographic systems, especially in scenarios where protecting the confidentiality of secret keys is paramount [13, 6, 4].

3.2 Differential Power Analysis (DPA)

In the late 1990s, researchers Paul Kocher, Joshua Jaffe, and Benjamin Jun published a groundbreaking paper on Differential Power Analysis (DPA). DPA is a side-channel attack that analyzes variations in a device's power consumption to extract cryptographic

keys and sensitive data. This marked the emergence of side-channel attacks as a practical threat. In the realm of side-channel attacks, Differential Power Analysis (DPA) has emerged as a powerful and widely recognized technique for extracting cryptographic secrets. This chapter explores the foundational principles of DPA, its historical context, and its profound implications for the security of cryptographic systems [13, 15].

Differential Power Analysis (DPA) constitutes a cornerstone technique within the domain of side-channel analysis (SCA) for cryptanalysis. Unlike conventional cryptanalysis that probes vulnerabilities within the cryptographic algorithm itself, DPA capitalizes on information leakage emanating from the physical implementation of a device during cryptographic operations. This leakage can manifest in various forms, including power consumption fluctuations, electromagnetic emanations, or timing variations [15]. In a DPA attack, the adversary specifically targets the statistical discrepancies in power consumption that arise due to the processing of meticulously chosen data patterns within the targeted cryptographic algorithm. The fundamental principle leverages the observation that identical cryptographic operations performed on data inputs with slight variations, often differing by a single bit, can induce variations in the intermediate values processed. These fluctuations in intermediate values can subsequently lead to discernible power consumption variations due to the inherent dependence of a device's power draw on the number of active bits being manipulated at a given time [13].

The Principles of Differential Power Analysis capitalizes on the physical properties of electronic devices, particularly their power consumption, to reveal cryptographic keys and sensitive data. At its core, DPA relies on the concept that the power consumed by a device varies depending on the data being processed [13, 15].

Power Consumption Variations Electronic devices exhibit varying power consumption patterns based on the computational operations they perform. For example, different bits of a secret key may cause distinct power spikes or fluctuations.

Statistical Analysis DPA leverages statistical techniques to analyze these power consumption variations. By measuring and comparing power traces for different inputs or key values, it becomes possible to identify patterns and correlations.

Key Recovery Through sophisticated statistical analysis, DPA can discern which bits of a cryptographic key are likely to be correct, ultimately leading to the recovery of the entire key. This process is iterative and involves accumulating evidence over multiple measurements to refine the key guess.

3.3 A DPA attack can be segmented into several key stages

Data Acquisition: The attacker initiates the attack by collecting a substantial quantity of power traces while the targeted cryptographic device executes the cryptographic operation under investigation. Each power trace represents a recording of the device's power consumption over time during a single execution of the operation, with different chosen plaintexts being used for each trace.

Data Selection: The attacker meticulously selects pairs of power traces where the corresponding chosen plaintexts differ in only a single bit position. This selection process aims to isolate the effect of the single-bit difference on the intermediate values and the subsequent power consumption variations.

Hypothesis Generation: The attacker formulates a well-defined hypothesis regarding the relationship between a specific intermediate value within the cryptographic algorithm and the observed power consumption patterns. This hypothesis could be based on the attacker's knowledge of the algorithm or through informed guesses about the data processing steps within the algorithm.

Differential Power Analysis: For each meticulously chosen pair of traces, the attacker performs a differential power analysis operation. This involves subtracting the corresponding power consumption values at each time point between the two traces, resulting in a differential power trace. This differential trace is then compared with a pre-computed template derived from the attacker's hypothesis about the targeted intermediate value. The template represents the anticipated power consumption variations associated with the hypothesized bit transitions within the intermediate value [13].

Statistical Analysis: The attacker employs rigorous statistical analysis techniques to evaluate the results of the differential power analysis across all the chosen plaintext

pairs. Traces where the secret key leads to a high correlation between the differential power trace and the template are considered informative. By statistically evaluating these informative traces, the attacker might be able to exploit the observed power consumption patterns to recover the secret key.

3.3.1 Advantages and disadvantages of DPA

Advantages: DPA offers a more potent approach compared to rudimentary power analysis techniques by leveraging a robust statistical framework. DPA can be effective against ciphers exhibiting low power consumption variations, making it a more versatile tool compared to Correlation Power Analysis (CPA) [13].

Disadvantages: DPA necessitates a statistically significant number of power traces for reliable key recovery, which can be time-consuming to acquire. For complex algorithms with numerous intermediate values, DPA can become computationally expensive. DPA has been demonstrably successful against various real-world cryptographic implementations, highlighting the criticality of considering side-channel vulnerabilities during the design phase of secure cryptographic systems. Countermeasures such as masking and hiding techniques are frequently employed to mitigate the effectiveness of DPA attacks [13].

3.4 Hamming distance

The Hamming Distance Model (HD Model) is used in Correlation Power Analysis (CPA). CPA is a type of side channel attack (SCA) that analyzes information leakage from a device, such as power consumption, to recover the secret key. The Hamming distance (HD) is the number of different bits between two data strings. In cryptography, it refers to the difference between two states in a cryptographic algorithm. The HD Model assumes a relationship between the power consumption and the HD between an intermediate value and a reference state value.

An intermediate value is a temporary value created during a cryptographic operation. The reference state value is a known value used as a reference for comparison. In the AES encryption algorithm, the ciphertext of the last round can be the reference state, while the input data of the S-Box in the last round can be the intermediate value. The HD Model is based on the idea that the larger the HD between the intermediate value and the reference state value, the greater the difference in power consumption. This difference in power consumption can be exploited by an attacker to learn information about the secret key. However, the HD Model requires knowledge of the cryptographic device's internal operations. This makes it difficult to implement in practice. For this reason, a modified version of the HD Model, called the Hamming Weight (HW) Model, is often used instead.
Equation 3.1 defines the Hamming distance between two binary strings x and y of length n:

$$\mathrm{HD}(x, y) = \sum_{i=1}^{n} \mathbf{1}_{[x_i \neq y_i]} = \|\mathbf{x} \oplus \mathbf{y}\|_1 \tag{3.1}$$

3.4.1 Hamming weight

Hamming Weight (HW) is a leakage model used in Correlation Power Analysis (CPA). It discusses the leakage of information during cryptographic operations and shows that HW leakage can be observed in power traces. In simpler terms, Hamming weight refers to the number of "1"s in a binary string. In the context of cryptography, it is used to estimate the amount of information leakage that might occur during cryptographic operations. This leakage can be exploited by attackers to recover secret keys. The Hamming weight model is a simpler alternative to the Hamming distance model, which requires knowledge of the cryptographic device's internal operations. This makes the Hamming weight model more practical for real-world applications as shown at image 3.1 and 3.2.

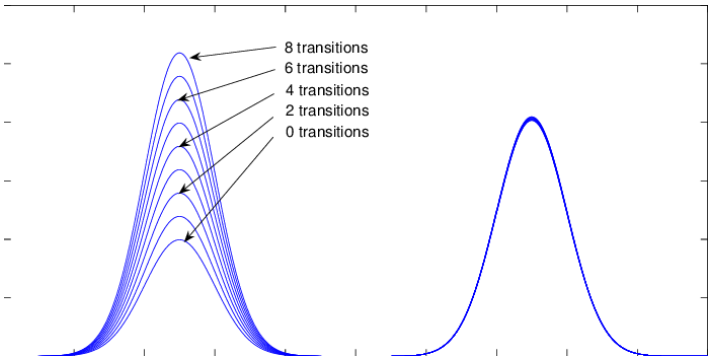

Fig. 3.1 Illustration of Hamming weight data-dependencies in the power consumption traces of a smart card using an 8-bit data bus [27].

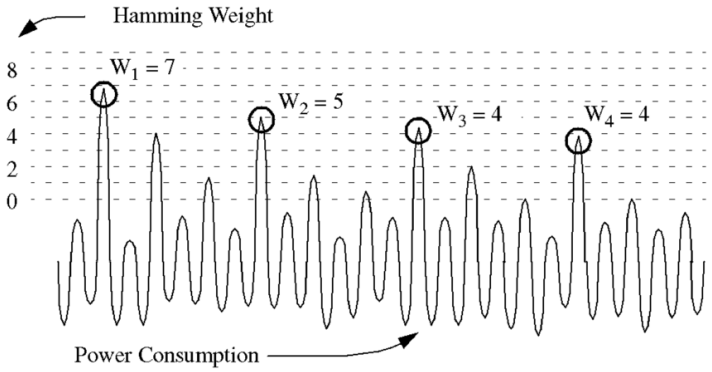

Fig. 3.2 The weight of the byte being processed is proportional to the height of the power consumption pulse [18].

4 TRIVIUM

Trivium is a synchronous stream cipher designed with a hardware implementation in mind. The core principle behind its creation was to explore the boundaries of simplification in stream cipher design without compromising security, speed, or flexibility. While simple designs can inherently be more susceptible to basic attacks (hence the strong discouragement of using Trivium at this stage), their very simplicity can inspire greater confidence compared to complex schemes, particularly if they withstand a prolonged period of public scrutiny. The following section provides a detailed description of the Trivium stream cipher proposal. Sections below offers a brief overview of its security considerations and performance [8].

4.1 The Cipher Structure

The Trivium stream cipher is a synchronous design capable of generating a keystream of length up to 2^{64} bits. This keystream is derived from an 80-bit secret key and an 80-bit initialization vector (IV) (table 4.1).

Parameters	
Key size	80 bit
IV size	80 bit
Internal state	288 bit

Tab. 4.1 TRIVIUM - cipher parameters [8]

Initialization: During this stage, the internal state of the cipher is established using the provided secret key and initialization vector.

Keystream Generation: The internal state undergoes continuous updates, and these updates serve as the basis for generating the keystream bits. The cipher internal structure describes image 4.1.

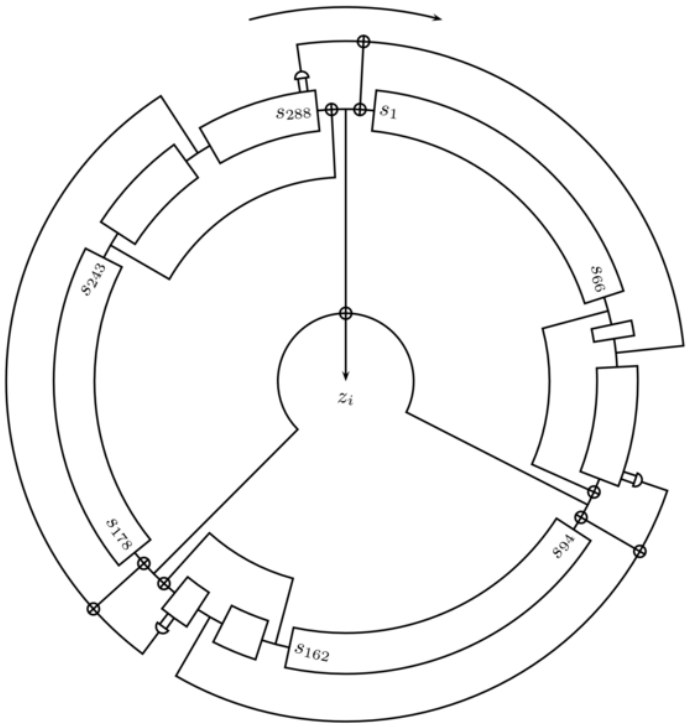

Fig. 4.1 TRIVIUM - diagram of the cipher structure [8]

4.2 Generation of keystream

The proposed Trivium design utilizes a 288-bit internal state denoted as $\mathbf{s} = (s_1, \ldots, s_{288})$. Keystream generation is achieved through an iterative process.

1. **Extracting State Bits:** A specific set of 15 state bits, denoted as $\mathbf{t} = (t_1, \ldots, t_{15})$, are extracted from the internal state during each iteration.
2. **State Update and Keystream Generation:** The extracted bits are used for two purposes:
   - Updating 3 specific bits within the internal state. This update can be represented as a function $f : \{0,1\}^{15} \to \{0,1\}^3$, where $f(\mathbf{t}) = (u_1, u_2, u_3)$.
   - Computing a single keystream bit denoted as z_i. The computation of z_i depends on a specific function $g : \{0,1\}^{15} \to \{0,1\}$ using the extracted bits, i.e., $z_i = g(\mathbf{t})$.

3. **State Rotation:** Following the update and keystream generation, the internal state undergoes a bitwise rotation operation. This can be represented as a shift function $\rho : \{0,1\}^{288} \rightarrow \{0,1\}^{288}$, where $\mathbf{s} = \rho(\mathbf{s}')$ and $\mathbf{s}'$ is the updated state after step 2.

This iterative process continues until the desired number of keystream bits ($N \leq 2^{64}$) has been generated.

The following pseudocode 1 describes the iterative process used for keystream generation, where operations are performed over GF(2), meaning that '+' and '·' represent XOR and AND operations [8].

Algorithm 1 Keystream Generation for Trivium

for $i = 1$ **to** N **do**
  $t_1 \leftarrow s_{66} + s_{93}$ ▷ XOR operation
  $t_2 \leftarrow s_{162} + s_{177}$ ▷ XOR operation
  $t_3 \leftarrow s_{243} + s_{288}$ ▷ XOR operation
  $z_i \leftarrow t_1 + t_2 + t_3$ ▷ XOR operation
  $t_1 \leftarrow t_1 + (s_{91} \cdot s_{92}) + s_{171}$ ▷ XOR and AND operations
  $t_2 \leftarrow t_2 + (s_{175} \cdot s_{176}) + s_{264}$ ▷ XOR and AND operations
  $t_3 \leftarrow t_3 + (s_{286} \cdot s_{287}) + s_{69}$ ▷ XOR and AND operations
  $(s_1, s_2, \ldots, s_{93}) \leftarrow (t_3, s_1, \ldots, s_{92})$
  $(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (t_1, s_{94}, \ldots, s_{176})$
  $(s_{178}, s_{279}, \ldots, s_{288}) \leftarrow (t_2, s_{178}, \ldots, s_{287})$
end for

Note that in this document, and the rest of the descriptions, the '+' and '·' symbols stand for addition and multiplication over GF(2) (i.e., XOR and AND).

4.3 Key and IV Setup

The algorithm is initialized by loading an 80-bit key and an 80-bit Initialization Vector (IV) into the 288-bit initial state, setting all remaining bits to 0, except for s_{286}, s_{287}, and s_{288}. Then, the state undergoes rotation over 4 full cycles, similar to the process described previously but without generating keystream bits. This process is summarized in the pseudo-code 2 [8, 16]:

Algorithm 2 Initialization of the Trivium Algorithm

1: $(s_1, s_2, \ldots, s_{93}) \leftarrow (K_1, \ldots, K_{80}, 0, \ldots, 0)$
2: $(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (IV_1, \ldots, IV_{80}, 0, \ldots, 0)$
3: $(s_{178}, s_{279}, \ldots, s_{288}) \leftarrow (0, \ldots, 0, 1, 1, 1)$
4: **for** $i = 1$ **to** 4×288 **do**
5: $\quad t_1 \leftarrow s_{66} + s_{91} \cdot s_{92} + s_{93} + s_{171}$ ▷ XOR and AND operations
6: $\quad t_2 \leftarrow s_{162} + s_{175} \cdot s_{176} + s_{177} + s_{264}$ ▷ XOR and AND operations
7: $\quad t_3 \leftarrow s_{243} + s_{286} \cdot s_{287} + s_{288} + s_{69}$ ▷ XOR and AND operations
8: $\quad (s_1, s_2, \ldots, s_{93}) \leftarrow (t_3, s_1, \ldots, s_{92})$
9: $\quad (s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (t_1, s_{94}, \ldots, s_{176})$
10: $\quad (s_{178}, s_{279}, \ldots, s_{288}) \leftarrow (t_2, s_{178}, \ldots, s_{287})$
11: **end for**

4.4 Hardware Implementation

The Trivium stream cipher is designed with a focus on hardware practicality, targeting flexibility especially in resource-constrained environments. Its design is primarily aimed at achieving three key objectives:

1. **Compactness:** Suited for scenarios with limited gate count availability.
2. **Power Efficiency:** Optimized for platforms with restricted power resources.
3. **High Speed:** Capable of delivering fast encryption for applications requiring high throughput.

The requirement for a compact implementation naturally leads to a bit-oriented approach, which also necessitates a non-linear internal state to maintain the generated non-linearity at the keystream output stage. To enable power-efficient and high-speed implementations, the design incorporates features that allow for the parallelization of operations. In Trivium, this is facilitated by ensuring that any state bit remains unused for at least 64 iterations after being modified. This design choice permits the parallel computation of up to 64 iterations, assuming that the original scheme's 3 AND gates and 11 XOR gates are replicated accordingly. Such parallelization enables a 64-fold increase in clock frequency without sacrificing overall throughput. Leveraging figures provided in 4.2 (i.e., 12 NAND gates per flip-flop, 2.5 gates per XOR, and 1.5 gates per AND), the gate count for various levels of parallelization has been estimated. The results of this estimation are presented in Table 4.2 [8].

Components	**1-bit**	**8-bit**	**16-bit**	**32-bit**	**64-bit**
Flip-flops	288	288	288	288	288
AND gates	3	24	48	96	192
XOR gates	11	88	176	352	704
NAND gates	3488	3712	3968	4480	5504

Tab. 4.2 Estimate gate counts for Trivium hardware implementation [8]

4.5 Security Considerations

The primary security objective for Trivium is to maintain resistance against cryptographic attacks at a level comparable to any theoretical stream cipher sharing its external parameters. In simpler terms, any attack on Trivium should not be demonstrably easier to mount compared to an attack on an imaginary stream cipher with the same capabilities:

- Generating up to 2^{64} bits of keystream,
- Utilizing an 80-bit secret key,
- Employing an 80-bit initialization vector (IV).

Definitively verifying this ideal level of security is a challenging task. Therefore, the following sections will present arguments that support the belief in Trivium's resilience against various cryptanalytic techniques [16].

4.5.1 Correlations

Security analysis of synchronous stream ciphers often involves investigating two distinct types of correlations:

State-Keystream Correlations: These correlations exist between linear combinations of keystream bits z_i and specific internal state bits s_i. Ideally, an attacker shouldn't be able to exploit these correlations to efficiently recover the entire internal state, which would compromise the cipher's security [8].

Keystream-Keystream Correlations: Exploited in distinguishing attacks, these correlations exist between the keystream bits themselves.

While linear correlations between keystream bits and internal state bits are readily identifiable in Trivium (due to the explicit definition of z_i), the cipher's non-linear state update mechanism (unlike LFSRs) hinders straightforward recovery of the state by combining these equations.

Finding correlations of the second type traditionally involves tracing linear paths through the cipher and approximating the outputs of encountered AND gates as 0. However, the specific tap positions within Trivium were chosen to ensure any such linear trail necessitates approximating at least 72 AND gate outputs [8]. This significantly increases the complexity of exploiting these correlations for cryptanalysis. The expression involving the sum of various terms is given by 4.1 [8]:

$$z_1+z_{16}+z_{28}+z_{43}+z_{46}+z_{55}+z_{61}+z_{73}+z_{88}+z_{124}+z_{133}+z_{142}+z_{202}+z_{211}+z_{220}+z_{289} \quad (4.1)$$

Assuming a hypothetical scenario where the observed correlation in a specific linear combination of keystream bits is solely attributable to the considered linear trail, the corresponding correlation coefficient would theoretically be 2^{-72}. Detecting such a weak correlation would necessitate analyzing a minimum of 2^{144} keystream bits, exceeding the established security requirements for Trivium. While the possibility of more intricate linear trails with stronger correlations cannot be entirely ruled out, current analysis suggests it's improbable that these correlations would surpass 2^{-40} [8].

4.5.2 Guess and Determine Attacks

In each iteration of Trivium, only a few bits of the state are used, despite the general rule-of-thumb that sparse update functions should be avoided. As a result, guess and determine attacks are certainly a concern. A straightforward attack would guess the bits $(s_{25}, \ldots, s_{93})$, $(s_{97}, \ldots, s_{177})$, and $(s_{244}, \ldots, s_{288})$, totaling 195 bits, after which the rest of the bits can immediately be determined from the keystream. Further research should be conducted to examine to what extent more sophisticated attacks can reduce this number [16].

4.5.3 Algebraic attacks

Trivium presents a potentially appealing target for cryptanalysis using algebraic attacks. The entire cipher can be readily described by a minimal set of low-degree equations. However, the cipher's state update mechanism deviates from a linear model, posing a challenge for applying the efficient linearization techniques typically employed to solve equation systems derived from LFSR-based schemes.

While alternative techniques might be applicable for solving Trivium's specific system of equations, further investigation is necessary to assess their effectiveness [8].

4.5.4 Resynchronization atacks

Resynchronization attacks pose a threat where an adversary attempts to manipulate the initialization vector (IV) and analyze the resulting keystream to extract the secret key. Trivium implements a countermeasure against such attacks by cycling the internal state a predetermined number of times before generating any keystream output.

Theoretical analysis demonstrates that after two complete cycles (comprising 2 x 288 iterations), each internal state bit exhibits a non-linear dependency on every key and IV bit. This non-linear relationship significantly increases the difficulty for an adversary to exploit the manipulated IV and recover the key [8]. Based on this analysis, it is anticipated that an additional two cycles (for a total of four cycles) would provide a sufficient level of protection against resynchronization attacks in Trivium.

II. ANALYSIS

5 SCAAML: Side Channel Attacks Assisted with Machine Learning

Cryptographic algorithms represent the foremost barrier against attacks. Nevertheless, these algorithms are vulnerable to side-channel attacks. This area of study, which presents significant potential for further research and evaluation, is exemplified by the project whose emblem is depicted in Figure 5.1.

These attacks vector inadvertent leakage of information, such as variations in power consumption, to extract confidential data.

SCAAML (Side Channel Attacks Assisted with Machine Learning), a project from Google, emerges as a powerful tool for researchers and security professionals delving into this intricate domain.

This paper presents a comprehensive exploration of SCAAML, delving into its architecture, functionalities, and broader implications [6]. Project is accessible via GitHub repository located at [9].

Fig. 5.1 SCAAML: Side Channel Attacks Assisted with Machine Learning [6]

5.1 The Architecture

SCAAML is a powerful tool built on top of TensorFlow 2.x specifically designed to analyze side-channel information in cryptography [5]. Side-channel data in this case are like how much power a device uses while performing an operation (program, encryption, instruction etc.). By analyzing this data, attackers might be able to steal

secret information. SCAAML helps researchers understand these leaks and develop better defenses [9]. One of the key strengths of SCAAML is its flexibility. It allows researchers to choose from various building blocks to create custom analysis tools.

Data Pre-processing: This stage is crucial for cleaning and preparing the raw side-channel data. Assume a scenario with hundreds of measurements a device's power consumption, but it might also include some background noise or irrelevant information. SCAAML can normalize the data (ensuring everything is on the same scale), filter out noise, and even create new features from the existing data to uncover hidden patterns [6].

Model Selection and Training: SCAAML offers a variety of deep learning model architectures, like Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs). Each type of model is good at finding different kinds of patterns in the data. Once a model is chosen, SCAAML trains it using labeled datasets. These datasets contain side-channel information paired with the secret information it reveals (like specific parts of a cryptographic key). By training on this data, the model learns to identify the link between the leak patterns and the hidden secrets [6].

Once a suitable architecture is selected, the model undergoes **training** using carefully curated **labeled datasets**. These datasets consist of side-channel data points **annotated with the corresponding secret information** they represent (e.g., specific key bytes). During training, the model learns to map the intricate relationships between the leak patterns in the data and the hidden secrets. The choice of **loss function** (a metric quantifying the model's prediction error) is crucial, as it guides the model towards optimal learning and accurate secret recovery.

5.2 Attack Execution

After training, SCAAML is ready to be used for an actual attack. The model takes in new, unlabeled side-channel data (from an unknown cryptographic operation) and tries to predict the secret information based on what it learned during training. There are different attack strategies, like template matching (comparing the data to pre-existing profiles) or score-based attacks (assigning scores to different possibilities based on how well they fit the leak patterns) [6].

5.3 Unlocking the Potential

SCAAML's integration of machine learning offers several significant advantages for side-channel analysis.

- **Efficiency:** Deep learning models can automatically find subtle patterns in data that humans might miss. This leads to faster and potentially more successful attacks compared to traditional methods.
- **Flexibility:** SCAAML's modular design allows researchers to adapt it to various scenarios. By customizing the model, training data, and attack strategy, they can explore a wider range of vulnerabilities.
- **Innovation:** SCAAML provides a platform for researchers to experiment with new ideas and explore side-channel analysis in more depth. This can ultimately lead to the development of stronger cryptographic defenses.

Researchers have employed SCAAML in various ways, extending its reach beyond its core functionalities.

- **Benchmarking:** Researchers can use SCAAML to compare different deep learning models against traditional analysis methods. This helps them to understand which approaches work best for different situations.
- **Countermeasure:** SCAAML can be used to test the effectiveness of new techniques designed to protect against side-channel attacks. By simulating attacks on these countermeasures, researchers can identify weaknesses and improve the defenses.

5.4 Datasets and models

The SCAAML project offers a powerful framework for exploring side-channel analysis, but getting started can involve a learning curve. The included *scaaml_intro* directory provides tutorial for researchers to delve into this domain. Both datasets and pre-trained models are available at project repository on GitHub [9].

Pre-Trained models: One of the significant advantages of *scaaml_intro* lies in its extensive collection of pre-trained models. Fourty pre-trained models are served as a

valuable starting point for researchers. They encompass various deep learning architectures, each optimized for specific side-channel analysis tasks. This eliminates the need to build models from scratch, saving time and allowing to focus on exploring different attack scenarios. Training all the models take about 24 hours running on GPU, with performance compared to RTX4090 [5].

The pre-trained models cater to diverse cryptographic algorithms and side-channel leak types. Whether investigating power consumption variations during AES encryption or electromagnetic emissions associated with RSA key generation, there is a good chance to find a pre-trained model suited for needs within *scaaml_intro*.

Data collection To effectively train and evaluate deep learning models for side-channel analysis, substantial amounts of data are required. Scaaml introduction recognizes this need and provides a staggering 8.2GB of datasets [9]. These datasets encompass various real-world scenarios, incorporating side-channel traces alongside the corresponding secret information they reveal (e.g., key bytes), all measured on **TinyAES** algorithm [9].

The rich diversity of these datasets allows to experiment with different attack strategies and assess the effectiveness of various pre-trained models. They can also serve as a foundation for building custom datasets tailored to specific research endeavors.

6 A Three-Step Approach To State/Key Recovery

This section is based on the research documented in the article by Kumar et al. [16]. For further steps and practical demonstrations, the repository located at [24] will be utilized.

In the realm of symmetric key cryptography, stream ciphers are critical for ensuring secure electronic communication. Unlike block ciphers, which have been extensively studied, stream ciphers have received less attention, making them a prime target for side channel attacks (SCAs). These attacks exploit physical leakages such as electromagnetic emissions or power consumption to infer cryptographic keys. Mentioned research builds upon foundational work by Sim, Bhasin, and Jap at TCHES'21, advancing a comprehensive framework that integrates several computational techniques to refine the attack on stream ciphers [16].

Symmetric key cryptography plays a pivotal role in securing modern electronic communications. Symmetric key algorithms are generally more efficient than asymmetric ones, making them preferable in protocols that support their use. Therefore, it is crucial to understand the potential threats to the security of these ciphers.

The vulnerabilities of symmetric key systems can be primarily categorized into two types. The first type, known as *classical attacks*, involves thorough analysis of the algorithm's structure. The second type exploits the physical attributes of the device executing the cipher, circumventing the sophisticated mechanisms designed to protect against classical threats [16].

Research used as reference [16] focuses on a specific category of physical attacks, known as **Side Channel Attacks (SCA)** [13, 15, 23]. These attacks involve observing the physical outputs of a device, such as timing, power consumption, and electromagnetic emissions, during the cipher operation. These observations can reveal critical information about the cipher's secret key. Another prevalent type of physical attack is the **Fault Attack (FA)** [1], which also poses significant security risks.

6.1 Research Focus and Contribution

Despite the importance of Side Channel Attacks, it appears that the focus tends to lean more towards block ciphers and similar constructs, often leaving stream ciphers and related mechanisms less examined. This work proposes a framework aimed at providing an effective model for analyzing stream ciphers through side channel attacks. Proposed framework is robust, capable of handling real, noisy data traces and supports both Hamming weight (software) and Hamming distance (hardware) leakage models. Furthermore, it remains effective even after the cipher enters its pseudo-random phase, also referred to as the key-stream phase [16].

To illustrate the significance, consider the following observations:

> "Knowledge about the Hamming weight of intermediate values can indeed help in narrowing down possible keys, thus reducing the search space for the key. However, information on Hamming weight alone often falls short of revealing the secret key [20]."

> "There is no straightforward method to deduce the internal state of the cipher post-initialization phase, where the key is distributed across 288 bits of internal state in TRIVIUM. Hence, any attempt to apply a side channel attack at a later stage seems unpromising [20]."

6.2 Methodological Framework

Focus aims to use a generic framework capable of deducing the key of a stream cipher or a similar cryptographic system based on side channel leakage, with minimal human intervention. Framework is designed to extract significant information from the leakage during the pseudo-random generation algorithm and to account for noise present in the leakage. The source code is available as open-source on the Bitbucket repository [24]. Further details about modeling approach are provided in next chapters, while the workflow is summarized in next chapter [16].

6.3 Framework Workflow

The operation of framework is divided into two main stages [24].

Offline Stage

1. Collect side channel traces from the target device. Depending on whether the leakage model is based on Hamming weight (software) or Hamming distance (hardware), these traces are treated as a multi-class classification problem. For instance, in a 32-bit microcontroller, up to 33 distinct Hamming weights are possible, defining 33 classes for classification .
2. Train a Machine Learning (ML) model suitable for classification to learn from these traces.
3. Evaluate the tolerance limit, tl, of an SMT (Satisfiability Modulo Theory) solver using simulated noisy information.

Online Stage

1. Use the trained ML model to estimate the class (i.e., Hamming weight or distance) of the target traces.
2. Fit these estimations into an SMT solver. If all ML predictions are accurate, the SMT solver efficiently returns a solution for the unknown state or key.
3. Adjust the definition of a correct prediction to include a tolerance $\epsilon > 0$. A prediction is deemed correct if $c' - \epsilon \leq c \leq c' + \epsilon$, where c' is the predicted class and c is the actual class.
4. To manage the inherent inaccuracies in ML predictions and the resulting SMT inconsistencies, employ a Mixed Integer Linear Programming (MILP) model. This model refines the sequence of ML predictions by integrating cipher-specific constraints, like the similarity in Hamming weights across consecutive clock cycles.
5. Feed the output from the MILP model back into the SMT solver, which then processes it with the established error tolerance, tl.

The naiveté of the initial ML modeling often results in low accuracy for class predictions, necessitating the rejection of many traces since the SMT solver requires high accuracy. To address this, was introduced an error tolerance concept, allowing for a pragmatic compromise between accuracy and computational feasibility [16].

6.3.1 Practical Evaluation and Results

The effectiveness of framework is demonstrated through experiments with the TRIVIUM stream cipher on an ARM Cortex-M3 board. The MILP model's corrections to the ML output sequence, followed by a description of the SMT model provides a summary of the SMT module. Experimental results focusing on the Hamming weight and distance models, respectively.

The primary innovation of framework lies in its integration of ML, MILP, and SMT under a unified system that addresses both the initialization and pseudo-random generation phases of stream ciphers.

- Automates the process of key recovery, reducing the need for human intervention.
- Enhances the handling of noisy data, a common issue in practical SCAs.
- Demonstrates practical key recovery on TRIVIUM, an ISO-standardized stream cipher, highlighting the framework's effectiveness.

Practical Application Applied framework to TRIVIUM, conducting extensive tests on a 32-bit software platform. The results underscored the framework's capability to effectively decipher keys during various cipher phases, even with noisy electromagnetic traces. This showcases not only the versatility of approach but also its potential applicability to other ciphers [16].

Recovery Procedures for Inconsistent SMT Instances The occurrence of inconsistencies in SMT instances, although rare, necessitates a robust set of strategies for effective resolution. Here are proposed several approaches:

1. **Retraining the Machine Learning Model:** Run the prediction again using the same traces but retrain the ML model with randomized initial parameters. This could potentially lead to a different sequence of predictions that may resolve the inconsistencies.

2. **Increasing the Number of Trace Rounds:** Collect traces for a significantly higher number of rounds. As demonstrated in Section 4, while over 100 rounds are generally sufficient, collecting up to 500 rounds provides additional data that may be useful if certain segments prove problematic. This approach leverages the high success probability of the MILP (Mixed Integer Linear Programming) model, often resolving issues within one or two attempts.

3. **Recollecting Traces:** If the issue persists, another viable strategy is to recollect traces from the device. However, it is important to highlight that increasing the number of classes in the MILP model reduces its success probability, and expanding the number of classes in SMT can lead to increased solution times, crossing a practical threshold. Consequently, maintaining a lower number of classes is advisable as long as the SMT instances remain solvable [16].

6.3.2 Note on State/Key Recovery

During the pseudo-random generation phase of the cipher, where the cipher is prepared to produce key-stream or tag, it is possible to recover the unknown state. Depending on the cipher's internal structure, full key recovery may be achievable. For instance, with the TRIVIUM cipher, the state is invertible, which allows it to be reverted back to the key/IV loading phase, thereby enabling full key recovery. However, in cases where the state update is not invertible, such as with the LIZARD cipher [10], full key recovery may not be feasible [16].

Experimental setup: The targeted cryptographic operation was implemented in assembly on an Arduino DUE, featuring an ARM Cortex-M3 processor, 512KB of flash memory, 96KB of SRAM, and an operating frequency of 84 MHz [16].

Leakage Capture: For the leakage capture, was employed a high-precision EM probe from Riscure, coupled with a Lecroy WaveRunner 610zi oscilloscope. Initially, a preliminary test using known fixed data was conducted to determine the optimal measurement spot on the Arduino board an these datasets are available at [24, 16].

For the profiling phase, was employed stratified sampling to ensure representation from all Hamming Weight (HW) classes. Approximately 33,000 traces were collected, with each HW class from 0 to 32 contributing about 1,000 traces. In total, around 2,362,000 traces, approximately $2^{21.17}$, were gathered for the analysis [16, 24].

7 Evaluation

In final chapter of this thesis wraps up the key, the key findings of this research on side-channel analysis (SCA) for microcontrollers, focusing on how machine learning (ML) can be used to improve attack effectiveness.

The primary challenge arises from the classification of Hamming weight or distance by the machine learning (ML) model. This difficulty is exacerbated by the close proximity among classes. With a basic implementation, the ML model's accuracy struggles to reach 40%, despite extensive efforts to enhance it. Such low accuracy presents a significant issue, as subsequent processes attempting to deduce unknown components rely on entirely accurate predictions. Even a single incorrect prediction can lead to inconsistencies throughout the entire Satisfiability Modulo Theories (SMT) system. Preliminary evaluations suggest that hundreds of predictions for a typical cipher, such as TRIVIUM [7], are required, necessitating an impractically large number of independent experiments to secure all correct predictions from the ML model [16]. To improve the accuracy of the ML model, have been observed a crucial pattern: if the correct class is c, the model is more likely to predict a class within the vicinity of c—specifically, from $c - \epsilon$ to $c + \epsilon$, where $\epsilon \geq 1$—rather than exactly c. This led to the introduction of a tolerance parameter, ϵ, which redefines accuracy as follows:

1. When $\epsilon = 0$, the ML model's accuracy is assessed strictly, requiring the prediction to match class c exactly.
2. For $\epsilon > 0$, accuracy includes any predicted class within the range from $c - \epsilon$ to $c + \epsilon$.

Increasing ϵ not only enhances the model's accuracy according to this tailored definition but also allows SMT constraints to be adjusted to accommodate more flexible predictions. Specifically, an SMT constraint of the form $x = c'$ can be modified to $c' - \epsilon \leq x \leq c' + \epsilon$, where x might represent the Hamming weight of the state, and c' is the predicted class. This strategy of introducing error tolerance effectively increases the accuracy of ML predictions and aligns well with the requirements for SMT modeling results are shown at 7.1 [16].

Tolerance (e)	**0**	**1**	**2**	**3**	**4**
ML Accuracy	0.3933	0.86678	0.98262	0.99784	0.99967
SMT Solving Time (s)	5.42	254.36	1819.21	28755.36	76797.41

Tab. 7.1 Trade-off between ML tolerance and SMT solution time (sec.) for TRIVIUM [16]

7.1 ML: Prediction Classes From Traces

Simulation data was from collected 1,000 traces for each HW class, resulting in a total of 33,000 traces. For the machine learning process, including training, validation, and testing phases, the dataset size reached approximately $2.36 \times 10^6 \approx 2^{21.17}$ traces. It is important to note that while all classes are represented, the dataset is inherently imbalanced with unequal class distribution [16].

Model Configuration: The Multi-Layer Perceptron (MLP) model was selected for its simplicity, configured with input and output layers of sizes 500 and 33, respectively. The data was split in a 62.5/12.5/25 ratio for training, validation, and testing. Models were trained using Python 3.8.5, PyTorch 1.8.1+cu102, and Numpy 1.20.3 on an Intel Xeon Platinum 8260 CPU and NVIDIA Quadro GV100 GPU setup [16].

Training Details ReLU activation functions were utilized post-hidden layers, with the AdamW optimizer set at a learning rate of 0.0001, incorporating a weighted cross-entropy loss function to address class imbalance [17]. Training was halted prematurely if overfitting was detected [16].

Hyper-parameter Optimization For hyper-parameter tuning was choosen Optuna, a framework for automatic optimization [12]. The process aimed to identify the optimal configuration for the number of hidden layers and their sizes using a dataset of approximately $2^{19.345}$, with an 80/20 training/validation split. Despite numerous trials (n=500), the optimal model suggested by Optuna, featuring three hidden layers each of size 416, yielded results similar to simpler two-layer model [16].

7.2 Model Performance

The performance of the MLP model with two hidden layers, each with 128 neurons, was found to be optimal when compared to more complex models suggested by Optuna. The

accuracy of this model, with zero tolerance for error, approached 40%. These results are shown in table 7.2, highlighting the average accuracy rounded to five decimal places for different data splits.

The findings suggest that simpler models may be adequate for profiling side-channel attacks in practical scenarios, avoiding the complexity and computational overhead associated with more extensive hyper-parameter tuning. This efficiency is crucial in deploying machine learning for real-time SCA applications [16].

Model	**Train. Acc**	**Val. Acc.**	**Testing Accuracy with Tolerance (ϵ)**								**Train Time (s)**
			0	1	2	3	4	5	6	≥ 7	
MLP-I	0.3601	0.39389	0.3933	0.86678	0.98262	0.99784	0.99967	0.99991	0.99998	1.0	5530.93
MLP-II	0.3661	0.3931	0.39266	0.86615	0.98234	0.99784	0.99965	0.99992	0.99999	1.0	6340.84

Tab. 7.2 Results for machine learning prediction (with varying tolerance) [16]

The experiments were conducted under different HW models as described. Examined the SMT solver's performance under various error tolerances and the number of rounds, which were initially set to approximately half of the original number of variables (110 rounds).

Solution Time and Error Tolerance The solution times for unique solutions under the HW/8, HW/16, and HW/32 models with varying error tolerances as shown at table 7.3. The key distinctions between results with only update and HW equations versus those including key-stream equations are also presented [16].

- Under the HW/8 model, achieved state bit recovery up to an error tolerance of 3. Beyond this tolerance, the Z3 solver returned multiple solutions, requiring increased computation times even with a higher number of rounds.
- For the HW/16 model, solutions were found up to an error tolerance of 4 within approximately 1–2 hours.
- Detailed results for the 32-bit microcontroller are reported in Tables 7.2 and 7.4, noting that the SMT solution times were manageable up to an error tolerance of 4.

MILP Modelling and Success Probability The success rate of the Mixed Integer Linear Programming (MILP) modelling was specifically noteworthy at an error tolerance of 3. This setting showed a very high success rate in correcting predicted HW classes that fell outside the expected tolerance limits, thus maintaining system consistency [16].

7.3 Overall Performance

For error tolerance 3, with 150 rounds, the SMT solver achieved a success probability of 0.968 within 49981.87 seconds and it's shown in tables 7.3 and 7.4. Increasing the number of rounds to 170 and 180, as observed reduced solution times and slightly lower success probabilities. The best result was observed at 170 rounds with a solution time of 28763.22 seconds and a success probability of 0.946 [16].

Leakage Model	Tolerance	# Rounds	Trials	Mean (sec.)	S.D. (sec.)
HW/8	1	110	20	1.16	0.05
	2	110	20	1.37	0.07
	3	110	20	1.58	0.13
HW/16	1	110	20	3.91	0.94
	2	110	20	7.38	2.18
	3	110	20	15.41	6.35
	4	110	20	91.40	187.50
		130	20	40.01	22.85
		150	20	39.89	18.62
HW/32	1	110	20	239.74	192.64
	2	110	20	7975.88	9277.67
		130	20	4764.87	4489.14
	3	130	6	122582.24	65397.23
		150	6	49975.49	31924.09
		180	5	36288.82	27153.63
	4	130	1	475778.30	–
		150	3	494445.14	38190.05
		170	2	226005.79	71432.18

Tab. 7.3 Solution times under different HW models without key-stream information [16]

Leakage Model	Tolerance	Rounds	Trials	Mean (sec.)	S.D. (sec.)
HW/32	0	110	20	5.42	1.25
	1	70	20	1309.19	1144.36
		90	20	821.76	1444.34
		110	20	254.36	195.43
	2	70	1	3408.72	–
		90	8	17404.32	27533.08
		110	16	10628.26	13962.97
		130	20	1819.21	1558.38
	3	110	1	140523.60	–
		130	3	44911.09	31619.74
		170	1	28755.36	–
	4	130	1	76797.41	–
		170	1	12582.60	–

Tab. 7.4 Results for TRIVIUM under HW/32 model in the Pseudo-random Phase [16]

7.4 Reduction of Unknown Variables

During the initialization phase of TRIVIUM, the number of unknown variables significantly reduces from 288 to 80. This reduction occurs because only the secret key bits, totaling 80 bits, remain unknown. This contraction in variable space inherently simplifies the complexity of the problem the SMT solver needs to address [16].

Error Tolerance The reduction in unknown variables allows the SMT solver to operate with higher error tolerances. This capability is critical for improving the robustness of the cipher against side-channel attacks, as it allows the system to maintain accuracy even when the input data (side-channel leakage) includes a higher degree of noise [16].

Handling Multiple Initialization Vectors (IVs) While the results reported in studies are based on scenarios involving a single Initialization Vector (IV), it is important to note that SMT model is versatile enough to handle multiple IVs effectively. Each IV introduces new information to the system, potentially reducing the solution time for the SMT solver [16].

Operational Efficiency With the introduction of each new IV, and due to the reduced number of variables during the initialization phase, the SMT solver can process instances faster and with higher error tolerance. This efficiency makes the system particularly adept at handling real-world operational scenarios where rapid key recovery is essential and result are provided in table 7.5 [16].

For the initial trials, set the number of rounds at 140, based on previous results in the key-stream generation phase indicating feasible SMT solution times. Adjustments to the number of rounds were made depending on the incidence of multiple solutions. Table 7.5 presents the solution times for HW/8, HW/16, and HW/32 during the initialization phase of TRIVIUM. Highlighted how adjustments in the number of rounds and error tolerances impact the solver's performance [16].

For HW/32, with an error tolerance of 15, the SMT solver efficiently resolves instances in 79.49 seconds at 170 rounds, with a success probability of 1. This efficiency stems from the ability to predict HW class with 100% accuracy at an error tolerance of 7, allowing for SMT instance creation without needing MILP corrections [16].

For lower tolerances, as demonstrated, the accuracy of HW prediction allows for reliable

Leakage Model	Tolerance	# Rounds	Trials	Mean (sec.)	S.D. (sec.)
HW/8	4	150	20	1.63	0.28
HW/16	5	140	20	2.12	0.23
	6	140	20	2.46	0.698
	7	140	20	4.45	2.13
	8	200	20	153.01	223.71
HW/32	3	140	20	4.93	1.55
	4	140	20	8.58	6.90
	5	140	20	10.39	11.39
		160	20	11.64	6.44
	6	140	20	27.07	34.97
	7	170	20	79.49	121.89
	8	160	20	211.89	636.58
	9	160	20	162.08	186.37
	10	160	20	585.56	1484.28
	11	160	20	1018.86	1629.53
	12	160	20	4560.34	5749.65
	13	160	11	3646.35	3632.99
	14	200	2	18379.19	5143.47
		300	2	5566.90	4175.17
		280	3	1460.03	1234.7
	15	280	1	5859.60	-

Tab. 7.5 Results for TRIVIUM in the Initialisation Phase [16]

SMT solutions at shorter times. For instance, with an error tolerance of 4, nearly all HW data (approximately 1260 blocks) fall within the error tolerance, leading to solutions within 8.58 seconds. If inconsistencies arise, adjustments in the tolerance and the number of rounds are made to optimize the solution time, following the recovery procedures outlined [16].

The initialization phase of TRIVIUM shows significant potential for rapid and accurate state recovery using the SMT solver, particularly when leveraging insights from error tolerance and rounds adjustments. These findings are critical for enhancing the cipher's resilience and effectiveness in cryptographic applications [16].

Error Tolerance and SMT Solving Error tolerance is introduced to improve the ML model's predictive accuracy, thereby assisting the Satisfiability Modulo Theory (SMT) solver in recovering the cipher's state/key more effectively. However, this increases the solution time, necessitating a balance to optimize both accuracy and computational efficiency. For the TRIVIUM cipher, optimal results are obtained with an error tolerance of three, achieving 99.7% accuracy and manageable SMT solution times [16].

Enhancements and Noise Handling The framework also incorporates Gaussian noise handling to test robustness under various Signal-to-Noise Ratios (SNRs). Lower SNRs decrease ML accuracy, impacting the overall success probability of the key recovery process. Methods to mitigate this include optimizing the ML architecture and implementing pre-processing techniques [16].

7.5 Pre-trained model and datasets

Project offers a collection of pre-trained models [24], serving as a launchpad for researchers investigating this specific attack scenario. These models are likely trained on datasets containing side-channel information (potentially power consumption traces) captured during Trivium encryption with a three-step attack strategy.

Project likely incorporates datasets specifically curated for the three-step attack against Trivium. These datasets are crucial for training and evaluating the effectiveness of the pre-trained models and any custom models researchers might develop.

The characteristics of the datasets, such as their size, content, and formatting, are readily available from the provided Bitbucket repository. Could be approximated to the sections:

- **Side-channel traces:** Recordings of power consumption or other leakages captured during Trivium encryption with the three-step attack.
- **Corresponding secret information:** The key or plaintext bytes targeted during the attack, which the models learn to associate with specific patterns in the side-channel traces.

Total number of traces: $2362000 \approx 2^{21.17}$[24]

The table 7.6 shows the distribution of samples according to their Hamming Weight (HW):

Hamming Weight (HW)	**Number of samples**
0	14000
1	14000
2	14002
3	14005
4	14012
5	14098
6	14377
7	15514
8	18505
9	26461
10	42142
11	70991
12	113913
13	168198
14	222760
15	264618
16	280021
17	263850
18	221995
19	167451
20	114025
21	71292
22	42623
23	26475
24	18645
25	15532
26	14381
27	14102
28	14010
29	14002
30	14000
31	14000
32	14000

Tab. 7.6 Distribution of samples according to their Hamming Weight [24]

CONCLUSION

This study presents a significant step forward in the field of cryptographic research. It introduces a method that systematically combines various analytical techniques, offering a novel tool for both researchers and practitioners in cybersecurity, enhancing collective ability to secure digital communications against emerging threats.

Research relies on the fact that datasets for both the TRIVIUM cipher and the SCAAML framework were available in repositories, which provided valuable resources for this research. The two key areas have been explored: the three-step attack on TRIVIUM and SCAAML.

The first step of this research involved getting a solid understanding of modern trends in SCA. Thesis reveal the history and possibilities of various techniques, which helped me build a strong foundation for comprehending its strengths and weaknesses. With this theoretical knowledge focused on describing various attacks specifically designed to target microcontrollers. There were highlighted the vulnerabilities these devices possess and the potential consequences regarding the SCA.

The implementation phase of research involved a comparison of the two chosen approaches. SCAAML emerged as a user-friendly framework with a strong foundation built by Google and Elie Bursztein. Its focus on practicality and its extensive capabilities, including connections to ChipWhisperer and a variety of optimization methods, make it a very attractive option for SCA practitioners. Project is focused on software library TinyAES (lightweight library with software AES implementation).

This thesis navigated the complexities of SCA research by combining a theoretical foundation with practical implementation using pre-trained models. In fact, the pretrained models had close to the same performance on datasets included with the framework itself. This approach contributes to the understanding and potential applications of ML-powered SCA. As moving forward, it's important to prioritize ethical considerations. While SCA techniques are valuable for research, they can pose security risks if misused. Ensuring that this research adheres to responsible practices and focusing on theoretical understanding and educational purposes are critical aspects to consider.

LIMITATIONS AND FUTURE WORKS

The challenges encountered during the initial stages of this research reflect a broader issue in the field of SCA research. This initial plan was to create this own tools, but this quickly highlighted the difficulty of securing the proper resources. From the limited availability of oscilloscopes with proper computer connectivity to the scarcity of powerful GPUs in today's AI boom, these limitations can significantly impact research progress. Have been even tried leveraging Google Cloud and deploying machines with advanced GPUs like V100s and K80s, but unfortunately, the current demand for AI resources made it very difficult to get access. Despite communicating with support and sales teams, the fact that this was for research purposes wasn't enough. This experience underscores the need for more readily available resources to support academic exploration in this crucial field.

While SCAAML offers a practical and well-established approach, this exploration of the TRIVIUM attack also revealed its unique potential for further research. This framework presents significant research opportunities, especially for those interested in delving deeper into advanced SCA techniques. Further research could explore the use of different machine learning models and preprocessing techniques to enhance accuracy and reduce the dependency on MILP corrections. Other models like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), or Long Short-Term Memory (LSTM) networks could be considered for analyzing time-series data from electromagnetic traces [16]. Developing analytical models such as Hidden Markov Models (HMM) might allow the framework to operate effectively with less than perfect accuracy, potentially eliminating the need for intermediate MILP corrections [16]. Adapting the framework to accommodate polynomial and weighted leakage functions could provide a broader application range, although it may require enhancements to the current SMT model to handle these complexities [16]. The future of SCA research is full of potential. By addressing resource limitations and fostering responsible research practices, the field can flourish, ultimately contributing to the development of more robust and secure cryptographic systems. This thesis serves as a valuable stepping stone on this path, offering insights and highlighting areas for further exploration.

REFERENCES

[1] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In B. S. Kaliski, editor, *Advances in Cryptology - CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer Berlin Heidelberg, 1997.

[2] E. Biham and A. Shamir. Differential power analysis of the aes. In *Advances in Cryptology—CRYPTO 2002*, pages 387–398, 2002.

[3] E. Brown. Electromagnetic radiation from digital devices: An early study. In *Proceedings of the International Symposium on Electromagnetic Compatibility*, pages 45–52, 1978.

[4] E. Bursztein and J.-M. Picod. A hacker guide to deep learning based side channel attacks. In D. CON, editor, *DEF CON 27*, 2019.

[5] E. Bursztein, L. Invernizzi, K. Král, D. Moghimi, J.-M. Picod, and M. Zhang. Generalized power attacks against crypto hardware using long-range deep learning. *arXiv preprint arXiv:2306.07249*, 2023.

[6] E. Bursztein et al. Scaaml: Side channel attacks assisted with machine learning, 2019. URL `https://github.com/google/scaaml`.

[7] C. D. Cannière and B. Preneel. Trivium. In M. J. B. Robshaw and O. Billet, editors, *New Stream Cipher Designs - The eSTREAM Finalists*, volume 4986 of *Lecture Notes in Computer Science*, pages 244–266. Springer, 2008.

[8] eSTREAM Project. Trivium Specification and Supporting Documentation, 2006. URL `https://www.ecrypt.eu.org/stream/p2ciphers/trivium/trivium_p2.pdf`. [Online; accessed 28-April-2024].

[9] Google. Scaaml introduction, 2023. URL `https://github.com/google/scaaml/tree/main/scaaml_intro`. Accessed: 2024-05-07.

[10] M. Hamann, M. Krause, and W. Meier. Lizard - a lightweight stream cipher for power-constrained devices. *IACR Trans. Symmetric Cryptol.*, 2017(1):45–79, 2017.

[11] R. Jones. *Acoustic Signals in Electronics: An Overview*. TechPress, 1982.

[12] A. R. Kazmi, M. Afzal, M. F. Amjad, H. Abbas, and X. Yang. Algebraic side channel attack on trivium and grain ciphers. *IEEE Access*, 5:23958–23968, 2017.

[13] P. C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In M. J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397, Santa Barbara, California, USA, 1999. Springer.

[14] P. C. Kocher, J. Jaffe, and B. Jun. *Cryptographic systems and their side-channel attacks*. Springer, 1999.

[15] P. C. Kocher, J. Jaffe, and B. Jun. Differential cryptanalysis of des-like cryptosystems. In *Advances in Cryptology—CRYPTO 2003*, pages 19–30, 2003.

[16] S. Kumar, V. A. Dasu, A. Baksi, S. Sarkar, D. Jap, J. Breier, and S. Bhasin. Side channel attack on stream ciphers: A three-step approach to state/key recovery. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2022(2): 166–191, Feb. 2022. doi: 10.46586/tches.v2022.i2.166-191. URL `https://tches.iacr.org/index.php/TCHES/article/view/9485`.

[17] I. Loshchilov and F. Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017. URL `http://arxiv.org/abs/1711.05101`.

[18] T. Messerges, E. Dabbish, and R. Sloan. Examining smart-card security under the threat of power analysis attacks. *IEEE Transaction on Computers*, 51(5):541–552, 01 2002.

[19] N. I. of Standards and Technology. Privacy-enhancing cryptography: A complement to differential privacy, 2023. URL `https://www.nist.gov/blogs/cybersecurity-insights/privacy-enhancing-cryptography-complement-differential-privacy`. Accessed: 2024-05-07.

[20] Y. Oren, M. Renauld, F.-X. Standaert, and A. Wool. Algebraic side-channel attacks beyond the hamming weight leakage model. In E. Prouff and P. Schaumont, editors, *Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop*, volume 7428 of *Lecture Notes in Computer Science*, pages 140–154, Leuven, Belgium, 2012. Springer.

[21] E. Oswald and S. Mangard. Power analysis attacks: Revealing the secrets of smart cards. *IEEE Computer*, 38:62–68, 2005.

[22] M. S. Pedro, V. Servant, and C. Guillemet. Side-channel assessment of open source hardware wallets. Cryptology ePrint Archive, Paper 2019/401, 2019. URL `https://eprint.iacr.org/2019/401`. `https://eprint.iacr.org/2019/401`.

[23] E. Peeters. *Advanced DPA Theory and Practice: Towards the Security Limits of Secure Embedded Circuits*. Springer-Verlag New York, 1 edition, 2013.

[24] A. Sen. Trivium 3-step sca: Source code and analysis, 2023. URL `https://bitbucket.org/anubhab001/trivium-3step-sca/src/master/`. Accessed: May 7, 2024.

[25] A. Shamir. Side-channel cryptanalysis of product ciphers. *Cryptographers' Track at the RSA Conference*, 3152:97–110, 2003.

[26] J. Smith. Early exploration of unintended signals in electronic devices. *Electronics History Journal*, 5:23–36, 1965.

[27] F.-X. Standaert. *Introduction to Side-Channel Attacks*, pages 27–42. 12 2010. ISBN 978-0-387-71827-9. doi: 10.1007/978-0-387-71829-3_2.

[28] T. Trouchkine, S. K. Bukasa, M. Escouteloup, R. Lashermes, and G. Bouffard. Electromagnetic fault injection against a system-on-chip, toward new micro-architectural fault models. *CoRR*, abs/1910.11566, 2019. URL `http://arxiv.org/abs/1910.11566`.

[29] Wikipedia contributors. The thing (listening device) — wikipedia, the free encyclopedia, 2024. URL `https://en.wikipedia.org/wiki/The_Thing_(listening_device)`. [Online; accessed 28-April-2024].

[30] Yan. Side channel attacks part 2: Dpa and cpa applied on aes attack. `https://yan1x0s.medium.com/side-channel-attacks-part-2-dpa-cpa-applied-on-aes-attack-66baa356f03f`, 2021. Accessed: 2024-05-09.

LIST OF ABBREVIATIONS

AES	Advanced Encryption Standard
CNN	Convolutional Neural Networks
CPA	Correlation Power Analysis
DES	Data Encryption Standard
DPA	Differential Power Analysis
EDAC	Error Detection and Correction
EMI	Electro-Magnetic Interference
EM	Electromagnetic
FA	Fault Analysis
GF	Galois Field
HD	Hamming Distance
HMM	Hidden Markov Model
HSM	Hardware Secure Module
HW	Hamming Weight
IV	Initial Vector
KB	Kilo Byte
LFSR	Linear-feedback shift register
LSTM	Long Short-Term Memory
MILP	Mixed Integer Linear Programming
MLP	Multi-Layer Perceptron
ML	Machine Learning
NSA	National Security Agency
RNN	Recurrent Neural Networks
SCA	Side-Channel Analysis
SCAAML	Side Channel Attacks Assisted with Machine Learning
SNR	Signal to Noise Ratio
SMT	Satisfiability Modulo Theory
SRAM	Static Random Access Memory

LIST OF FIGURES

LIST OF TABLES