Detect Network API-related Call Modifications of Embedded Systems through EM-SCA.

A Thesis Submitted for the Degree of Master of Computer Science

W.D Rasika
University of Colombo School of Computing
2025

DECLARATION

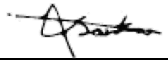
Name of the student: W.D Rasika
Registration number: 2020/MCS/072
Name of the Degree Programme: MCS in Computer Science
Project/Thesis title: Detect Network API-related Call Modifications of Embedded Systems through EM-SCA

1. The project/thesis is my original work and has not been submitted previously for a degree at this or any other University/Institute. To the best of my knowledge, it does not contain any material published or written by another person, except as acknowledged in the text.
2. I understand what plagiarism is, the various types of plagiarism, how to avoid it, what my resources are, who can help me if I am unsure about a research or plagiarism issue, as well as what the consequences are at University of Colombo School of Computing (UCSC) for plagiarism.
3. I understand that ignorance is not an excuse for plagiarism and that I am responsible for clarifying, asking questions and utilizing all available resources in order to educate myself and prevent myself from plagiarizing.
4. I am also aware of the dangers of using online plagiarism checkers and sites that offer essays for sale. I understand that if I use these resources, I am solely responsible for the consequences of my actions.
5. I assure that any work I submit with my name on it will reflect my own ideas and effort. I will properly cite all material that is not my own.
6. I understand that there is no acceptable excuse for committing plagiarism and that doing so is a violation of the Student Code of Conduct.

Signature of the Student	**Date (DD/MM/YYYY)**
	28/05/2025

Certified by Supervisor(s)

This is to certify that this project/thesis is based on the work of the above-mentioned student under my/our supervision. The thesis has been prepared according to the format stipulated and is of an acceptable standard.

	Supervisor 1	**Supervisor 2**	**Supervisor 3**
Name	Dr. A.P Sayakkara		
Signature			
Date	2025-05-28		

I would like to dedicate this thesis to

my beloved wife,

for their love, support, and encouragement.

&

academic and non-academic staff of

University of Colombo School of Computing

for their support and guidance given

to make this thesis a success.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank the project supervisor, Dr. A.P Sayakkara, for his unrestricted assistance and direction in helping me complete this thesis. I also want to express my gratitude to the MCS3204 module coordinators for their constant advice during the project's duration. I would especially want to thank the University of Colombo School of Computing for allowing me to conduct this research study.

Finally, I would want to express my gratitude to everyone who wishes to remain anonymous. The assistance they gave me was invaluable and greatly appreciated.

ABSTRACT

In the rapidly evolving landscape of Industry 4.0, embedded devices play a pivotal role in connecting cyber-physical systems and enabling real-time data exchange. However, this interconnectedness also exposes IoT ecosystems to significant threats, including malicious code injections and unauthorized network API calls that can exfiltrate sensitive information. This thesis investigates the feasibility of employing EM-SCA as a non-invasive method to detect and quantify such firmware tampering in embedded systems. By capturing EM signals from a NodeMCU device during both legitimate (manufacturer-authenticated) and unauthenticated API executions, distinct emission patterns are identified.

A NodeMCU microcontroller to simulate both authenticated and unauthenticated network API exchanges within a controlled environment. EM emissions were monitored using an H-loop probe and HackRF One SDR device, targeting prominent frequency leakage bands. Statistical analysis, employing t-tests at a 95% confidence interval, identified significant EM signature deviations between authenticated and compromised firmware conditions. SVM classifiers achieved over 81% accuracy in distinguishing these states, while a refined analysis focusing solely on request payload modifications attained 79% accuracy. A Neural Network model further improved detection performance, reaching 82% accuracy for identifying subtle data leakage variations. These findings demonstrate the efficacy of EM-based side-channel monitoring as a robust, non-invasive approach for real-time malicious code detection in constrained IoT environments.

TABLE OF CONTENTS

Declaration i

ACKNOWLEDGEMENTS iii

ABSTRACT iv

Table of Contents v

LIST OF FIGURES ix

LIST OF TABLES xii

ABBREVATIONS xiii

Chapter 1 - INTRODUCTION 1

1.1 Introduction 1

1.2 Motivation 4

1.3 Statement of the problem 4

1.4 Research Aim and Objectives 6

1.4.1 Aim 6

1.4.2 Objectives 6

1.5 Scope 7

1.6 Contribution to Computer Science Domain 8

1.7 Research Challenges 8

1.8 Chapter Summary and Structure of the Thesis 9

Chapter 2 - Literature Review 10

2.1 Overview 10

2.2 Side Channel Analysis (SCA) 10

2.3 Electromagnetic Side Channel Analysis (EM-SCA) 11

2.3.1 Equipment for capturing EM Radiation 13

2.3.2 Related Work 13
2.4 Internet of Things (IoT) 15
2.4.1 Architecture of IoT 17
2.4.2 Security in IoT Devices 18
2.4.3 IoT Platform 20
2.5 Digital Forensics 21
2.6 Signal Processing 22
2.6.1 Digital Signal Processing 23
2.6.2 Fast Fourier Transform (FFT) 23
2.7 Software-defined Radio (SDR) 24
2.7.1 HackRF One 25
2.8 Chapter Summary 26
Chapter 3 - Methodology 28
3.1 Research Methodology 28
3.2 Overview 29
3.3 Procedure for Data Acquisition 29
3.3.1 Device Under Test (DUT) 29
3.3.2 Platform / Software Specification 30
3.3.3 Determining Data Acquisition Parameters 31
3.3.4 Type of APIs for Data Collection 32
3.3.5 Modification for Authenticated Code Base 35
3.3.6 Data Collection 39
3.4 Data Preprocessing 40
3.4.1 Normalization 40
3.5 Statistical Testing 41
3.5.1 T-test 41

3.5.2 Analysis of Variance (ANOVA) 42
3.6 Build the Machine Learning Model 42
3.6.1 Support Vector Machines (SVM) 43
3.6.2 Neural Network 44
3.7 Chapter Summary 46
Chapter 4 - Implementation 48
4.1 Hardware Setup 48
4.2 Software Setup 49
4.3 Explore the Leakage Frequency 50
4.4 Short-time Fourier transform (STFT) 53
4.5 Hyper Parameter Tuning 53
4.6 Chapter Summary 55
Chapter 5 - EVALUATION AND RESULTS 56
5.1 Overview 56
5.2 Summary of Dataset 56
5.2.1 Distribution of the Dataset 56
5.3 Statistical Analysis 59
5.3.1 T-Test Results 59
5.3.2 ANOVA Test Results 60
5.4 Support Vector Machine Model 62
5.5 Neural Network Model (NN model) 65
5.6 Chapter Summary 67
Chapter 6 - CONCLUSION AND FUTURE WORK 69
6.1 Conclusion 69
6.2 Limitation 70
6.3 Future Works 70

APPENDICES 72

Appendix A 72

Appendix B 76

REFERENCES 77

LIST OF FIGURES

Figure 1: The 4 Industrial Revolutions Adapted from (ISA, 2023). 1
Figure 2: Flow chart for authenticated code base setup in the NodeMCU MCU 7
Figure 3: (A) An image showing the placement of an EM probe on an FPGA; (B) Setup for EM Side-channel investigation. Adopted from (Bhunia and Tehranipoor, 2018). 12
Figure 4:Near field probes for oscilloscopes 13
Figure 5: Internet of Things (IoT) elements 17
Figure 6: IoT Components and Architecture 18
Figure 7: LoLin style NodeMCU 21
Figure 8: Common Types of Digital Forensics 22
Figure 9: Every signal can be depicted in a three-dimensional space involving time, amplitude, and frequency. Complex waveforms can be constructed by combining multiple sine waves with varying amplitudes, phases, and frequencies, as illustrated. 23
Figure 10: Power spectral density plot derived (Mcnames et al., 2002) 24
Figure 11: Spectrogram plot derived from (Mcnames et al., 2002) 24
Figure 12: High-level summary of the different parts that make up an SDR systems transmit (TX) and receive (RX) chains 25
Figure 13: HackRF One device with its default antennas 25
Figure 14: Overview of the research process 29
Figure 15: LoLin style NodeMCU measures 58mm x 32mm with a pin spacing of 0.1″ between pins and 1.1″ between rows derived from https://pino-tech.eu/product/nodemcuv3/ 30
Figure 16: Postman Request and Response of Random Number Generation API 32
Figure 17: Postman Request and Response for Subscriber API 33
Figure 18: Postman Request and Response for Weather API 33
Figure 19: Postman Request and Response for Notification API 34
Figure 20: Postman Request and Response for PING API 34
Figure 21: Sequence diagram for API activities in the in the Authenticated firmware 35
Figure 22: The Sequence diagram for remove RGN API 36
Figure 23: Sequence diagram for invoke PING API for difference location in the NodeMCU code 37
Figure 24: Sequence diagram for adding DG API 38
Figure 25: EM Trace binary data store represent in hierarchically 39

Figure 26: Proposed experimental setup for the EM data acquisition 48
Figure 27: The arrangement of the H-Loop antenna on top coner of the NodeMCU 48
Figure 28: Setup of the GNU Radio Companion component to record and save EM data using HackRF 49
Figure 29: H-loop antenna set to chip's center position would result 364MHz 50
Figure 30: Output generated by the GNU Radio Companion EM radiation of NodeMCU at the leakage frequency of 364MHz. Center frequency is 360MHz when antenna is on center position without Node MCU is running 51
Figure 31: Output generated by the GNU Radio Companion EM radiation of NodeMCU at the leakage frequency of 364MHz. Center frequency is 360MHz when antenna is on center position with NodeMCU is running. 51
Figure 32: Output generated by the GNU Radio Companion EM radiation of NodeMCU at the leakage frequency of 390MHz. Center frequency is 387MHz when antenna is on corner position without Node MCU is running 52
Figure 33: Output generated by the GNU Radio Companion EM radiation of NodeMCU at the leakage frequency of 390MHz. Center frequency is 387MHz when antenna is on corner position with NodeMCU is running. 52
Figure 34: Python code snippet for STFT to the captured EM signal with 20MHz sample rate, 2048 points FFT window 256 sample overlap 53
Figure 35: Mean cv score for combination of hidden layers and neurons 54
Figure 36: EM Trace in Time domain for first 1second representation 57
Figure 37: Authenticated APIs execution in frequency domain 58
Figure 38: Remove Random generated number API execution in frequency domain 58
Figure 39: Ping First API execution in frequency domain 58
Figure 40: Ping middle API execution in frequency domain 58
Figure 41: Ping Last API execution in frequency domain 58
Figure 42: Info Gather API execution in frequency domain 58
Figure 43: Change the payload of the weather API execution in frequency domain 59
Figure 44: Bar chart illustrating −log(p) values for each unauthenticated API test, highlighting statistically significant differences from the authenticated baseline. 60
Figure 45: Box plot representing Average Magnitude of STFT for each PING API invoke position 61

Figure 46: Mean accuracy of 0.8959 for authenticated API vs remove RGN API 63
Figure 47: Mean accuracy of 0.8802 for authenticated API vs PING first API..... 63
Figure 48: Mean accuracy of 0.9719 for authenticated API vs PING middle API 63
Figure 49: Mean accuracy of 0.9719 for authenticated API vs PING last 63
Figure 50: Mean accuracy of 0.8496 for authenticated API vs data gather API..... 63
Figure 51: Mean accuracy of 0.9730 for authenticated API vs update payload of weather API .. 63
Figure 52: Confusion matrix of the SVM classification..... 65
Figure 53: Cross validation for data leakage classification..... 65
Figure 54:Training and Validation Loss Curves for implemented model..... 66
Figure 55: The confusion matrix of classifying information leakage using neural network model 67
Figure 56: Properties of a periodic signal from (The Open University, n.d.) 73
Figure 57: this illustration, represents the initial analog signal (depicted in green), the quantized signal represented by black dots, the signal reconstructed from the quantized data (shown in yellow), and the variance between the original signal and the reconstructed one (highlighted in red). This discrepancy between the original and reconstructed signals constitutes the quantization error. Adopted from ("Wikipedia," n.d.)..... 74
Figure 58: Cartesian coordinate representation of an SDR data sample. Adopted from ("I/Q Data for Dummies," n.d.)..... 75

LIST OF TABLES

Table 1: Hyper Parameter for NN model 53
Table 2: P-value table of unauthenticated APIs vs Authenticate API for t-test experiment 59
Table 3: SVM result of authorized and unauthorized API for binary classification 62
Table 4: SVM classification results for Data leakage 64
Table 5: NN model classification for information leakage statistics..... 66

ABBREVATIONS

4IR - Fourth (4th) Industrial Revolution

IoT - Internet of Things

CPS - Cyber-Physical Systems

DoS - Denial of Service

MitM - Man-in-the-Middle

EM – Electromagnetic

SCA - Side-Channel Analysis

EM-SCA - Electromagnetic Side-Channel Analysis

API - Application Programming Interface

MSISDN - Mobile Station International Subscriber Directory Numbers

CRT - Cathode-ray Tube

DPA - Differential Power Analysis

DES - Data Encryption Standard

CPA - Correlation Power Analysis

ML – Machine Learning

PLC - Programmable Logic Controller

CPU – Central Processing Unit

RF – Random Forests

DL – Deep Learning

CNN - Convolutional Neural Network

ANOVA - Analysis of Variance

DUT – Device Under Test

EMI – Electromagnetic Interference

ADC – Analog to Digital Conversion

SDR – Software Define Radio

CHAPTER 1 - INTRODUCTION

1.1 Introduction

The term "Industrie 4.0" (I4.0) was first introduced in 2011 by Henning Kagermann, the head of the German National Academy of Science and Engineering (Acatech) (Kagermann et al., 2011). And it was officially unveiled at the World Economic Forum's yearly meeting in 2016 (“World Economic Forum,” n.d.). Since its introduction, the concept has gained widespread recognition and popularity (Kagermann et al., 2018). This revolution signifies a significant transition towards a digital era, where the fabric of our world is increasingly woven with digital data at its core. This transformative shift is reshaping the way we conduct business, industry, and daily life, with digital data serving as the linchpin for these changes (Hermann et al., 2016; ONIK et al., 2019).

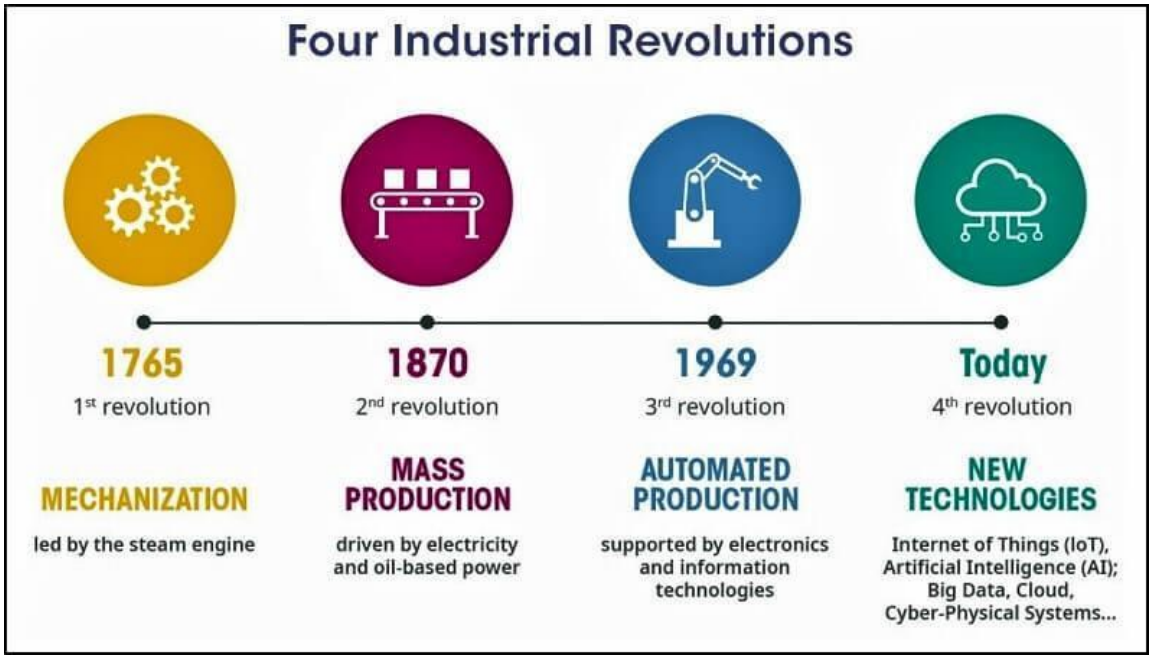


Figure 1: The 4 Industrial Revolutions Adapted from (ISA, 2023).

4IR primarily seeks to integrate advanced technologies into production and manufacturing processes, aiming to enhance automation while minimizing human intervention (Joshi et al., 2022). Hermann et al. identified four foundational pillars of Industry 4.0: **Interconnection**, which facilitates seamless communication between people, machines, and sensors to form a networked ecosystem; **Information Transparency**, which focuses on increasing the connectivity of objects and individuals to improve data sharing and visibility; **Decentralized Decisions**, where embedded

systems and sensors autonomously manage the physical world, enhancing operational efficiency; and **Technical Assistance**, which provides decision-making support to improve the speed and quality of decisions (Hermann et al., 2016).The core principles of 4IR are centered on driving a new era of enhanced productivity and operational efficiency. A key enabler of 4IR is the integration of the Internet of Things (IoT) into manufacturing processes. In this framework, "Things" encompass devices such as sensors, actuators, and mobile phones, all working collaboratively within Cyber-Physical Systems (CPS) to achieve common objectives in Smart Factories. The primary components of 4IR include IoT, CPS, and Smart Factories. CPS serve as the bridge between the physical and virtual realms, utilizing embedded devices to perform computations, make decisions, and regulate physical processes through feedback loops. By merging IoT with CPS, Smart Factories play a pivotal role in the realization of 4IR (Hermann et al., 2016).

Verma et al. pinpointed seven potential threats and vulnerabilities that could impact the security of 4IR (Verma and Bharot, 2023). They are:

- **Cyber Attacks:** As connectivity and integration increase within Industry 4.0, so does the vulnerability to cyberattacks. Malicious actors, including cybercriminals, exploit vulnerabilities in IoT devices, network configurations, and software systems to gain unauthorized access. Their objectives may include stealing sensitive data, disrupting critical operations, or extorting organizations through ransom demands. Common attack methods include Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks, Man-in-the-Middle (MitM) attacks, and Ransomware attacks (Verma and Bharot, 2023).
- **Data Breaches:** In the context of 4IR the extensive generation and circulation of data present significant challenges. Malicious actors, such as cybercriminals, may attempt to exploit this data to access sensitive information, including trade secrets, customer data, and financial records. Such breaches can result in substantial financial losses and damage to a company's reputation (Verma and Bharot, 2023)
- **Insider Threats:** In the context of 4IR systems, internal personnel can pose significant security risks. This includes employees with authorized access to critical data and systems, who may intentionally or inadvertently compromise system integrity or functionality (Verma and Bharot, 2023).

- **Physical Threats**: 4IR is also susceptible to various physical threats, including theft, vandalism, and natural disasters, which can compromise the infrastructure and devices integral to its operations. Damage or disruption caused by these incidents can lead to significant operational challenges and downtime (Verma and Bharot, 2023).
- **Supply Chain Vulnerabilities:** Due to the interconnected nature of 4IR systems, they are particularly vulnerable to supply chain attacks. Cybercriminals may target companies that provide components or manufacture products utilized within 4IR ecosystems. Such attacks can disrupt the supply chain, leading to issues such as malware infections, data breaches, and other security vulnerabilities (Verma and Bharot, 2023).
- **Lack of Standards and Regulations:** 4IR faces significant challenges due to the lack of established rules and standardized protocols, which can compromise security. The absence of universally accepted security measures and best practices creates difficulties for manufacturing companies in ensuring the reliability and security of their systems (Verma and Bharot, 2023).
- **Human Error:** Despite the advanced technologies underpinning 4IR human errors remain a significant source of security vulnerabilities. Employees may inadvertently compromise systems through misconfigurations, poor password management, or by falling victim to phishing attacks (Verma and Bharot, 2023).

A major security issues in 4IR is data privacy. In this environment, individuals willingly share their personal information with IoT devices, viewing this data as valuable. However, due to security weaknesses in IoT devices, this sensitive information is at risk of being exposed. IoT devices typically have constrained resources, such as low-cost components, basic processors, and limited random-access memory (RAM). While these devices are capable of connecting to the internet, they often face challenges in maintaining security. Implementing additional security measures can be expensive and may negatively impact their performance(ONIK et al., 2019; Verma and Bharot, 2023). This thesis aims to address the risks associated with unauthorized network Application Programming Interface (API) calls, which can lead to significant data leakage, and seeks to develop methods for detecting and quantifying these threats to enhance IoT security.

1.2 Motivation

Hardware vulnerabilities are a major source of security threats, as attackers can exploit the physical components that underpin secure and encrypted software systems. One of the most critical threats in this domain is side-channel analysis (SCA), which poses significant risks to hardware security (Amrouche et al., 2022). Among the various types of side-channel attacks, Electromagnetic Side-Channel Analysis (EM-SCA) stands out as a particularly impactful method. EM-SCA has been used to extract sensitive information, such as cryptographic keys, from both traditional computing systems and IoT devices (Sayakkara and Le-Khac, 2021). This demonstrates the efficacy of SCA in circumventing conventional security mechanisms (Poussier et al., 2016).

In the context of cost-effective IoT edge devices, the NodeMCU microcontroller offers a practical solution due to its ability to interface with multiple sensors while remaining affordable and efficient. It is commonly employed for tasks such as home automation, temperature monitoring, and enhancing location-based security (Rathour et al., 2023). However, such embedded devices are susceptible to attacks, including crypto-mining or unauthorized activities, when compromised by malware designed to manipulate their hardware and functionality (Mukhtar and Kong, 2018). These attacks often involve a high volume of unauthorized API-related network calls initiated with malicious actors. Consequently, there is a growing need for non-invasive methods to detect and monitor these network API-related activities.

1.3 Statement of the problem

Embedded systems have broadened the scope of digital forensic inquiries by offering a diverse array of new sources of evidence. These systems encompass a range of devices, including health implants, wearable sports technology, intelligent home security systems, programmable thermostats, and sophisticated electrical appliances. The wide range and intricate nature of embedded environments present a significant challenge to digital investigators about ensuring its security. There is no assurance that an embedded device is using the original implementation provided by the manufacturer. If the device's source code has been altered or infected by malware, any digital evidence gathered from the linked smartphone app or cloud servers may become untrustworthy. For example, according to a Gartner report, 20% of organizations have encountered at least one IoT-related attack within the last three years (Gartner, n.d.). This statistic highlights the

growing vulnerability of IoT devices in both enterprise and consumer environments. These devices are not only targeted for data breaches but also face the risk of being co-opted into botnets, such as the infamous Mirai botnet. The Mirai botnet leveraged thousands of compromised IoT devices like cameras and routers to launch large-scale DDoS attacks, disrupting major websites and services (Alieyan et al., 2019).

Recently, researchers have proven that the SCA can be applied for defensive purposes (Adnan Khan et al., 2018; Han et al., 2017) .A key application of this approach is utilizing SCA for remote anomaly detection, which enables external monitoring without adding processing overhead to the target device. Of the various modalities available for such analysis, such as power consumption, acoustics, and thermal emissions electromagnetic(EM) signals are gaining traction. This preference is due to the EM spectrum's ability to support higher bandwidth and faster sampling rates, making it particularly effective for identifying irregularities in device behavior (Sehatbakhsh et al., 2018). Therefore, EM-SCA has emerged as a promising method for extracting valuable forensic data from embedded devices over other SCA methods.

In the context of an IoT ecosystem, the volume of data being transmitted through cloud networks is immense, as IoT devices worldwide continuously communicate with cloud-based services. While manufacturers' authenticated APIs are essential for legitimate data exchange, the system is also vulnerable to unauthorized or malicious API calls initiated by attackers aiming to gather sensitive information. Detecting and identifying these malicious API calls is critical, as they may result in significant data leakage. Quantifying the extent of this leakage is equally important for developing effective countermeasures. Proactively addressing these security vulnerabilities can mitigate the risk of unauthorized data extraction and ensure the integrity of IoT systems. This research mainly focuses on testing following hypothesis.

Hypothesis 1:

- H_0: EM probes cannot detect firmware tampering in embedded systems by monitoring network API-related activity.

Hypothesis 2:

- H_0: Cannot quantify the extent of information leakage (character-wise) resulting from network API-related modifications using EM probes.

1.4 Research Aim and Objectives

1.4.1 Aim

- To explore the feasibility of using EM-SCA for detecting firmware tampering in embedded systems by evaluating whether variations in electromagnetic emissions can reliably indicate anomalies caused by unauthorized changes in the firmware. This aim seeks to enhance the trustworthiness of forensic analysis in embedded systems by leveraging EM-SCA as a non-invasive detection method.
- To investigate the application of EM-SCA for monitoring network API-related activities in IoT ecosystems, focusing on its ability to distinguish between legitimate and malicious API calls in real-time without interfering with device performance. This aim examines how EM signals can be utilized as a proactive security measure to detect and prevent unauthorized access in IoT environments.
- To quantify and assess the extent of information leakage resulting from unauthorized network API modifications in IoT devices by analyzing EM side-channel emissions and categorizing leakage levels. This aim aims to provide critical insights into the risks posed by malicious API activities, supporting the development of robust security protocols to safeguard sensitive data in IoT ecosystems.

1.4.2 Objectives

- Investigating if EM probes can detect tampering in embedded systems by monitoring network API-related activities, particularly changes resulting from malicious/ unauthenticated API calls.
- Measuring the extent of information leakage caused by network API-related modifications, with the goal of understanding how unauthorized or malicious interactions affect the integrity of the IoT ecosystem and embedded systems using deep neural network
- Execution order of the APIs will affect the EM probes.

1.5 Scope

The scope of this research investigates the influence of code modifications, specifically those related to network API related calls, on the EM radiation emitted during code execution. The primary objective is to assess how API changes affect EM side-channel leakage by comparing variations in radiation patterns with those generated by a manufacturer-authenticated implementation of the code.

The experiment is conducted using an NodeMCU microcontroller, executing various permutations of API calls. EM radiation emitted during execution is captured by a HackRF device equipped with an H-loop antenna. This setup facilitates a comprehensive analysis of the EM side-channel data. The flow chart shown as Figure 2, serves as a manufacturer-authenticated implementation, acts as a reference point for benchmarking radiation patterns against those produced by the modified code.

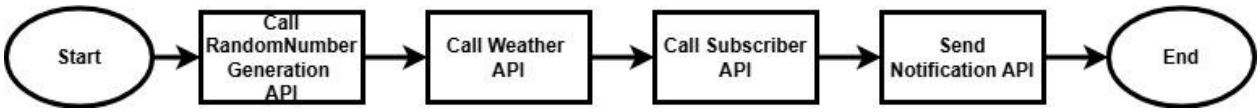


Figure 2: Flow chart for authenticated code base setup in the NodeMCU MCU

Figure 2 presents the flow chart for the base program. The **Call Random Number Generation API** function initiates a GET request to a web server, receiving a random number in the response designated as manufacturer-authenticated API call number one. The **Call Weather API** function then sends a POST request to the server with location data and the server responds with weather information, manufacturer-authenticated API call number two. Additionally, the **Call Subscriber API** function retrieves a set of Mobile Station International Subscriber Directory Numbers (MSISDN), corresponding to manufacturer-authenticated API call number three. Finally, the **Send Notification API** function sends weather-related notifications to subscribers, representing the fourth and final manufacturer-authenticated API call.

This set-up enables a rigorous examination of the identifying and quantifying information leakage between network API-related modifications and variation in EM radiation patterns.

1.6 Contribution to Computer Science Domain

In previous studies, researchers have explored deep learning approaches for detecting activities and anomalies in embedded systems (Sayakkara et al., 2020; Sehatbakhsh et al., 2018; Wanga, et al., 2018). More recent work has demonstrated that EM-SCA can be used to detect Trojan attacks or malware in these systems (Adnan Khan et al., 2018; Khan et al., 2021; Lu et al., 2020). These studies employed classification models, often using deep learning methods, to identify malicious activities.

This research takes a different aspect by focusing on the detection of network API-related activities in embedded systems, specifically through the analysis of EM traces. Unlike previous work, which primarily looked at general anomalies or malware, this study captures API-level interactions between the embedded system and external networks, such as communication with clouds, servers, and clients using various protocols, a one of core components of IoT systems. By identifying unauthenticated API calls through EM trace analysis, organizations can enhance the security and integrity of their services, ensuring that only authorized interactions occur. This contribution opens up new avenues for future researchers to explore the detection of unauthorized API activities in IoT and embedded systems using SCA techniques

1.7 Research Challenges

The major challenges faced during the study are as follows:

- Collecting a massive EM traces dataset
- Determine the most significant leakage frequencies of the NodeMCU.
- The H-loop antenna should be securely positioned in a fixed location relative to the NodeMCU processor during data collection to ensure consistent and reliable electromagnetic signal measurements

1.8 Chapter Summary and Structure of the Thesis

This chapter provides a summary of the research work. This chapter provides a comprehensive overview of the research background, scope, requirements, and objectives. This chapter briefly outlines the challenge, limitations of present systems, and other relevant factors. The thesis documents the research work with five main chapters. The literature review, which is the second chapter, provides a background analysis of the research topic by citing previously published research materials, papers, books, journals, internet articles, etc. An in-depth literature review was performed to identify the previous methodologies followed in similar research studies. The methodology which is the third chapter details the research approach and describes the technical aspects of the dataset, characteristics, design, and modeling. The fourth presents the evaluation of the results obtained through performing the research methodology. The study effort is concluded in the last chapter, which summarizes the findings as well as recommendations for future research and the project's limits.

CHAPTER 2 - LITERATURE REVIEW

The literature review chapter will summarize the related information on this study. This chapter includes an in-depth analysis of the research, along with a critical assessment of related earlier publications. Also, this chapter includes an algorithmic study, an assessment of present approaches, and the methodologies and current technologies that are in existence.

2.1 Overview

In electrical and magnetic experiments, one of the most noticeable things is how objects can affect each other's movement even when they're not touching. Scientists first try to figure out how strong and in what direction this force is between objects. They found that this force depends on the object's position and their electric or magnetic properties. So, scientists thought that there might be something in each object, either still or moving, that determines its electric or magnetic state, and this something can affect other objects from a distance following specific rules. Based on this idea, (Maxwell, 1865) developed mathematical explanations for static electricity, magnetism, how currents in wires affect each other, and how currents are created in nearby wires. These theories focus on the force between objects based on their properties and positions, without considering the surrounding environment explicitly. (see Electromagnetic Core Concepts)

2.2 Side Channel Analysis (SCA)

Based on that, the initial research on EM data leakage was first documented in 1985 (Wim van, 1985) by illustrating that cathode-ray tube (CRT) video displays emitted adequate EM emissions, which could be utilized from a distance to reconstruct the displayed content. Consequently, research into SCA explored the potential for exploitation by observing the current flow through transistors used in semiconductor logic gates. By introducing a small resistor into the circuit, researchers were able to measure the power consumption, demonstrating the concept of power-based SCA. Subsequent studies expanded this idea, investigating whether a device could also be compromised through processor emissions. Paul et al. and S. Messerges et al. illustrated how Differential Power Analysis (DPA) could be used to extract parts of the Data Encryption Standard (DES) key from smart cards(Paul et al., 1999; S. Messerges et al., 1999). These pioneering efforts validated the effectiveness of such attacks and laid the groundwork for future SCA research. Building on this success Brier et al. explored Correlation Power Analysis (CPA) as an alternative

approach to DPA, addressing some of its limitations and advancing the field further (Brier et al., 2004).

Building on these studies, Hanley et al. explored template attacks to recover encryption keys (Hanley et al., 2009). Extending this concept, Lerman et al. applied machine learning to relax some of the assumptions required for template attacks (Lerman et al., 2013). In a similar vein, Maghrebi et al. and Bartkewitz investigated the use of deep learning models and other machine learning(ML) techniques to perform key recovery attacks or enhance their effectiveness (Bartkewitz., 2016; Maghrebi et al., 2016). Other researchers have focused on improving SCA methodologies, such as Socha et al. who concentrated on AES implementations on a specific board (Socha et al., 2018). Xu and Heys and Chevallier-Mames et al. compared static and dynamic side-channel attacks against a masked S-box circuit (Chevallier-Mames et al., 2004; Xu and M. Heys, 2018) and Vosoughi and Köse who explored how distinguishers classifiers that separate possible keys into correct and incorrect could be combined to increase attack success rates (Vosoughi and Köse, 2019). Moos et al. also examined various factors influencing information extraction using static power analysis (Moos et al., 2019). Although extensive research has been conducted on side-channel attacks, there has also been considerable focus on developing defenses against these attacks. Chevallier-Mames et al. proposed straightforward modifications to algorithms to enhance their resistance to simple side-channel attacks (Chevallier-Mames, 2016). Similarly, Fournaris et al. explored various real-time embedded system attacks and discussed countermeasures to mitigate them(P. Fournaris et al., 2017).

2.3 Electromagnetic Side Channel Analysis (EM-SCA)

Electronic devices make noise in the form of EM signals at different frequencies when they're running. This includes computer systems and gadgets like IoT devices that use edge computing. When CPU works fast, they create these EM signals, and the pattern of these signals depends on the software being used and the data being processed. In edge computing devices, various parts like microcontrollers can give off this EM noise while they're working (Das and Boro, 2023).

In EM-SCA attacks, the attacker measures the EM waves coming from a device and secretly studies those waves to get sensitive info. The action that makes the device give off these waves is called EM emanation. These emanations can be on purpose or by accident. When it's on purpose, the

attacker intentionally makes the device produce these waves by messing with the electric flow in the device. This is done to create an EM response that the attacker can use. When it's unintentional, it happens because modern ICs are getting smaller and more complex, and this makes them give off these waves without anyone trying to do it (Das and Boro, 2023; Le et al., 2021). For an EM-SCA attack, the attacker needs to set up a special system to capture the EM signals from the physical device. They use tools like oscilloscopes, electromagnetic probes, and spectrum analyzers to study these signals coming from electronic devices as Figure 3. There are different methods to do EM-SCA attacks and steal the secret key used in encryption. These methods include SEMA (Simple Electromagnetic Analysis), DEMA (Differential Electromagnetic Analysis), and CEMA (Correlation Electromagnetic Analysis) (Le et al., 2021)

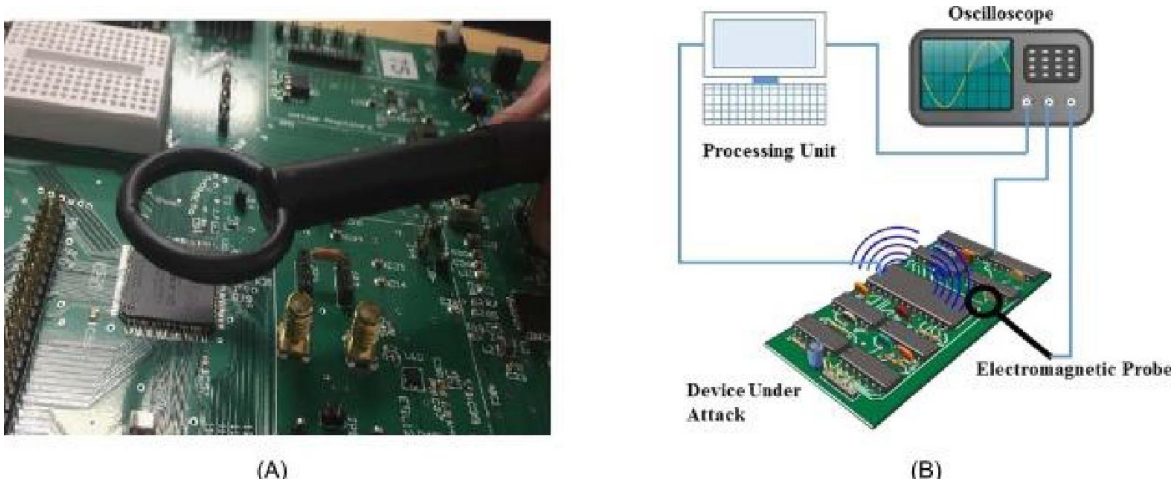


Figure 3: (A) An image showing the placement of an EM probe on an FPGA; (B) Setup for EM Side-channel investigation. Adopted from (Bhunia and Tehranipoor, 2018).

In a SEMA, the bad actor only needs to take a single reading or just a few readings to figure out the secret key or what software is doing on the device. SEMA works on devices where it can be physically see EM signals (Lakshminarasimhan, 2011). DEMA is for devices where it cannot be directly seen the signals. The attacker doesn't need to know anything about the device's architecture. DEMA compares the collected data to expected values and looks for spikes that suggest a correct guess. CEMA is a simpler version of DEMA. It uses something called the Pearson correlation coefficient to compare EM signals to predicted data ("EM Side Channels in Hardware Security: Attacks and Defenses," 2022). Usually, they use the Hamming distance and the Hamming weight model (see Hamming Theory) to find out where electromagnetic signals are leaking (Brier et al., 2004).

2.3.1 Equipment for capturing EM Radiation

Tirumaladass et al. have demonstrated the effective use of compact magnetic H-loop antennas for capturing EM radiation emitted by computing devices. In their research, they used deep learning TensorFlow Keras to classify three encryption algorithms (3DES, AES128 and AES256) with an accuracy of 95%. Near-field RF probes, also known as sniffer probes shown in Figure 4, play a vital role in pinpointing the sources of emissions emanating from circuit boards or devices. These probes find significant use in conducting preliminary EMC testing on circuit boards or electronic products. Their primary purpose is to identify any electromagnetic interference (EMI) that exceeds the regulatory standards. These probes must be capable of precisely measuring both the electric and magnetic fields of a module. To be effective, they need to maintain a consistent and unwavering frequency response to allow users to accurately locate emission sources (Tirumaladas et al., 2020).

Figure 4:Near field probes for oscilloscopes

2.3.2 Related Work

In recent times, there have been several initiatives aimed at using EM SCA to detect malware. Han et al. concentrated on the control flow integrity of PLC code execution. They utilized frequency representations of sections of electromagnetic signals to analyze the traces they collected. With these representations, they trained a model to recognize control flow transitions. By comparing

subsequently collected signals with the learned transitions, the team aimed to identify and alert on illegitimate control flows that did not align. When evaluating their model on Allen Bradley PLCs, they achieved a detection accuracy of 98.9% for malicious code executions (Han et al., 2017). Sayakkara et al. demonstrated that the electromagnetic radiation from IoT devices can potentially be used to gather valuable forensic information. The electromagnetic patterns from the CPU of these devices closely relate to their software activities. Using Raspberry Pi and Arduino Leonardo devices as examples, the study shows that various useful software behavior information can be detected. It was found that cryptographic algorithms running on IoT devices can be identified with 82% accuracy, and variations in software code behavior can be detected with 90% accuracy using a combination of EM-SCA techniques and ML model in real-world scenarios (Sayakkara, et al., 2019). A subsequent study by Sayakkara et al. also demonstrated that Random Forests (RF) and Deep Learning (DL) are effective ML models for predicting outcomes using EM-SCA (Sayakkara et al., 2020) .The dataset was created by employing ten distinct widely used sorting algorithms with varied computational time complexities and executing them on an Arduino Leonardo device to simulate a low-powered IoT device. These algorithms continuously sorted arrays of 100 elements that were randomly generated in ascending order. Additionally, further experiments were conducted to determine the performance of the methods based on the window size of raw samples and the number of training examples. The experimental findings indicate that it is possible to accurately predict the activity being performed with a high degree of accuracy (99.6%) using a Convolutional Neural Network (CNN) (Sayakkara et al., 2020).

Khan et al. proposed a framework that compares EM emanations with a reference dictionary using Euclidean distance. After capturing an initial trace, the team demodulates, low-pass filters, and samples the trace before proceeding with signal processing. They populate the reference dictionary by gathering known safe EM signals and dividing these signals into short-duration windows that serve as dictionary entries. During monitoring, the signal is also segmented into short-duration windows, which are then compared to the dictionary entries using Euclidean distance. The team reconstructs the signal using the entry from the dictionary that best matches, and this reconstructed signal is compared against the monitored signal to detect anomalies. If the moving average of the squared difference between the two signals exceeds a certain threshold, it indicates the presence of anomalous behavior. In tests conducted on various systems with specific attacks, their implementation achieved over 98% accuracy in all instances(Khan et al., 2021).

Alongside the research focused on embedded device processors, Ibrahim et al. introduced a framework to verify the authenticity of USB flash drives by analyzing their unintended magnetic emissions. In their study, the team successfully identified specific brands and models of USB flash drives based on the magnetic emissions generated during the boot process (Ibrahim et al., 2021). Yang and Jia present a novel black-box leakage detection method for SCA using one-way analysis of variance (ANOVA). Traditionally, SCA is used for security assessments to evaluate cryptographic devices' vulnerability, but dealing with large amounts of side-channel data can be challenging. The proposed approach helps improve efficiency by identifying leakage points that expose secret information (Yang and Jia, 2021).

Although EM-SCA has been effectively demonstrated in various contexts such as cryptographic key recovery and generic malware detection, current literature reveals a critical gap concerning its application to specifically detect unauthorized network API calls in embedded IoT devices. Prior research often lacks a detailed investigation into quantifying the character-level extent of information leakage from unauthorized API interactions using electromagnetic emissions. Furthermore, most forensic methodologies are invasive or computationally intensive, unsuitable for real-time monitoring in resource-constrained IoT environments. This study addresses these gaps by introducing a comprehensive non-invasive method leveraging EM-SCA, statistical validation, and advanced machine learning models to specifically detect and quantify unauthorized API modifications in IoT ecosystems.

2.4 Internet of Things (IoT)

IoT refers to a network of interconnected devices equipped with advanced capabilities to interact with other devices, individuals, and the physical environment to perform various tasks. To enable such interactions, IoT devices are integrated with sensors that facilitate seamless connectivity with the real world. Modern IoT devices incorporate a wide range of sensors, such as accelerometers, gyroscopes, microphones, and light sensors. These sensors enable IoT devices to detect changes in their environment and respond appropriately, enhancing their functionality. This ability to perceive and react to environmental changes allows IoT devices to make autonomous decisions(Bari et al., 2013; Sikder et al., 2018). The IoT is like a big net that connects many parts of our daily lives. It includes many of things, like smart devices and gadgets, that can talk to each other and make our

lives easier. So, it's basically about everything being connected and working together to make things more convenient for us (Husamuddin and Qayyum, 2017).

One core area is Monitor Environment. Environmental protection is carried out by using sensors and monitoring atmospheric conditions such as air and water quality. The habitats of wildlife are also observed to understand them better. The results of the monitoring are used to create improved ways to safeguard our environment. Manage Infrastructure is also very important area. One of the significant applications involves monitoring and controlling the operations of infrastructure such as roads, bridges, and railway tracks, etc. Changes in structural conditions can compromise safety and increase risk, so IOT infrastructure management can be used to monitor them. The quality of service can be improved. Automate Manufacturing Process is another area to use IoT. Real-time optimization of manufacturing can be achieved. The management of production and supply can be handled by using sensors and control systems. This also results in the rapid manufacturing of new products. Most popular area is Automate Home. Information about gas, water, and power can be sent to their utility company by an automated system. This process can enhance the efficiency of resources. The home automation process can be used to manipulate devices like washing machines, air conditioners, windows, doors, lighting, and refrigerators to achieve optimization. Manage Transportation is another popular area. This sector first used IOT technologies, which involve integrating light sensors, GPS, and GSM. The vehicle can serve as an entity and communicate with each other as well as roadside infrastructure. Sensors in the vehicles can be utilized to avoid collisions, manage traffic, and provide parking space. One of the very useful Manage Medical and Health Care. One of the promising areas of IOT technology is the transmission of a patient's vital parameters by medical devices to a platform like a secure cloud, where they are stored and analyzed. Special care can be given to the aged and chronic disease patients (Husamuddin and Qayyum, 2017).

In IoT applications, communication typically involves the connections outlined in Figure 5. They are Machine to People (M2P), Machine to Machine (M2M), and People to People (P2P) connections (Farhan et al., 2018).

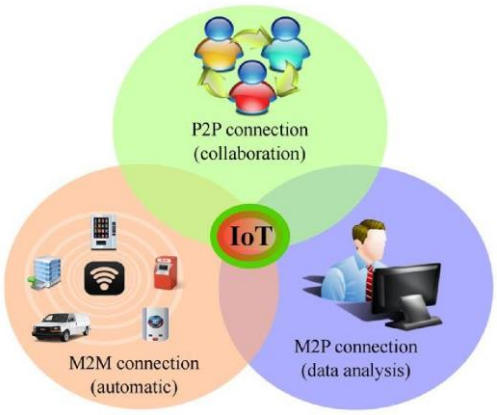


Figure 5: Internet of Things (IoT) elements

When delving deeper, the communication models are categorized into four models. They are, Device to Device Communication, Device to cloud Communication, Device to Gateway model and Back-End Data sharing model (Husamuddin and Qayyum, 2017).

2.4.1 Architecture of IoT

Generally, the architecture of IoT devices consists of four layers as mentioned in Figure 6. They are Sensing Layer, Network Layer, Data Processing Layer and Application Layer (Sikder et al., 2018).

- **Sensing Layer**: The sensing layer's main job is to notice things happening around the devices and collect real-world data. It's made up of various sensors. One key thing about IoT devices is they use multiple sensors. These sensors in IoT devices are often grouped using something called a sensor hub. A sensor hub acts like a central spot where many sensors gather and send their data to the device's brain for processing. Typically, there are three types of sensors. They are Motion Sensors, Environmental Sensors, and Position Sensors.
- **Network Layer**: The network layer is like a pathway that moves the data collected in the sensing layer to other devices that are connected. In IoT devices, this network layer uses various ways to talk to other devices. These methods include things like Wi-Fi, Bluetooth, Zigbee, Z-Wave, LoRa, and cellular networks. They help the data move between devices in the same network.
- **Data Processing Layer**: The data processing layer is like the brain of IoT devices. It's where all the data from the sensing layer goes for analysis. This analysis helps the device make decisions. In some IoT gadgets, like smartwatches and smart home hubs, this layer also stores

the results of past analyses to make things better for the user. Sometimes, it shares these results with other devices using the network layer

- **Application Layer**: The application layer is like the part of the IoT device that does the stuff users care about. It's where the results from the data processing layer are used to make different things happen with the device. This layer is all about serving the users and doing lots of different tasks. IoT has all kinds of uses, like making transportation smarter, improving homes, helping with personal care, and even in healthcare.

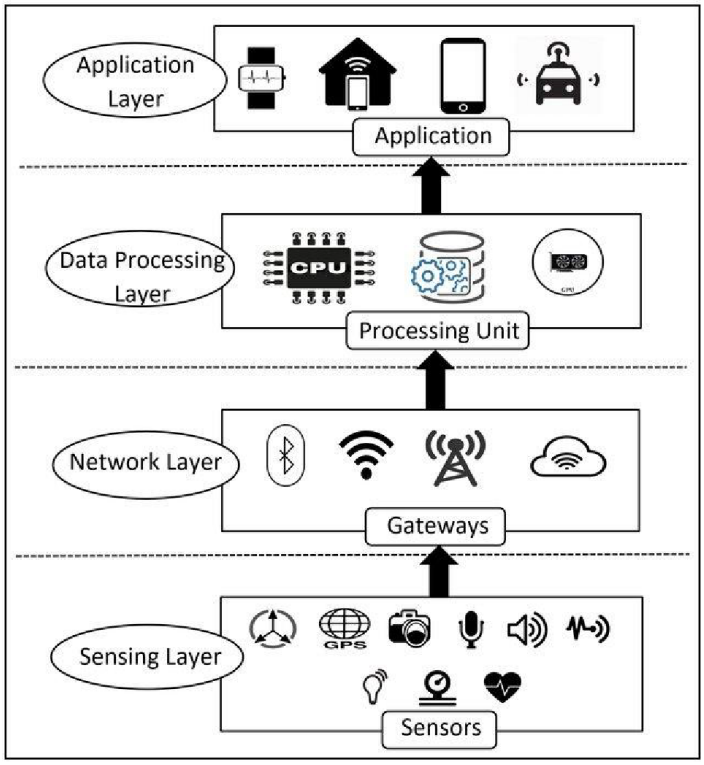


Figure 6: IoT Components and Architecture

2.4.2 Security in IoT Devices

The increase in technology brings risks and privacy concerns. Smart devices talk to each other and share data in a network. If one device is attacked, the entire system is in danger. For instance, if a machine is hacked, the production and important data could be at risk (Husamuddin and Qayyum, 2017).

In data security, there are four main concerns in the security of data. They are Confidentiality, Integrity, Authenticity, and Availability. Confidentiality means, the information sent between two

points must be kept private. Only the sender and receiver should be able to see the data. For instance, if someone hacks into infrastructure data, it could lead to damage to roads and bridges, and it might also put security at risk. Making sure the data sent between two points is correct is a big concern. It is calling integrity of data. So, it's important to keep the data accurate. For instance, in a manufacturing company, if a hacker tells the machines to stop producing, it's a very serious problem. Availability of data is making sure authorized users can get to the data for IoT. If users can't reach the data, that's a significant problem. In addition to these concerns authenticity of data is also a major concern. Authentication ensures that the received data is real and can be trusted. For instance, in the medical and healthcare system, a patient's information is sent to various medical centers. If a hacker changes this data before it's received, it could jeopardize the patient's treatment (Aldowah et al., 2019).

Security threats can be put into three groups. First, there are threats that attack the hardware, like the IC applications. Second, there are threats that use harmful software to take complete control of devices. Lastly, there are threats that grab and change data as it's being sent. Hardware trojans, side-channel attacks, tampering and dos attacks are few examples for the hardware threats to the IoT security (Williams et al., 2022).

In Hardware Trojan, the attacker watches, changes, or stops either the data stored in the circuit or the communication of the circuit using a Trojan. This happens when the device is being designed or made. Hardware Trojans can be split into two types: Combinational and Sequential. In Combinational, the Trojan turns on when a specific condition happens at certain places inside the circuit. In Sequential, the Trojan turns on when a specific sequence of uncommon logic values occurs at internal places (Williams et al., 2022). In a Side-Channel Attack, an attacker takes advantage of the release of physical information from a system while an application is running. The attacker uses non-invasive hardware-based methods by watching and measuring power usage, EM radiations, timing information, and sound. The collected information can be examined to get private details like cryptographic keys. Techniques for carrying out a side-channel attack include differential fault analysis, power monitoring attack, EM analysis attack, and acoustic cryptanalysis key extraction attack (Williams et al., 2022). More details regarding side-channel attacks are discussed under EM-SCA. Tampering happens when an attacker changes the data linked to an IC after it's used in an application. Many IoT devices are put in places without physical protection to

guard against an attacker getting physical access to the device or tampering with the software wirelessly. The attacker can add harmful hardware or software to change how an IC or the device behaves. DoS or DDoS occurs when attackers mess with the inside of an IC to stop users from using the service (Williams et al., 2022).

EM-SCA has gained significant traction as a method for extracting sensitive information from electronic devices, primarily due to its non-invasive nature and high potential for detailed information extraction. Previous research has primarily concentrated on leveraging EM-SCA to recover cryptographic keys and detect malware infections broadly within embedded systems. However, a detailed review of existing literature reveals limited efforts towards specifically detecting unauthorized modifications of network APIs in IoT environments through EM emissions. This represents a critical oversight since unauthorized network API interactions pose significant risks, including data leakage and compromised device integrity, particularly within resource-constrained IoT systems. Moreover, most current methodologies involving EM-SCA for digital forensic purposes focus on post-event analyses, which do not adequately address the need for real-time detection and prevention. Real-time detection of unauthorized API interactions is vital, as delayed detection can lead to significant irreversible data breaches or operational disruptions. The existing gap in literature around developing efficient, real-time detection mechanisms using EM-SCA highlights a crucial opportunity for further research. Addressing this gap would substantially enhance IoT security protocols and forensic readiness, enabling quicker response and mitigation of potential threats.

2.4.3 IoT Platform

NodeMCU (refer to Figure 7) Wi-Fi chips are cheaper than many other similar chips and it has a built-in MCU and TCP/IP layer. The main things to know about them are they are affordable, use less power, and work well. People use NodeMCU for things like automating their homes, making electronic stuff, and medical equipment. The in-built Wi-Fi component works with 2.4 GHz and under 802.11 Wi-Fi standards. NodeMCU can do two main things: it can link up with a Wi-Fi network that already have, or it can run its applications using the internet's HTTP protocol. Upon acquiring a NodeMCU module, it already comes preloaded with a specialized set of commands known as AT commands. The NodeMCU is a well-known development board that uses the ESP8266 as its core. It includes the ESP12 module, which houses the ESP8266 System-on-Chip

(SoC). Additionally, it's equipped with a USB connector and pins that are compatible with breadboards, making it a convenient choice for experimenting and creating projects with the ESP8266. The ESP8266 was made for things like mobile devices, wearable tech, and IoT gadgets, and its main goal is to use as little power as possible. It does this by using some special techniques. It has three modes for saving power: one where it's active, one where it takes a nap (sleep mode), and another where it's in deep sleep (deep sleep mode) (Espressif IOT Team, n.d.; Gupta and Johari, 2019; "NodeMCU ESP8266 Detailed Review," n.d.).

NodeMCU comes in two versions: version 0.9 and version 1.0. Version 0.9 uses the ESP-12, while version 1.0 uses the ESP-12E, where the "E" stands for "Enhanced". The ESP8266EX has a special chip inside called the Tensilica L106. It's a 32-bit processor known for using very little power, and it can go as fast as 160 MHz This fast clock helps with sending and receiving data. The clock is made using a special type of rock inside the ESP8266EX and an external one. This clock can go anywhere from 24 MHz to 52 MHz (Espressif IOT Team, n.d.).

Figure 7: LoLin style NodeMCU

2.5 Digital Forensics

Digital forensics focuses on uncovering and analyzing digital evidence to investigate cybercrimes, using legal and technical expertise to collect, study, and report findings. Conversely, anti-forensics aims to obscure, alter, or destroy evidence, challenging its credibility. This field is vital due to the persistent threats of data breaches and hacks, employing specialized techniques to analyze electronic data. Computer forensics, a subset, handles the discovery, preservation, and presentation of digital evidence. As cybercrime evolves with new tools and attacks on social networks and systems, digital forensic methods improve, but hackers counter with anti-forensic tactics to erase or obscure evidence (Biggs and Vidalis, 2009; Majed et al., 2020; Tiwari et al., 2021).

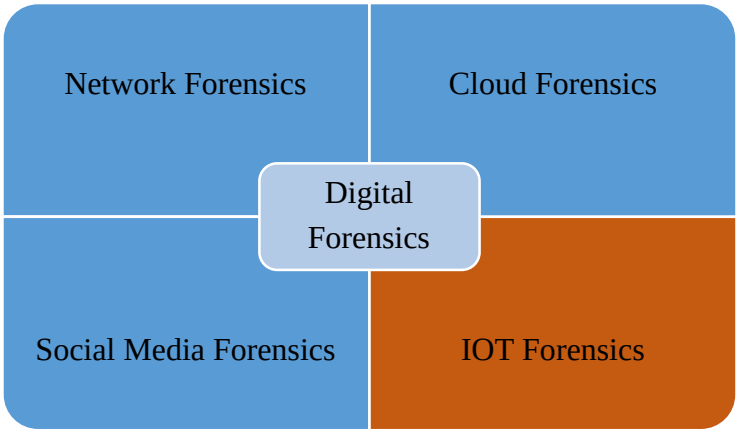


Figure 8: Common Types of Digital Forensics

Digital forensics has diversified into four areas as Figure 8. IoT forensics, for example, investigates cybercrimes involving IoT devices and their data, which are prone to threats like ransomware, DoS attacks, and mass surveillance. While IoT systems enhance connectivity in homes and businesses, they introduce vulnerabilities. From a forensic perspective, data from as discussed in Internet of Things (IoT) devices offers valuable insights for criminal investigations, often surpassing the utility of traditional computer systems by capturing visual events with minimal user interaction (Tiwari et al., 2021).

2.6 Signal Processing

A signal is classified as a periodic signal when it displays a clear pattern and repeats itself regularly at defined time intervals. On the other hand, a signal that doesn't repeat at consistent time intervals is referred to as an aperiodic or non-periodic signal (see Signal Theory).

There are two primary types of signals commonly encountered there are analog and digital. An analog signal is characterized by its continuity, meaning that the signal's changes over time reflect another continuously varying quantity, resembling another time-varying signal. For instance, in analog audio signals, the instantaneous voltage corresponds directly to the sound pressure and changes continuously. In contrast, a digital signal represents a sequence of discrete values using a continuous quantity. It's constructed from a finite set of waveforms of a physical quantity, and it

signifies a sequence of distinct values. A logic signal is a specific type of digital signal with only two possible values, often used to convey binary information.

2.6.1 Digital Signal Processing

To work with signals on modern computers, they must be in a digitized format. This means that analog signals need to transform digital signals before they can be input into computers. This conversion process is known as Analog-to-Digital Conversion (ADC).

In the signal processing field, aliasing occurs when frequency components overlap due to a sampling rate that falls below the Nyquist rate. This overlap leads to distortions or artefacts when reconstructing the signal from samples, causing the reconstructed signal to deviate from the original continuous signal. To avoid aliasing errors, the sampling rate must be a minimum of twice the highest frequency present in the analog signal (Smith, 1997).

2.6.2 Fast Fourier Transform (FFT)

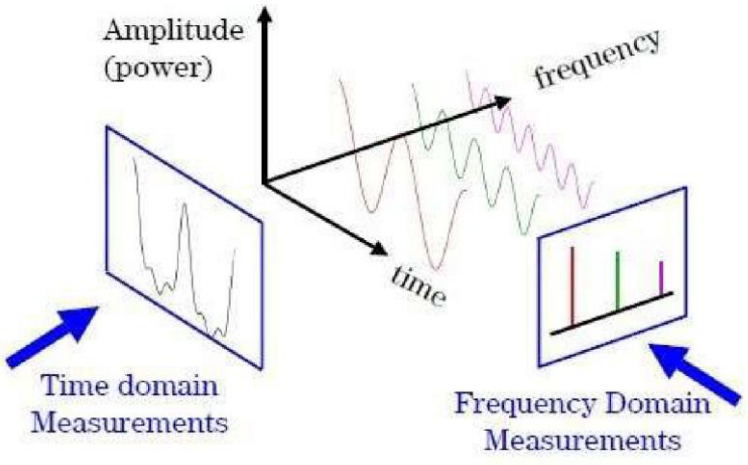


Figure 9: Every signal can be depicted in a three-dimensional space involving time, amplitude, and frequency. Complex waveforms can be constructed by combining multiple sine waves with varying amplitudes, phases, and frequencies, as illustrated.

Electrical signals come with both time and frequency domain representations. In the time domain, we express voltage or current as a function of time. Many people find time-domain representations of signals quite familiar. Signals observed on an oscilloscope, for instance, are depicted in the time domain, and digital data often relies on voltage as a function of time. However, signals can also be

represented in terms of magnitude and phase as a function of frequency. A signal can consist of several frequency components that remain concealed in the time domain. A time domain signal must be broken down and its individual frequency components extracted to be transformed into the frequency domain (refer to Figure 9). This breakdown and extraction can be achieved through a process called Fourier transform. Among the various algorithms used for Fourier transforms, the Fast Fourier Transform (FFT) is the most used one due to its speed and efficiency in comparison to alternative methods (Smith, 1997).

Similar to FFT plots, Power spectrum Density (PSD) as Figure 10 plots show the power spectrum density on the y-axis. Nevertheless, time-related information is not provided by FFT or PSD graphs.

A spectrogram (as Figure 11) is used to show frequency and temporal information at the same time. Time data is displayed along one axis of a spectrogram, while frequency data is displayed along the other. To show how each frequency component's amplitude varies over time, color coding is used.

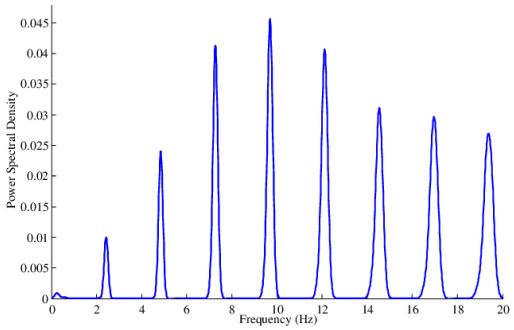


Figure 10: Power spectral density plot derived (Mcnames et al., 2002)

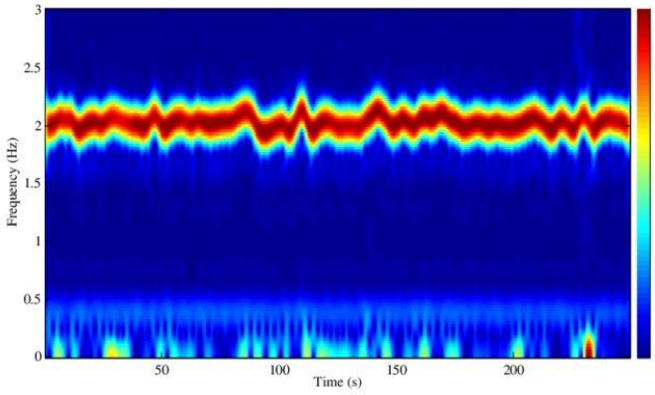


Figure 11: Spectrogram plot derived from (Mcnames et al., 2002)

2.7 Software-defined Radio (SDR)

A SDR system uses software for signal modulation and demodulation, performing signal processing on a general-purpose computer or reconfigurable digital electronics. This enables adaptability to new radio protocols through software updates (Nesimoglu, 2010). SDRs are vital in

cell phone services, enabling real-time adaptation to evolving radio protocols. Their hardware typically includes a super heterodyne RF front end for converting RF signals between analog and digital forms (see Figure 12) and ADCs/DACs for transforming intermediate frequency (IF) signals. While currently suited for basic radio modem technologies, SDRs are expected to dominate future radio communications (Garg, 2007).

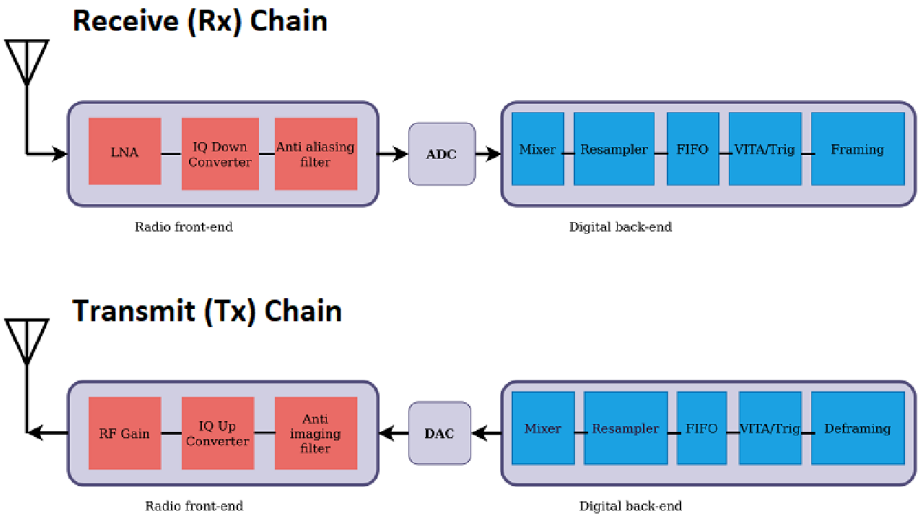


Figure 12: High-level summary of the different parts that make up an SDR systems transmit (TX) and receive (RX) chains

2.7.1 HackRF One

Figure 13: HackRF One device with its default antennas

The HackRF One (refer to Figure 13) is a versatile half-duplex transceiver in the form of a wideband SDR tool. It was developed and produced by Great Scott Gadgets. The HackRF One offers the capability to both receive and transmit signals over a broad frequency range, spanning from 1 MHz to 6 GHz. The maximum output power can reach up to 15 dB, with the exact output power dependent on the specific frequency band in use. This device seamlessly interfaces with well-known software-defined radio software applications like GNU Radio and SDR (Martoyo et al., 2020).

GNU Radio packages for Python enable the complete software construction of SDR applications, complemented by a GUI tool called GNU Radio Companion (GRC). GRC allows users to design SDR applications as Python programs via a drag-and-drop interface ("What Is GNU Radio," n.d.) as explained in Software Setup. Although data samples from various SDR hardware may differ in bit length and encoding, the GNU Radio library standardizes these samples when controlling the hardware. The standardized data is formatted as In-phase/Quadrature-phase (I/Q) Data.

Every EM data sample in the GNU Radio library is represented by two 32-bit (4-byte) floating-point values, which combine to generate a complicated I/Q sample. As a result, each sample of EM data is an 8-byte complex value. These I/Q samples are produced continuously, matching the SDR hardware's sample rate, when EM data is recorded using SDR hardware and the GNU Radio library. In order to avoid aliasing, the sample rate must to be twice the frequency of the signal of interest, according to the Nyquist sampling theorem. This restriction mostly affects real valued sampling and might cause difficulties when recording high-frequency sounds. SDR devices, however, get around this restriction by using I/Q sampling. First, they use a local oscillator in the analogue hardware front-end to move the center of the interest frequency range to the baseband, or 0 Hz. They then create intricate I/Q samples. SDRs can employ a sampling rate lower than the frequency of the signal they are collecting thanks to this technique. Furthermore, SDR devices' sample rates match the signal bandwidth they are collecting (Sayakkara and Le-Khac, 2021).

2.8 Chapter Summary

Although literature of Electromagnetic Side Channel Analysis (EM-SCA) have effectively demonstrated capabilities in detecting unauthorized operations and anomalies broadly, the specialized area of network API interactions remains notably underexplored. APIs serve as a

primary communication interface between IoT devices and external networks or cloud services, making them a critical security juncture. Unauthorized API interactions are particularly dangerous, as they can silently exfiltrate sensitive data without noticeable system disruption. Thus, explicitly studying EM emissions specific to API calls and their modifications could offer valuable new insights into detecting such subtle yet critical security violations, significantly contributing to the IoT security literature.

To address these identified gaps, the current study introduces an innovative non-invasive framework combining EM-SCA, statistical testing, and machine learning methodologies. By examining EM emissions generated specifically from network API interactions, this research seeks to identify subtle electromagnetic signatures that distinguish legitimate API activities from unauthorized modifications. Furthermore, by quantifying the extent of character-level information leakage through advanced machine learning algorithms, this study aims to provide a robust, real-time security measure tailored explicitly for embedded IoT devices. Ultimately, this research contributes significantly to the security, forensic, and IoT communities by proposing practical methodologies for enhancing detection capabilities and safeguarding data integrity in increasingly interconnected embedded environments.

CHAPTER 3 - METHODOLOGY

This chapter provides an overview of the research and development methodologies selected to continue this research. This chapter discusses the principles, research design, research technique, and the process involved in the research study. The study method includes the solutions to the research aims and objectives outlined in the introduction chapter.

3.1 Research Methodology

This research adopts pragmatism as its foundational philosophical framework, prioritizing practical outcomes and real-world applicability over abstract theorization. Pragmatism emphasizes the use of methodologies and tools that deliver measurable and effective results. Within this study, the integration of statistical analysis and machine learning techniques reflects this commitment to achieving tangible outcomes, particularly in the context of detecting unauthorized API activities in embedded systems. The research adheres to a non-invasive strategy, aligning with the pragmatic principle that effective solutions should be both functional and minimally disruptive in practice. This pragmatic orientation not only facilitates the identification of anomalous API behaviors but also underscores the role of practical technological interventions in enhancing system security.

The research methodology is further distinguished by its adoption of an inductive approach, which involves deriving generalizations from specific empirical observations. This process begins with the systematic collection of detailed data on API operations, followed by rigorous statistical analyses aimed at distinguishing between authorized and unauthorized behaviors. Insights gained through these analyses inform the development of a machine learning model designed to detect subtle indicators of information leakage. By evaluating the model using real-world data from embedded systems, the research substantiates the model's predictive capabilities, thereby reinforcing the efficacy of inductive reasoning in generating robust, evidence-based conclusions.

A core aspect of the research's strategic implementation of machine learning involves the analysis of EM emissions as a means to detect anomalies within embedded systems. Leveraging quantitative EM data, the study employs sophisticated statistical computations to identify patterns indicative of unauthorized API activities and potential information leakage. The machine learning algorithms are meticulously calibrated to distinguish between normal operational signatures and those

suggestive of tampering or security breaches. The model's accuracy and reliability are validated through extensive comparisons between predicted outcomes and observed instances of firmware manipulation. This methodological rigor enhances the credibility of the findings while demonstrating the potential of innovative, data-driven strategies for safeguarding embedded systems against cyber threats.

3.2 Overview

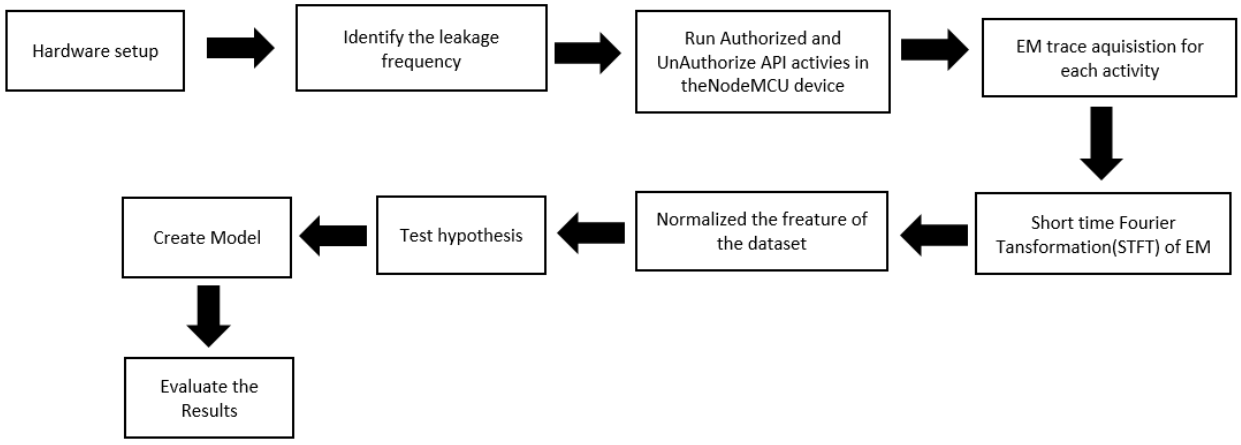


Figure 14: Overview of the research process

The methodology (refer Figure 14) begins with the hardware setup and identification of the leakage frequency of interest. Subsequently, both authorized and unauthorized API activities are executed on the NodeMCU device, followed by EM trace acquisition for each activity. The collected EM traces undergo short-time Fourier transformation (STFT) to extract time-frequency representations. The transformed traces are then normalized to create a uniform dataset for analysis. Finally, hypothesis testing is performed on experiments validate the observations, with the results presented accordingly. This chapter provides details explanations for each step of the process.

3.3 Procedure for Data Acquisition

3.3.1 Device Under Test (DUT)

The experiment in this section used a Lolin NodeMCU device as the DUT. This device includes the in-build ESP8266MOD Wi-Fi module which is designed and manufactured by Espressif Systems. which is commonly found in IoT devices like smart light bulbs, meters, and sensors. The

ESP8266MOD is equipped with a Tensilica L106 32-bit RISC processor that normally runs at a default speed of 80 MHz, but it can be overclocked to 160 MHz if needed ("NodeMCU ESP8266 Detailed Review," n.d.; Robyns et al., 2020). Moreover, Hardware Setup section outlines the experimental arrangement.

Figure 15: LoLin style NodeMCU measures 58mm x 32mm with a pin spacing of 0.1" between pins and 1.1" between rows derived from https://pino-tech.eu/product/nodemcuv3/

3.3.2 Platform / Software Specification

The host computer, an Asus with a CPU corei7 8th Gen 3.25 GHz processor and 16 GB RAM running Ubuntu 22.4 LTS, played a pivotal role in the experimentation process. Equipped with essential tools, including the PlatformIO, Gqrx SDR tool, and GNU Radio Companion, it provided a robust platform for data acquisition. This computer served as the central hub for controlling the experiments as explained Hardware Setup, managing the connected HackRF One SDR Device, and processing the captured electromagnetic data. The specifications of the above host computer ensured a reliable and efficient environment for conducting experiments and analyzing the outcomes seamlessly.

3.3.3 Determining Data Acquisition Parameters

The successful acquisition of EM data from an IoT device is contingent upon several critical parameters:

1. **Information-Leaking Signal Frequency:** To effectively detect EM radiation from a target IoT device, accurately identifying the frequency at which information leakage occurs is paramount. Different frequencies can provide critical forensic insights, yet a standardized methodology for pinpointing the precise leakage frequency remains elusive. One consistently reliable frequency, however, is the clock frequency of the IoT device's MCU. For instance, a NodeMCU device typically operates at a clock frequency of 80MHz (Callan et al., 2016). In practical scenarios, the presence of external electromagnetic noise, originating from surrounding devices or environmental factors, can interfere with observations at this frequency, complicating the detection process. To mitigate such interference, higher harmonic frequencies integer multiples of the original clock frequency can be leveraged as alternative frequencies for monitoring information leakage. These harmonics retain the essential signal characteristics while reducing susceptibility to overlapping noise, thereby enhancing the reliability of EM emission analysis for forensic and security applications (Sayakkara, et al., 2019). Furthermore, upon systematically analyzing the harmonic frequencies in NodeMCU devices, it was determined that the leakage frequency is 390MHz (refer Explore the Leakage Frequency)

2. **Sample Rate of Data Acquisition:** As outlined in Section SDR, the sample rate and bandwidth of SDR hardware are equal. Setting a specific sample rate impacts the EM signal acquisition process in several ways. Higher sample rates capture more signal detail, while lower rates risk data loss. Additionally, higher sample rates increase the bandwidth around the center frequency, which is beneficial for capturing multiple information-leaking frequencies near the IoT device's clock signal. Thus, maximizing the sample rate is advantageous, limited only by the computational demands it imposes. This work uses the HackRF SDR, which supports sample rates up to 20MHz, the rate employed in all experimental evaluations.

3. **Duration of Data Acquisition**: When observing the EM radiation of an IoT device, the data acquisition period must capture the internal activity of interest. However, since IoT firmware activities are typically repetitive, it is rarely necessary to precisely time the sampling to a specific event. Instead, a sufficiently long observation period can capture all relevant leaked information. However, longer sampling durations produce larger EM traces, making data handling and processing more challenging. Therefore, the sampling duration must be carefully balanced not too long or too short. In this work, EM traces were acquired with sampling periods of up to 20 seconds, adjusted based on experimental needs.

3.3.4 Type of APIs for Data Collection

In this study, two categories of APIs which are authenticated and unauthenticated are employed to simulate realistic IoT interactions and potential attack vectors. Authenticated APIs are Manufacture provided APIs for the firmware to ensure only trusted requests can access critical data or functionality. Unauthenticated APIs, by contrast, accept requests without verifying the caller's identity, making them potential entry points for malicious behavior. These API calls serve as the basis for measuring EM signals during both normal and tampered firmware operation

Authenticated APIs

- **Random number Generation API (RNG API)**

  Produces a random number that the device uses for subsequent operations (e.g., seeding other functions or adding randomness to requests). This endpoint is protected to avoid unauthorized manipulation of random values, which could introduce cryptographic weaknesses or unpredictability in the system.

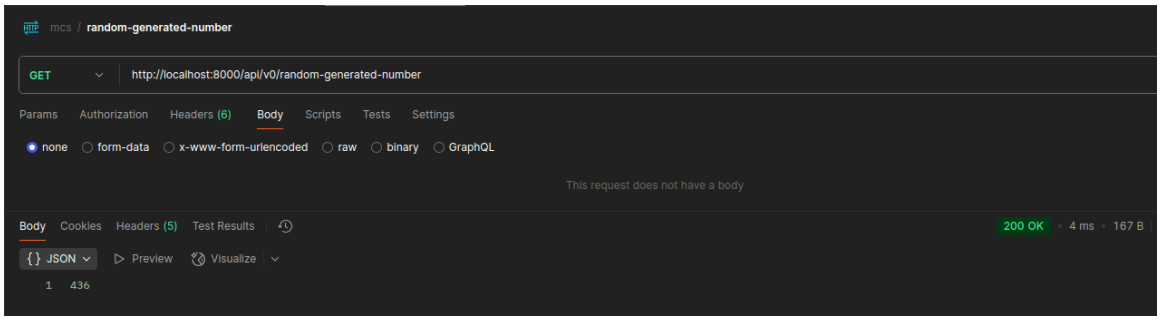


Figure 16: Postman Request and Response of Random Number Generation API

- **Subscriber API**

  Retrieves a list of subscribers (e.g., phone numbers or device IDs) that should receive specific notifications. Ensures that only authorized systems can access and manage sensitive subscriber data, thereby reducing the risk of privacy breaches.

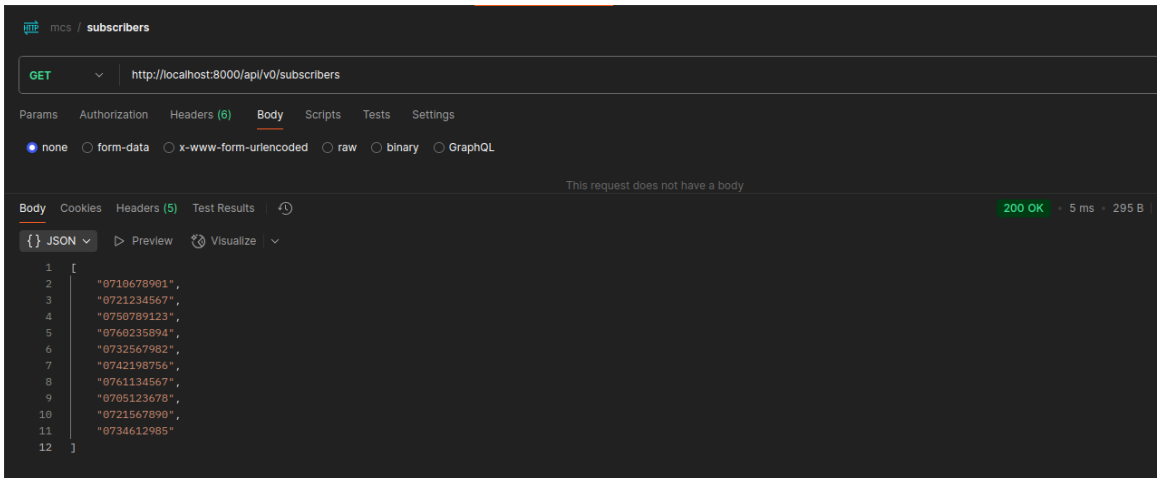


Figure 17: Postman Request and Response for Subscriber API

- **Weather API**

  Accepts location data (e.g., longitude and latitude) as a POST request and returns weather information (temperature, humidity, air quality, etc.). Since weather data can be critical for time-sensitive or location-based IoT functions (e.g., smart agriculture, logistics), legitimate requests must be authenticated to prevent unauthorized alterations or spoofing of weather information. And the random generated number that what we received from the previous API would pass as the query parameter.

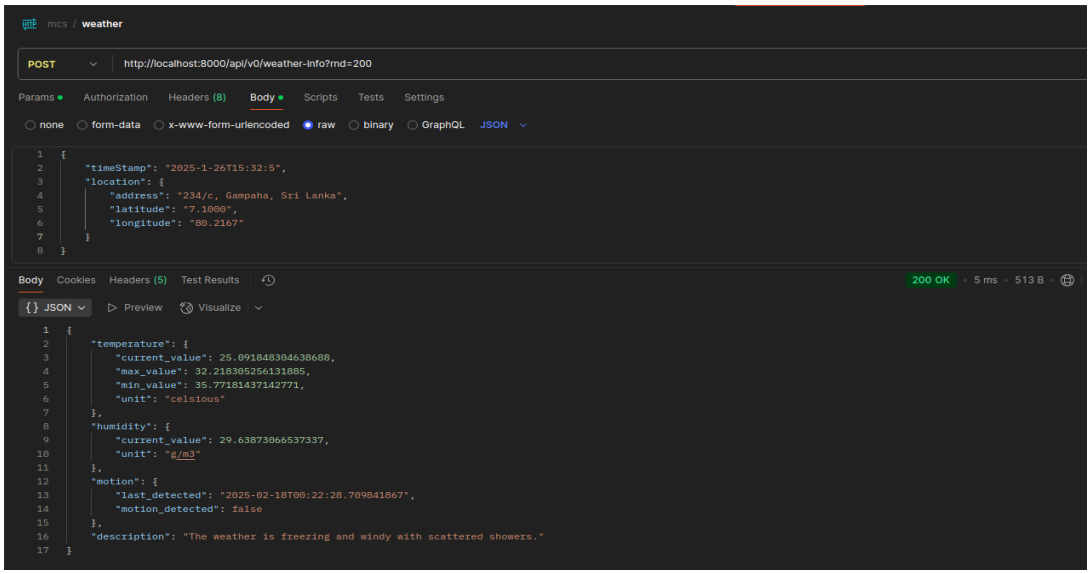


Figure 18: Postman Request and Response for Weather API

- **Notification API**

  Sends messages or alerts to the subscriber list obtained from the Subscription API. Requires authentication to protect against spam or malicious notifications that could disrupt users or compromise trust in the service. Here the MSISDNs that are received from the subscriber API and Weather Info that are received from the Weather API send in request body.

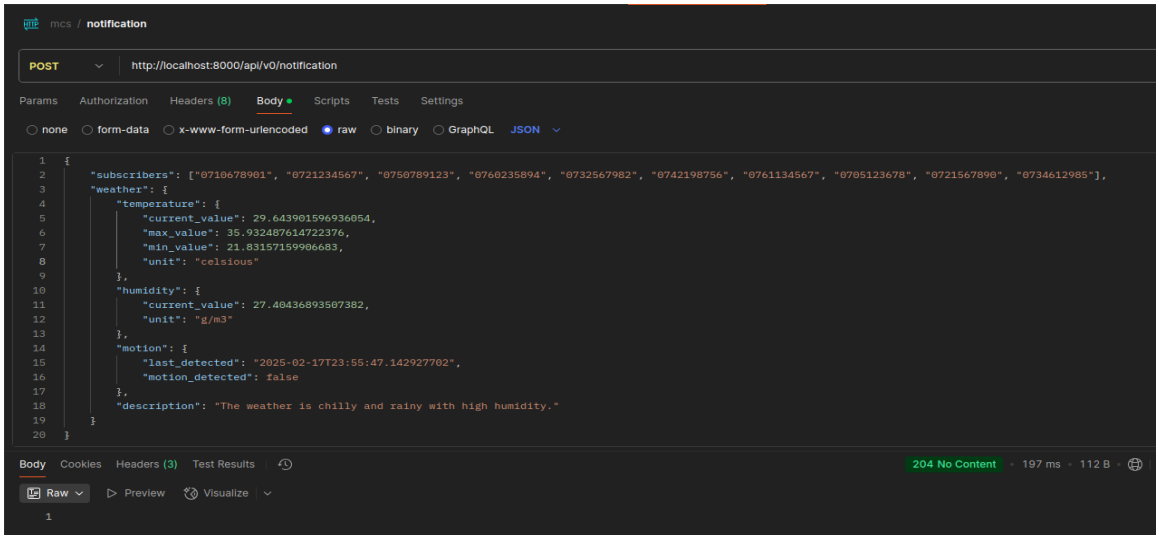


Figure 19: Postman Request and Response for Notification API

Un-Authenticated APIs

- **PING API**

  the NodeMCU initiates a GET request to a malicious server, signaling to unauthorized parties that the device is currently active. Rather than simply checking network reachability, this outbound call confirms the MCU's operational state to an attacker, potentially allowing them to coordinate further exploitation or data exfiltration. By monitoring EM side-channel emissions, researchers can detect these unauthorized connections and distinguish them from legitimate API calls in real-time.

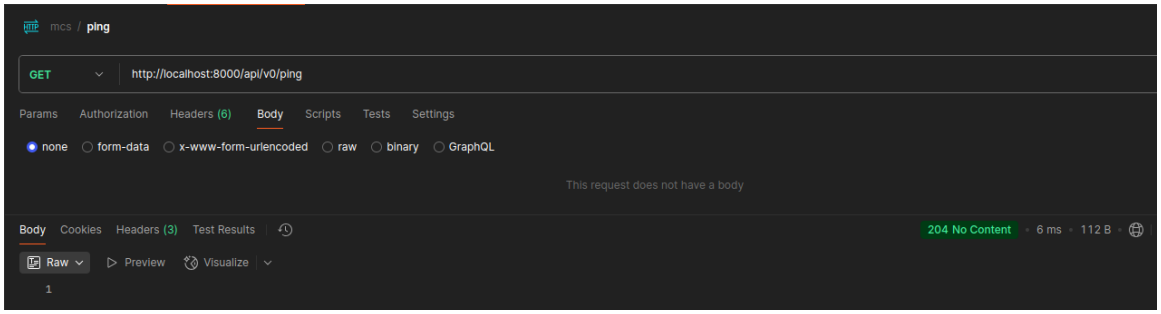


Figure 20: Postman Request and Response for PING API

- **Data Gathering API (DG API)**

  DG API returns basic device or system information, potentially including version data, environment variables, or other metadata. Since it does not require credentials, this API poses a security risk malicious actors can query valuable system information to plan advanced attacks. By observing EM traces from these calls, it becomes possible to distinguish normal usage from unauthorized querying attempts. The data structure of this API is in GitHub.

These APIs collectively provide a controlled environment to study how firmware handles both legitimate (authenticated) and potentially malicious (unauthenticated) requests. By comparing the electromagnetic signals emitted during each type of API call, researchers can identify anomalies tied to unauthorized modifications in the firmware and quantify any resulting data leakage.

3.3.5 Modification for Authenticated Code Base

Figure 21: Sequence diagram for API activities in the in the Authenticated firmware

Figure 21 Illustrate the sequence diagram how the MCU's authenticated firmware communicates with the API server. Each step in the flow represents a legitimate request and response exchange, with no interference or tampering by unauthenticated parties. The MCU sends requests as expected

by the authenticated code, and the server responds accordingly, ensuring only authorized interactions take place.

Targeted modifications were introduced to the MCU firmware to enable the capture of EM traces under various conditions. This approach facilitates the systematic evaluation of the hypotheses outlined in introduction. The following list details the specific firmware adjustments made, including modifications to API calls and network behaviors, to assess their impact on the device's EM emissions.

1. **Blocking Random Number Generation and Substituting a Maliciously Fixed Integer**
   In this first modification (refer Figure 22), the malware-injected firmware prevents the legitimate RNG API call from executing. Instead, it replaces the dynamic parameter with a maliciously fixed integer in the subsequent POST /api/v0/weather-info request. This scenario simulates how compromised code can undermine the integrity of the device's API-based interactions.

Figure 22: The Sequence diagram for remove RGN API

2. Insertion of PING API Calls at Multiple Stages

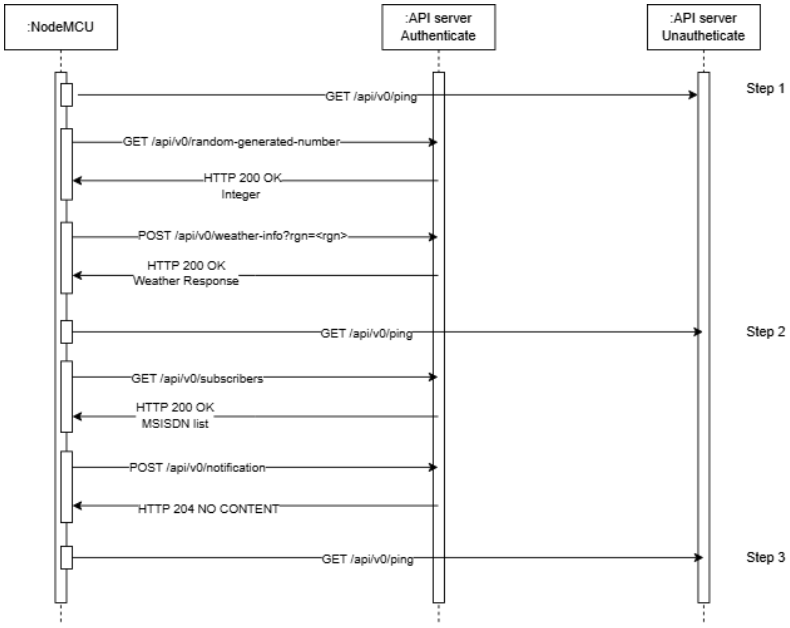


Figure 23: Sequence diagram for invoke PING API for difference location in the NodeMCU code

In this second modification (refer Figure 23), a PING API call targeting an unauthenticated or potentially malicious endpoint is introduced at three distinct points in the normal sequence of authenticated API requests:

1. prior to the first request. (PING first)
2. midway through the authenticated calls (PING middle)
3. immediately following the final request. (PING last)

By capturing the EM traces for each of these scenarios, the experiment aims to assess how unauthorized or spurious network interactions at different stages influence the device's emission patterns. Through this comparison, we seek to determine whether the presence and timing of the PING API call can be reliably detected via side-channel analysis, thereby providing deeper insights into how such modifications impact the integrity of the overall communication flow.

3. Introduction of Data Gathering API

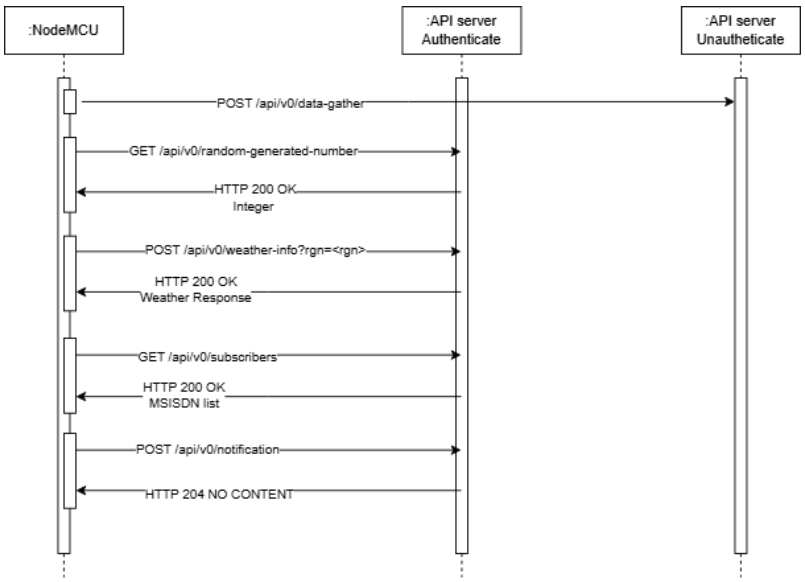


Figure 24: Sequence diagram for adding DG API

In this subsequent modification (refer Figure 24), a Data Gather endpoint is added prior to the first request. When invoked, this API collects potentially sensitive information, such as device parameters, network data, system settings, or control configurations thereby increasing the risk of unauthorized data leakage. Unlike the previously discussed PING API, which transmits minimal or no payload, the Data Gather API involves transferring a significant payload containing critical device details. By capturing EM traces of these interactions, the study investigates whether the presence and size of a payload can lead to distinguishable emission patterns, thereby offering deeper insights into the integrity and security implications of unauthenticated endpoints within the IoT ecosystem.

4. Systematic Variation of Weather API Payload Size

In the final modification, the payload size for the Weather API request is intentionally varied to evaluate how increasing character counts might influence EM emissions and potential information leakage. Specifically, three payload thresholds are tested:

1. Payload Under 500 Characters (low data leakage)
2. Payload Between 500 and 1,000 Characters (moderate data leakage)
3. Payload Exceeding 1,000 Characters (high data leakage)

For each threshold, the modified firmware submits the weather-related API call and then captures the corresponding EM traces. By comparing these recordings, the research examines whether the growing payload size results in observable changes in side-channel emissions, thereby offering insights into the potential correlation between data volume and leakage risk.

3.3.6 Data Collection

A HackRF SDR device was configured to sample EM radiation from NodeMCU device at a sampling rate of 20MHz. When capturing EM data using a SDR device, employing extremely high sample rates is essential to maximize the amount of information collected. However, this results in significantly large file sizes, even for relatively short time windows. For instance, consider a scenario in which a HackRF SDR captures EM data at a sampling rate of 20MHz for a duration of one minute. Each sample generated by the HackRF device through the GNU Radio library consists of two 32-bit floating-point values, representing the Quadrature (Q) and In-phase (I) components in an interleaved I/Q stream format. Consequently, each I/Q sample is 8 bytes in size.

Given these parameters, the total size of a one-minute signal capture is approximately 9 GB (calculated as 8 bytes × 20 MHz × 60 s ≈ 8.94 GB). To effectively apply electromagnetic EM-SCA techniques and machine learning algorithms, thousands of such EM traces are required, making data storage and management highly challenging (Sayakkara, n.d.). Data which are related to hypothesis 1 collected during 20 seconds and for the hypothesis two data collection is set to one min duration as Figure 25.

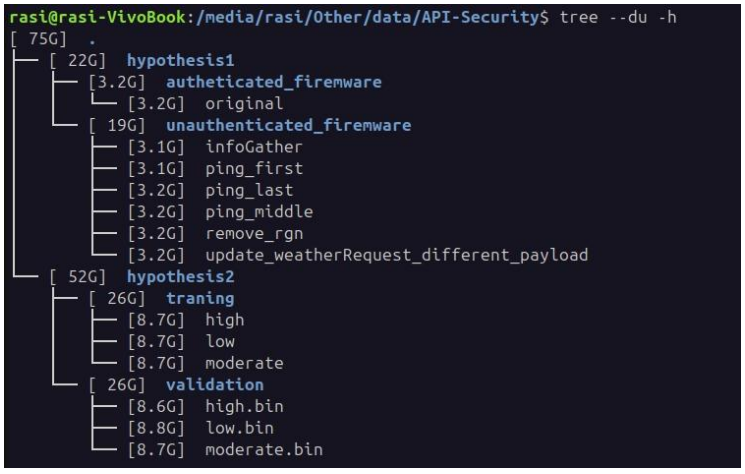


Figure 25: EM Trace binary data store represent in hierarchically

This would be achieving a reliable SVM or neural network performance with EM side-channel data requires ~10,000–20,000 samples for 20 s traces (~1,000 features) and ~50,000–100,000 for 60 s traces (~3,000–5,000 features). Weak signals and class imbalances demand vast data to detect meaningful effects, while robust feature importance (e.g., Spearman's $\rho > 0.9$) only emerges after thousands of independent captures. Only such large-scale data ensures models reflect true code behavior, not noise (Kalousis et al., 2006).

3.4 Data Preprocessing

The EM traces acquired during data acquisition may not fully capture the activities of interest within a single recording window and may show significant variability in their time-domain durations due to a variety of operational circumstances. Non-deterministic delays brought about by communication between the NodeMCU and the host system, inherent variations in the execution times of individual API calls, and the unavoidable latency in the HackRF data capture process which influences the start and stop times of sampling are all contributing factors.

The raw EM traces are not appropriate for direct use in ML classification tasks or the statistical analysis due to these timing and length differences. To solve this, a STFT is used to convert each trace into the frequency and Time domain (Smith, 1997). Achieving a balance is necessary when selecting an STFT window size; too wide windows result in cumbersome feature vectors, while too small windows run the risk of omitting important information. A window size of 2048 samples and a 256 overlapping ratio is used. This creates a two-dimensional dataset with frequency and time dimensions, generating 2048 features for each of the 10,000 time samples, was chosen as the best compromise based on empirical testing.

3.4.1 Normalization

Normalization is a fundamental step in the data preprocessing phase of machine learning. It is applied to translate features inside a dataset that may reside on various ranges of values into a standardized scale, often ranging from 0 to 1 or, in certain situations, -1 to 1. This standardization is crucial as considerable discrepancies in feature ranges might adversely affect the learning process. Various normalizing methods are available, including the standard scaler and the min-max scaler.

In the typical scaler approach, scaling is conducted individually on each feature by computing relevant statistics from the dataset. On the other hand, the min-max scaling strategy entails scaling each feature independently using the minimum and maximum values provided in the dataset. Standard Scaler to preprocess and normalize specified EM trace, which is a frequent preprocessing step in machine learning workflows to ensure that features are on the same size.

3.5 Statistical Testing

Statistical hypothesis testing is central to validating whether the EM signals truly differ between legitimate and tampered firmware conditions. In side-channel security evaluations, t-tests are widely favored for their ability to detect even subtle leakages. Both practitioners and researchers have adopted t-test–based methods (e.g., Welch's t-test, paired t-test) as part of side-channel resistance validation programs (Goodwill et al., 2011).Furthermore, (Ding and Eisenbarth, 2016) emphasize that t-test procedures are robust, relatively simple, and computationally efficient compared to alternatives like mutual information analysis.

3.5.1 T-test

Sensitivity to Minor Variations, Facilitation of Standardization, and Resilience to Noise Side-channel signals often display low-amplitude leakages that are masked by considerable noise. The t-test's ability to identify minor mean differences enables it to detect these slight leakages without depending on heavy modeling assumptions. By contrasting two sets of measurements, typically categorized as "fixed" versus "random" inputs, the test can ascertain whether any sensitive intermediate value is affecting the observed electromagnetic traces. Furthermore, Goodwill et al. emphasized a standardized t-test framework, commonly referred to as Test Vector Leakage Assessment (TVLA), offers a uniform methodology for gathering data under regulated settings and subsequently conducting a t-test (Goodwill et al., 2011). This standardization makes leakage testing reproducible and reduces the need for specialist operator skills. While other side-channel techniques (e.g., CPA, template attacks) may be intricate or tailored to certain attackers, a t-test only assesses statistically significant changes, irrespective of an adversarial approach. Ultimately, due to the susceptibility of physical experiments to environmental variability, research by Ding and Eisenbarth demonstrates that a paired t-test design, which captures "fixed" and "random" traces in

close temporal proximity, can alleviate drift and other common noise sources, thereby enhancing detection power for minor leakages(Ding and Eisenbarth, 2016). The widely used Welch's t-test, which does not necessitate equal variances, demonstrates robustness when noise levels differ between genuine and altered situations.

3.5.2 Analysis of Variance (ANOVA)

One-way ANOVA is a statistical approach used to compare the means of three or more independent groups to discover whether at least one group mean is substantially different from the others. In its conventional use, it examines whether the observed differences among sample means are attributable to actual effects or merely random fluctuations. The approach operates by splitting the overall variability in the data into variability between groups and variability within groups.

At the basis of one-way ANOVA is the F-statistic is a ratio that compares the mean square between groups (which indicates the variability of group means around the overall mean) to the mean square within groups (which reflects the variability inherent in each group). The null hypothesis (H_0) maintains that all group means are equal, while the alternative hypothesis (H_1) holds that at least one group mean differs. This statistical test is built on important assumptions: independence of observations, near normality within each group, and homogeneity of variances among the groups.

Beyond its usual uses in domains such as psychology and medicine, one-way ANOVA has found a fresh role in EM-SCA. In this context, researchers employ the approach to examine whether the variations in EM emissions recorded during cryptographic operations are statistically significant. For example, Yang, Li, and Wang conducted one-way ANOVA to electromagnetic traces gathered from an AES implementation (Yang and Jia, 2021). Their study proved that tiny yet statistically significant changes in the EM leakage might be related to specific cryptographic procedures, hence disclosing possible weaknesses that attackers might exploit

3.6 Build the Machine Learning Model

The primary aim of this study is to demonstrate that ML can effectively map the patterns of EM radiation signals to known internal behaviors of IoT devices, yielding forensically significant insights. Rather than comparing the performance of multiple ML algorithms across various

scenarios, our goal is simply to confirm that ML methods once properly trained can accurately detect and classify relevant features in the EM traces. Considerations such as training efficiency or validation runtime, while important in production environments, are given secondary emphasis here. Consequently, any ML approach capable of meeting the accuracy requirements is deemed suitable.

In practice, the task of mapping EM signatures to specific internal device states or anomalies is a classification problem. This research employs a selection of neural network–based classifiers that offer substantial flexibility through adjustable hyper parameters. First, we use a multilayer perceptron (MLP) (Cheng and Titterington, 1994), the simplest form of neural network, in which data flows in a feedforward manner from the input layer through one or more hidden layers to the output layer. Additionally, long short-term memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) are applied due to their ability to handle entire sequences of data points particularly beneficial for identifying temporal patterns in time-series data such as EM traces (Gers et al., 2002). Lastly, we utilize SVMs for binary classification tasks, capitalizing on their effectiveness and relative simplicity when dealing with more straightforward classification scenarios (Ben-Hur and Weston, 2010).

3.6.1 Support Vector Machines (SVM)

SVM, offer versatility in handling both classification and regression tasks effectively. Particularly in regression, SVMs can thrive, especially when dealing with high-dimensional datasets and nonlinear connections between variables. The kernel trick, a crucial feature of SVMs, permits the transfer of input data into higher-dimensional spaces, where nonlinear relationships can be better captured through the creation of decision boundaries.

Furthermore, SVMs are resilient against overfitting, especially in high-dimensional spaces, making them excellent for evaluating EM trace data that may entail several factors(Cortes and Vapnik, 1995). Additionally, SVMs are known for their generalization capability, which is critical in cases when datasets are restricted or noisy (Cortes and Vapnik, 1995). However, despite these advantages, it's crucial to realize that SVMs, like multi output regression model, may provide unsatisfactory results in certain cases. For instance, if the relationship between inputs and outputs

in drone data is particularly complicated or non-linear, SVMs may struggle to capture it adequately. In such circumstances, deep learning algorithms might offer superior performance.

Equation (1) indicates accuracy is defined as the number of correctly detected traces, true positives (TP) and true negatives (TN), over the total number of traces, TP and TN as well as false positives (FP) and false negatives (FN). The total number of traces will equal the number of traces in the test group.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Equation (2) demonstrates accuracy is defined as the number of traces correctly classified as unauthenticated APIs, TP, over the total number of traces detected as unauthenticated APIs, TP plus FP.

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

Equation (3) demonstrates recall is defined as the number of traces correctly classified as unauthenticated APIs, TP, over the total number of traces that actually were unauthenticated APIs, TP and FN.

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

Precision and recall provide another approach of evaluating prediction performance. Low accuracy suggests too much FP, which creates extra labor for a defender. Low recall signals too many FN, which means the most threatening invoke unauthenticated API unnoticed. In the instance where both precision and recall are about the same value as accuracy, then accuracy successfully adjusts for those conditions.

3.6.2 Neural Network

Traditional machine learning algorithms have difficulty in finding complicated connections within EM signals due to the sophisticated nature of the data (Che and Zhang, 2020). However, neural network models offer a possible answer to this challenge. TensorFlow, an open-source deep learning framework built by Google, has emerged as a significant player in this arena since its creation in 2015 (TensorFlow, https://www.tensorflow.org/api_docs). Renowned for its thorough

documentation, robust training tools, and scalability choices, TensorFlow provides a diverse framework for developing and implementing machine learning applications. Moreover, TensorFlow's vibrant ecosystem, containing a variety of community-driven resources, libraries, and tools, further improves its appeal. Among these resources, Keras, a high-level neural network library developed on top of TensorFlow, stands out for its user-friendly interface and accelerated model construction process.

Keras simplifies the building of artificial neural networks, offering developers an easy Python-based API for constructing and training models. With its seamless connection with TensorFlow, Keras enables developers to use the potential of deep learning without the complexity commonly associated with older neural network frameworks. Together, Keras and TensorFlow provide a formidable toolkit for addressing the difficulties of drone data analysis. By utilizing the power of neural networks, developers may reveal insights from drone data that may have been challenging to extract using traditional machine learning methodologies.

The neural network architecture adopted in this model has various configurable parameters, each playing a significant role in influencing the model's behavior and performance. Below explains required parameters used in this study (Kadhim et al., 2022; Naila and Kong, 2018).

- **Number of Hidden Layers (layers):** This parameter influences the depth of the neural network architecture. More layers might potentially capture complicated correlations in the data but may also lead to overfitting if not correctly regularized.
- **Number of Neurons per Layer (neurons):** Each hidden layer consists of a variable number of neurons, which are computational units responsible for processing the input data and creating output. The number of neurons per layer influences the model's potential to learn and represent patterns in the data.
- **Dropout Rate (dropout_rate):** Dropout is a regularization technique used to prevent overfitting in neural networks. It randomly deactivates a fraction of neurons during training, forcing the network to acquire more robust properties and lowering reliance on specific neurons.
- **Learning Rate (learning_rate):** The learning rate specifies the step size at which the model's parameters are updated during training. It plays a significant function in determining the convergence speed and stability of the training process. A greater learning

rate may lead to faster convergence but risks overshooting the ideal solution, while a lower learning rate may require more training cycles but ensures smoother convergence.

- **Number of Epochs (epochs):** An epoch refers to a single run over the complete training dataset throughout the training phase. The number of epochs indicates how many times the model will loop over the dataset. Training for additional epochs allows the model to learn from the data several times, potentially boosting performance, but may also increase the danger of overfitting.
- **Batch Size (batch_size):** During training, the dataset is divided into smaller batches, and the model updates its parameters based on the average loss estimated over each batch. The batch size determines the number of samples processed in each training iteration. Larger batch sizes can expedite training by using parallel processing and optimizing memory use, but lower batch sizes may give superior generalization and convergence features.

Hyper parameter tuning is critical for optimizing machine learning models, and Randomized Search Cross-Validation (RandomizedSearchCV) is a valuable technique for this purpose. It randomly samples from a predetermined distribution of hyper parameters to discover the ideal combination. These hyper parameters include factors relating to the neural network's architecture (e.g., number of hidden layers, neurons per layer) and training parameters (e.g., dropout rate, learning rate, epochs, batch size). By carefully analyzing each combination's performance using a defined scoring metric, RandomizedSearchCV determines the configuration that produces the best results. This approach efficiently explores hyper parameter space, leading to a well-optimized model capable of obtaining improved accuracy and better generalization on unseen data.

3.7 Chapter Summary

This chapter detailed the methodology devised to investigate EM emissions from a NodeMCU microcontroller and determine whether firmware tampering or unauthorized API modifications can be detected and quantified. First, it outlined the experimental setup, including the use of a HackRF SDR as GNU radio companion and an H-loop antenna strategically positioned at chip of the MCU to capture EM signals. The software configuration on a host computer running Ubuntu ensured real-time data acquisition via GNU Radio Companion with various modifications to the APIs such as blocking random number generation, inserting PING calls, adding a data gathering endpoint,

and varying payload sizes. This create multiple scenarios reflecting both legitimate and potentially malicious firmware states. EM traces were then collected under each scenario at a 20 MHz sampling rate, capturing up to 20 seconds per trace.

To address timing inconsistencies, the methodology employed STFT for converting raw time-domain signals into the frequency domain and time domain, enabling a more consistent feature set. The chapter also discussed standard data preprocessing routines, including normalization and windowing, to facilitate machine learning based analysis of anomalies. Through this combined approach targeted firmware modifications, careful EM data collection, and subsequent frequency-domain feature extraction the methodology provides a robust framework for testing the hypotheses on detecting and measuring information leakage caused by unauthorized network API activities.

CHAPTER 4 - IMPLEMENTATION

This chapter walks the reader through every practical element needed to reproduce the study. Together, these sections provide a replicate-ready blueprint that complements the design rationale laid out in Methodology.

4.1 Hardware Setup

The acquisition of EM radiation involved the utilization of a HackRF One SDR Device in conjunction with a compact H-Loop magnetic antenna, characterized by a diameter of 15 mm (refer to Figure 27). Establishing connectivity between the SDR device and the H-loop antenna was facilitated through a semi-rigid RF cable, permitting precise placement of the antenna in a predefined location throughout the extended experimental duration (refer to Figure 26). Both the SDR device and the IoT device under scrutiny were seamlessly linked to the same host computer.

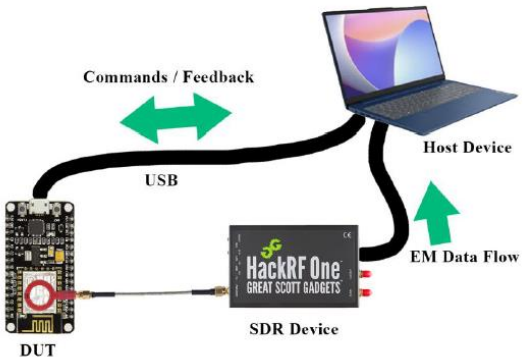


Figure 26: Proposed experimental setup for the EM data acquisition

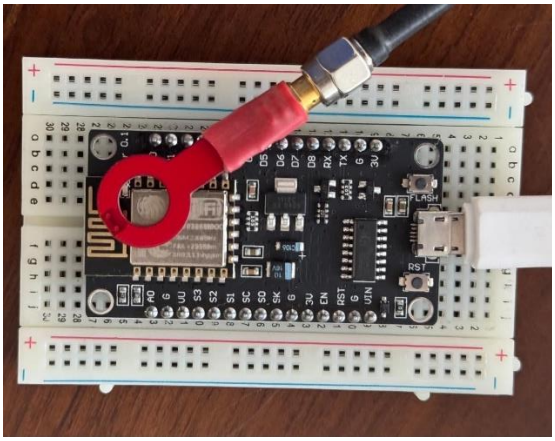


Figure 27: The arrangement of the H-Loop antenna on top coner of the NodeMCU

This setup was strategically designed to foster experimental control and efficient storage of the captured EM data. By connecting the SDR device and the IoT device to a common host computer, a synchronized environment was established for streamlined experimentation and data management. This synchronized configuration not only facilitated the coordination of experiments but also ensured the seamless integration of EM data from both devices. The use of the HackRF One SDR Device, coupled with the H-Loop magnetic antenna, provided a robust platform for capturing and analyzing EM radiation, contributing essential insights to the overall experiment.

4.2 Software Setup

In this GNU Radio flow graph (refer Figure 28), the Options block at the top level configures essential project parameters, including a title for the flow graph, the output language in this research Python is being used, and instructions on whether to generate code. A Variable block for example, samp_rate defines a single parameter for the sampling rate is 20 MS/s, making it easier to update in one place without manually reconfiguring multiple blocks.

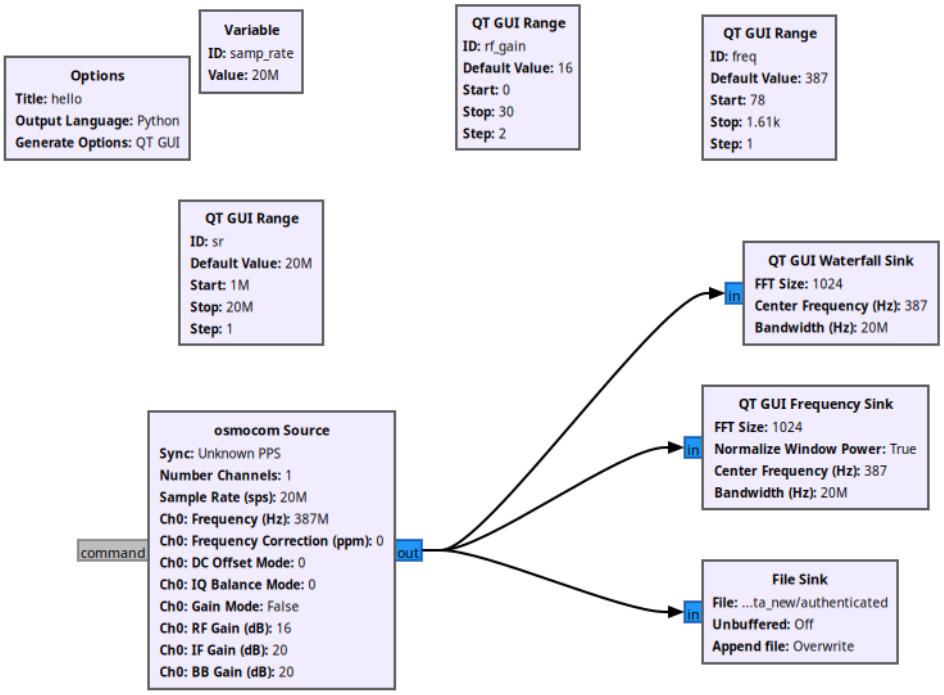


Figure 28: Setup of the GNU Radio Companion component to record and save EM data using HackRF

Several QT GUI Range blocks serve as interactive sliders to adjust parameters such as center frequency or gain on the fly. Each range specifies a default value along with minimum, maximum, and step increments. These blocks make it possible to change settings in real time during signal analysis. The Omnicom Source block interfaces with the SDR hardware, taking device-specific arguments to identify the target hardware which is HackRF and using the designated sample rate. It tunes to a specified center frequency and manages gain parameters (such as LNA and VGA gains), ultimately streaming complex I/Q data that represents the captured radio signals. Visualization of these signals occurs via the QT GUI Waterfall Sink and the QT GUI Frequency

Sink. The waterfall sink provides a time-versus-frequency view, where signal strength is illustrated by varying colors over time, while the frequency sink serves as a spectrum analyzer, displaying signal amplitude across the chosen bandwidth. Both sinks allow the user to see real-time changes in the RF environment as the system tunes different frequencies or alters gain.

Finally, the File Sink block writes the incoming I/Q samples to a file, either appending data or overwriting as needed. This is especially valuable for offline processing or machine learning experiments that require raw signal recordings. By tying these components together, particularly through the shared parameters from the variables and interactive sliders this flow graph achieves a flexible setup for both real-time analysis and data collection in an SDR based environment.

4.3 Explore the Leakage Frequency

During the capture of EM radiation using the Gqrx tool and GNU Radio, multiple frequencies were observed while adjusting the distance of the H-Loop antenna from the NodeMCU at a sampling rate of 20MHz After a thorough evaluation, it was determined that 390MHz exhibited the most significant leakage due to its higher signal strength. The positioning of the antenna relative to the chip is illustrated in Figure 27. This conclusion is reinforced by the observation that the 390MHz frequency appeared on the display when the H-Loop antenna was in close proximity to the chip and disappeared as the antenna was moved away. Furthermore, an additional significant frequency at 364MHz was identified while adjusting the H-Loop antenna's position, as shown in Figure 29.

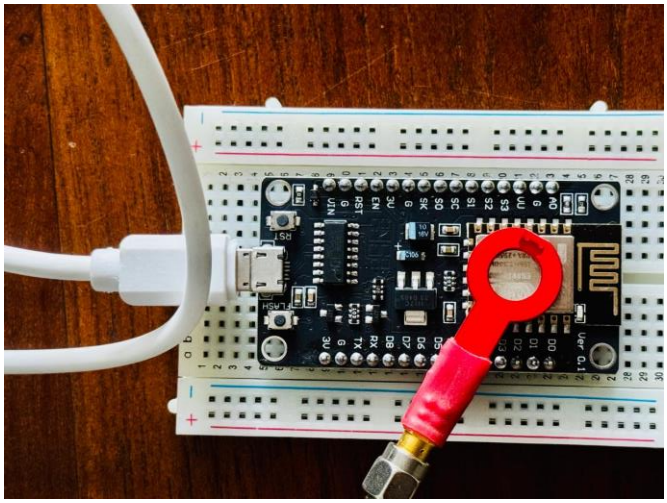


Figure 29: H-loop antenna set to chip's center position would result 364MHz

Notably, EM radiation from the corner positions of the chip was found to be more intense than that from the central positions. Therefore, the ***390MHz leakage frequency*** can be reasonably attributed to the NodeMCU for this research. The following images (see Figure 30, Figure 31, Figure 32, Figure 33) illustrate the Leakage frequencies that are capture using GNU radio companion software tool.

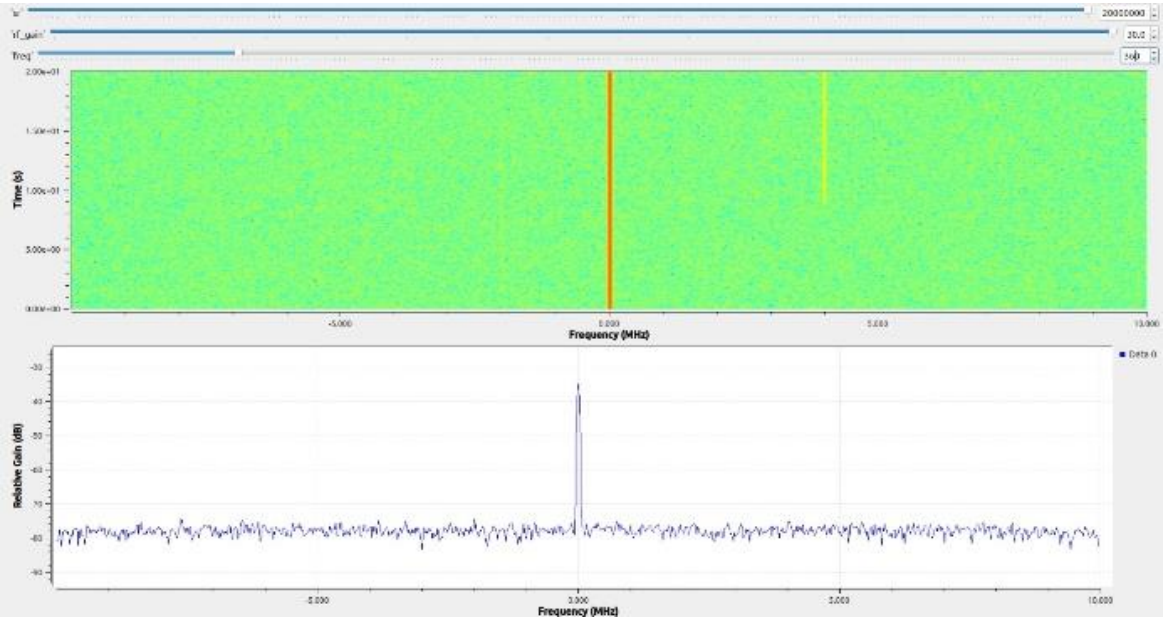


Figure 30: Output generated by the GNU Radio Companion EM radiation of NodeMCU at the leakage frequency of 364MHz. Center frequency is 360MHz when antenna is on center position without Node MCU is running

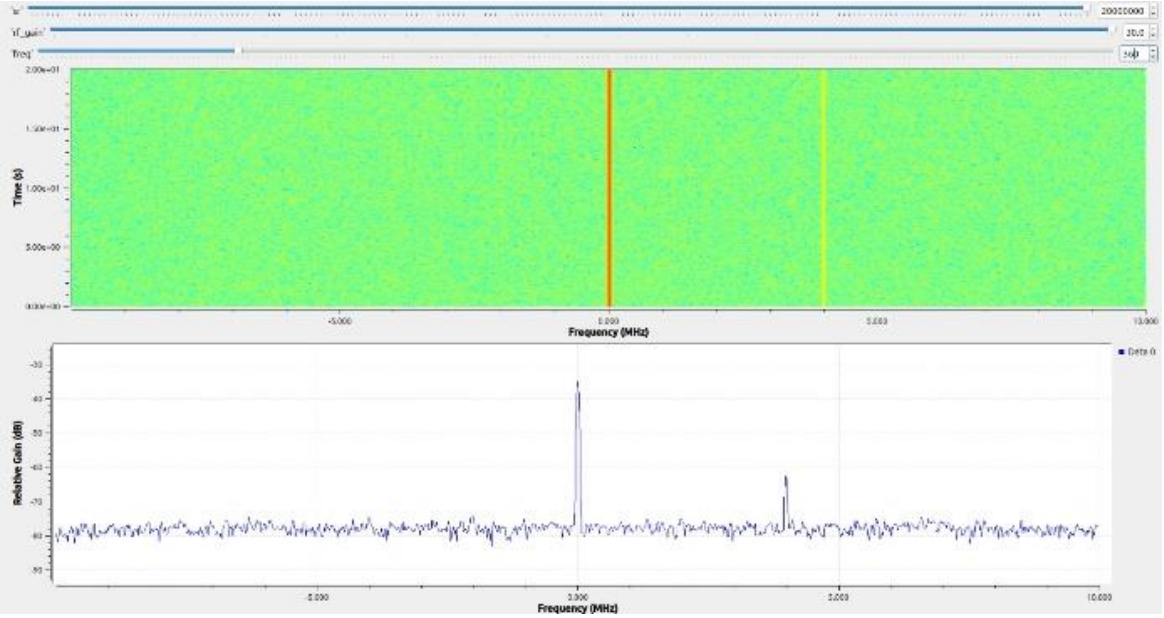


Figure 31: Output generated by the GNU Radio Companion EM radiation of NodeMCU at the leakage frequency of 364MHz. Center frequency is 360MHz when antenna is on center position with NodeMCU is running.

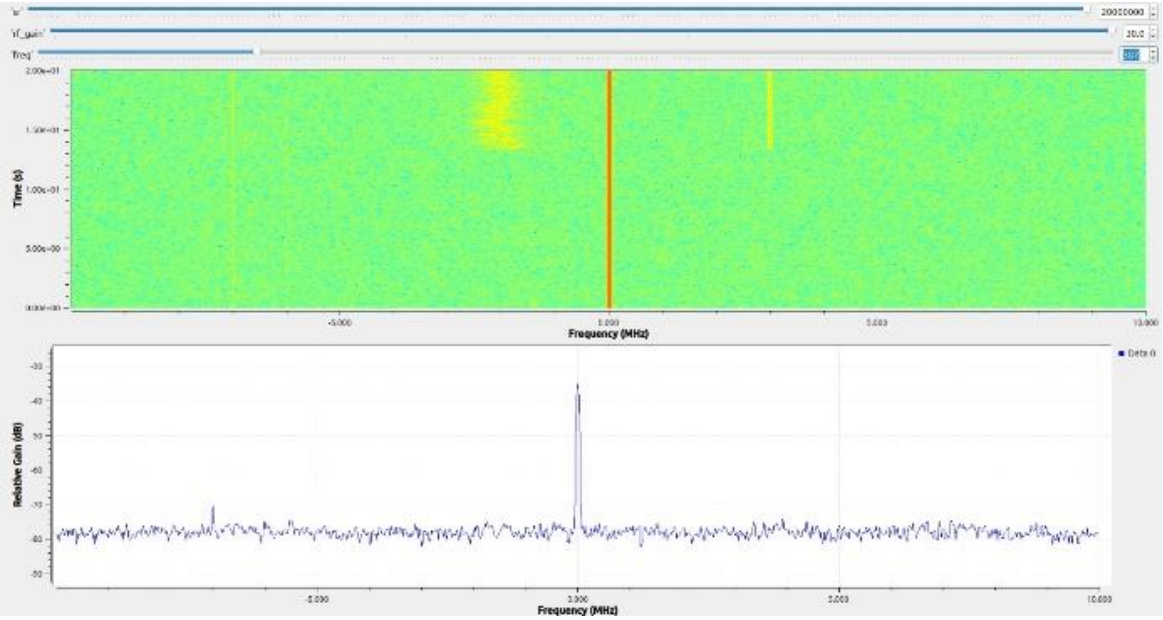


Figure 32: Output generated by the GNU Radio Companion EM radiation of NodeMCU at the leakage frequency of 390MHz. Center frequency is 387MHz when antenna is on corner position without Node MCU is running

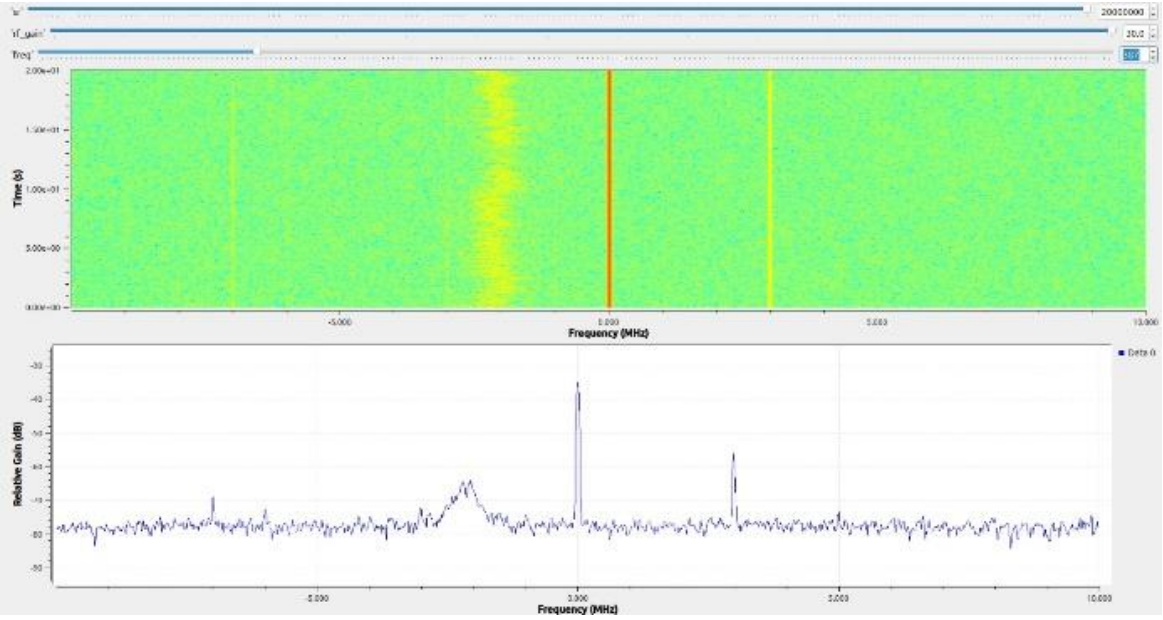


Figure 33: Output generated by the GNU Radio Companion EM radiation of NodeMCU at the leakage frequency of 390MHz. Center frequency is 387MHz when antenna is on corner position with NodeMCU is running.

A significant peak was observed at the center of the plots. The large spike is produced by the HackRF device and is referred to as a "DC offset" ("HackRF Great Scott Gadgets," 2025). The documentation for the HackRF One device advises users to disregard it as it is a common occurrence in every measurement.

4.4 Short-time Fourier transform (STFT)

EM traces were captured for 20 seconds and 60 seconds as explained in Data Collection only the first 10 seconds of each trace were used for further analysis due to hardware limitations. Following the STFT, the in-phase (I) and quadrature (Q) data stream is handled by a Python script employing a sliding-window approach, where each window spans 102.40 µs and advances in 89.6 µs increments (refer Figure 34). This technique preserves essential temporal context and enables the extraction of localized frequency features, paving the way for more robust statistical evaluations.

```
num_samp_per_class = 10000
fs = 20e6        # sampling frequency
fft_size = 2048  # FFT window size
fft_overlap = 256  # overlap between segments


original_half = getData_half("/media/rasi/Other/data/API-Security/hypothesis1/autheticated_firemware/original")
print(original_half.shape)

(212153772,)


f_original, t_original, Zxx_original = signal.stft(original_half, fs=20e6, nperseg=fft_size, noverlap=fft_overlap)
Zxx_original = Zxx_original.transpose()
```

Figure 34: Python code snippet for STFT to the captured EM signal with 20MHz sample rate, 2048 points FFT window 256 sample overlap

4.5 Hyper Parameter Tuning

RandomizedSearchCV was utilized to systematically explore a range of hyper parameters and identify the optimal configuration. The results of this search, including the best-performing parameter set, are summarized in Table 1.

Parameter	Value
Neurons	96
Activation	relu
# of hidden layers	2
Learning rate	0.003
Epoch	20
Batch size	64

Table 1: Hyper Parameter for NN model

Figure 35 presents a heat map depicting the mean cross-validation (CV) scores obtained by neural network architectures configured with varying numbers of hidden layers (y-axis) and neurons per layer (x-axis). The color scale represents the average performance across multiple folds, where lighter (yellow) regions indicate higher CV scores, while darker (blue) regions correspond to lower CV scores.

It (refer Figure 35) reveals that relatively simple architectures, particularly those with one or two hidden layers and a moderate number of neurons (e.g., 96 or 141 per layer), achieve consistently higher CV scores. In contrast, architectures with a greater number of hidden layers (four or five) or significantly larger neuron counts exhibit reduced CV scores, suggesting that increasing model complexity does not necessarily enhance generalization for this dataset. Notably, these more complex configurations appear more prone to overfitting, as evidenced by their lower average performance during cross-validation.

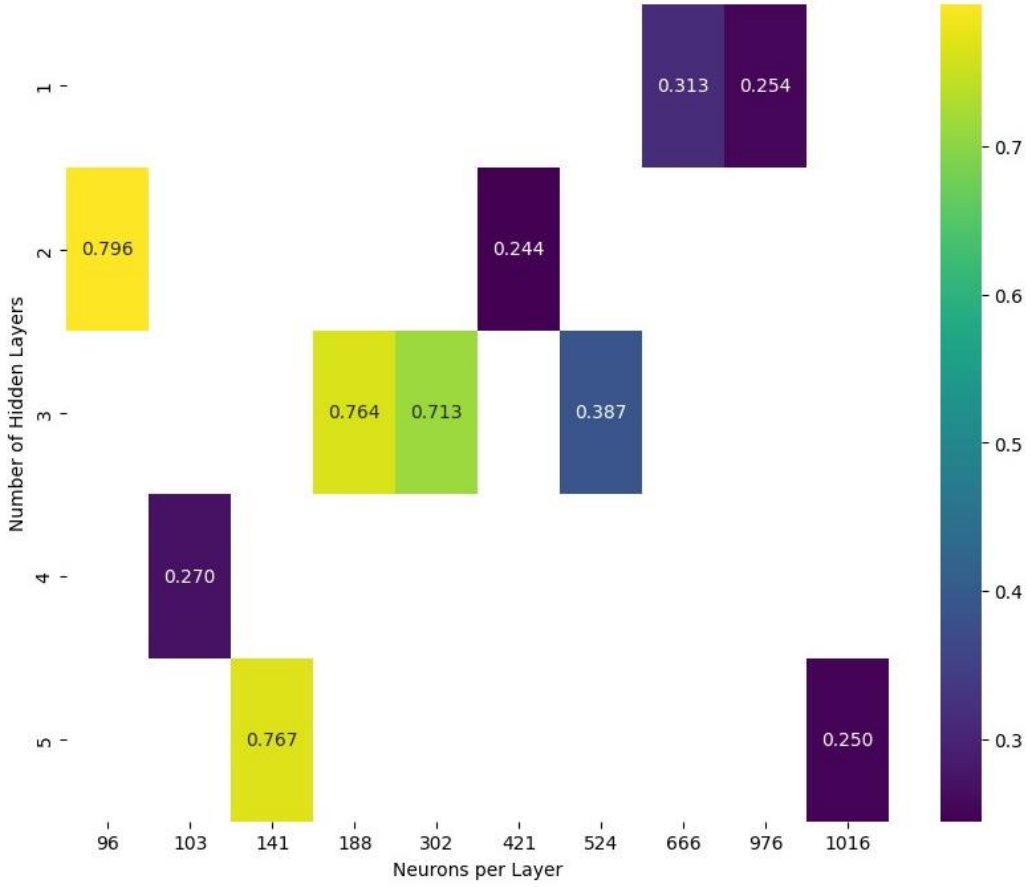


Figure 35: Mean cv score for combination of hidden layers and neurons

4.6 Chapter Summary

This chapter delineates the full, end-to-end realization of the study's design, converting the methodological blueprint into a reproducible working system. It first specifies the ESP8266 NodeMCU, fixed H-loop probe, HackRF One, and host workstation so that signal-capture conditions can be duplicated precisely. Then find the leakage frequency. It then details the GNU Radio flow-graph that streams 20 MS/s IQ data, Python orchestration scripts that align captures with firmware events, and PlatformIO based firmware builds that inject both baseline and tampered API sequences.

Together, these implementation details constitute the controlled environment and artefacts against which EVALUATION AND RESULTS is conducted; every metric reported there is derived directly from the hardware set-up, software set up. Founded leakage frequency and fine-tuned models established here.

CHAPTER 5 - EVALUATION AND RESULTS

5.1 Overview

This chapter presents the outcomes derived from the methodological steps detailed in Chapter 3, with an emphasis on distinguishing authenticated API traces from modified (unauthenticated) API traces. Initial analyses examine the mean EM signatures captured for both firmware configurations, offering a comparative baseline of normal versus tampered code behavior. Next, a statistical t-test is employed to determine whether variations in these EM traces particularly within the distribution of signal features are statistically significant indicators of firmware tampering or unauthorized API calls.

Further, the chapter investigates the efficacy of machine learning algorithms in classifying benign versus modified traces, thereby gauging how accurately an automated system can detect malicious interactions in real time. In addition to detection, ML techniques are employed to quantify the character-level leakage potentially occurring when malicious APIs exfiltrate sensitive data. Together, these empirical assessments demonstrate the robustness of the proposed EM-based detection method, highlighting how each analytical technique like statistical and machine learning can expose firmware compromises and protect against data leakage within an IoT ecosystem.

5.2 Summary of Dataset

5.2.1 Distribution of the Dataset

Raw time-domain signals (refer Figure 36), while encapsulating the overall waveform of EM emissions, do not inherently provide insights into the distribution of data points. The presence of multiple influencing factors, including noise, amplitude fluctuations, and transient device behaviors, results in a complex signal composition. As a consequence, identifying underlying patterns or statistically significant variations solely through time-domain observation becomes highly challenging (Waghmare and Megha, 2018).

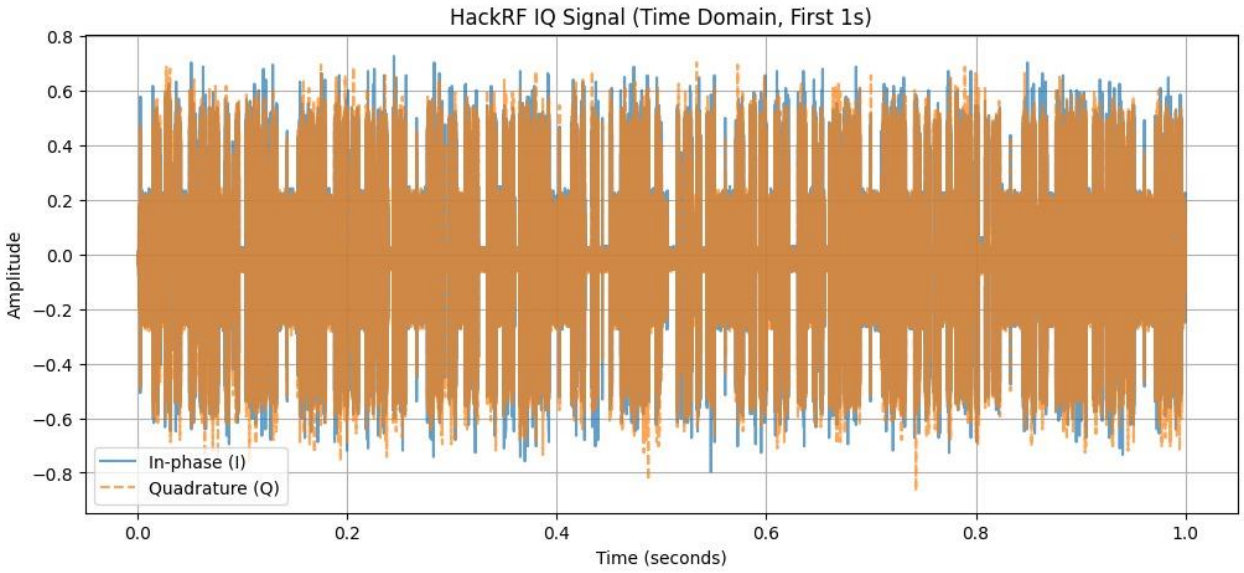


Figure 36: EM Trace in Time domain for first 1second representation

By contrast, converting the data into the frequency domain (using a Fourier transform) enables the isolation of specific frequency components, which can then be examined for their amplitude and phase characteristics. This process not only clarifies which frequencies carry the most informative content but also allows for a more accurate assessment of how the data are spread (Waghmare and Megha, 2018). The following illustration (Figure 37, Figure 38, Figure 39, Figure 40, Figure 41, Figure 42, Figure 43) are shown Power Spectral Density of the EM radiation emitted by the NodeMCU for different firmware modification related to API related activities. The distributions presented in the frequency domain exhibits a prominent central peak with symmetrically balanced tails on either side, characteristic of a bell-shaped curve commonly associated with a normal (Gaussian) distribution. The absence of significant skew where one tail would extend substantially farther than the other further reinforces this interpretation. Additionally, the gradual decline on both sides of the peak suggests a high degree of symmetry, which is a defining feature of normal distributions.

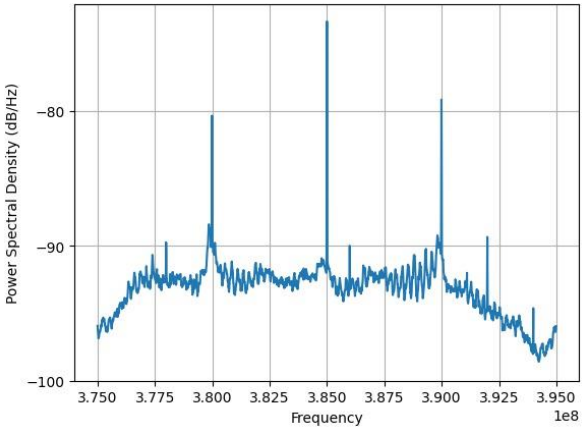


Figure 37: Authenticated APIs execution in frequency domain

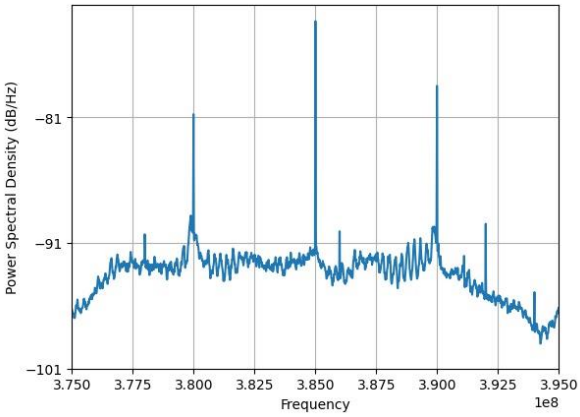


Figure 38: Remove Random generated number API execution in frequency domain

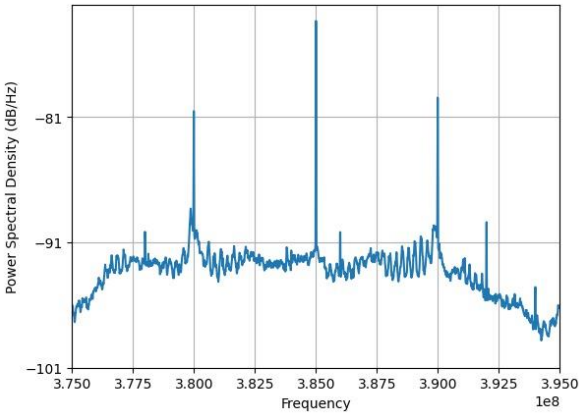


Figure 39: Ping First API execution in frequency domain

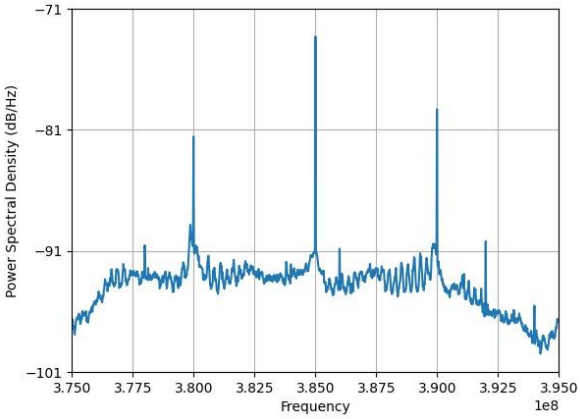


Figure 40: Ping middle API execution in frequency domain

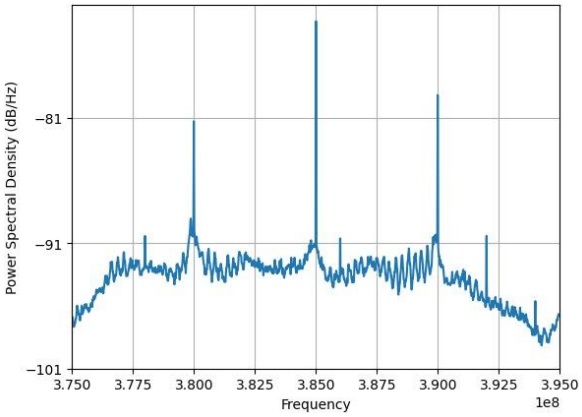


Figure 41: Ping Last API execution in frequency domain

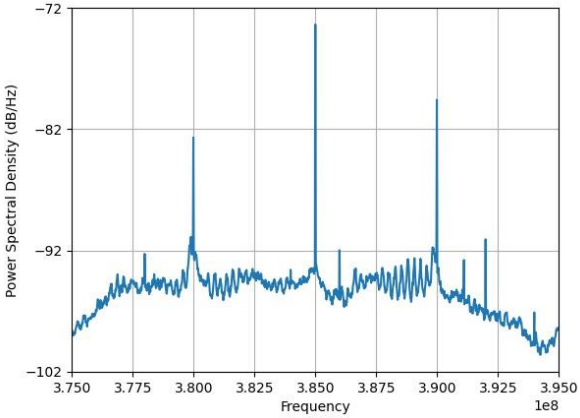


Figure 42: Info Gather API execution in frequency domain

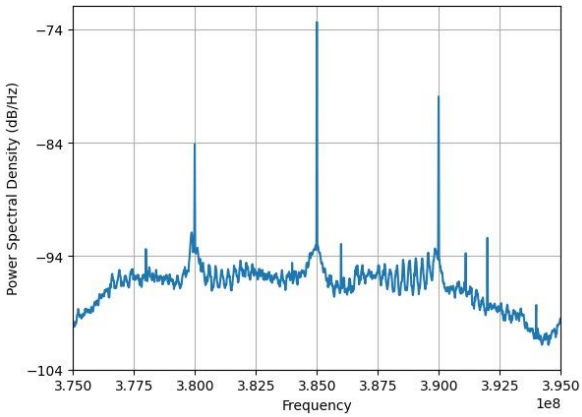


Figure 43: Change the payload of the weather API execution in frequency domain

5.3 Statistical Analysis

In order to perform meaningful statistical analysis on the EM signals, the raw time-domain data must first be converted to the frequency domain using a Short-time Fourier transform (STFT). This step reveals the underlying frequency components in localized time segments, which is crucial for spotting any patterns or irregularities that remain hidden in the raw signals.

5.3.1 T-Test Results

An initial t-test was conducted to investigate whether statistically significant differences exist in 95% confidential interval between authenticated and unauthenticated API calls. Each of the four unauthenticated APIs (discussed in the Modification for Authenticated Code Base) is individually compared to the baseline (i.e., authenticated) dataset, and the resulting t-statistics and p-values were recorded to assess whether disparities in electromagnetic (EM) signals were significant.

Unauthenticated API		**p-value**	**t-statistics**
Remove random generated number API		5.78×10^{-29}	56.15
PING API	Prior to the first request.	3.80×10^{-30}	99.73
	Midway through the authenticated calls	4.44×10^{-30}	150.23
	Immediately following the final request	2.47×10^{-30}	85.36
Data Gather API		7.90×10^{-42}	692.24
Update Weather API payload		6.11×10^{-37}	349.50

Table 2: P-value table of unauthenticated APIs vs Authenticate API for t-test experiment

Based on the results presented in the Table 2, all unauthenticated APIs exhibit exceptionally small p-values, well below conventional significance thresholds (e.g., $\alpha = 0.05$). For example, removing the random number from the API yields a p-value of 5.78×10^{-29}, while the PING API scenarios demonstrate similarly low values across the three measured timings (3.80×10^{-30}, 4.44×10^{-30}, and 2.47×10^{-30}). Likewise, the Data Gather API produces a p-value of 7.90×10^{-42}, and the Weather API payload update results in 6.11×10^{-37}, further highlighting the highly significant differences in electromagnetic emissions between unauthenticated and authenticated interactions.

Given this strong statistical significance, a machine learning model specifically, a SVM is subsequently employed to classify and assess the accuracy of distinguishing each unauthenticated API call from the legitimate API calls.

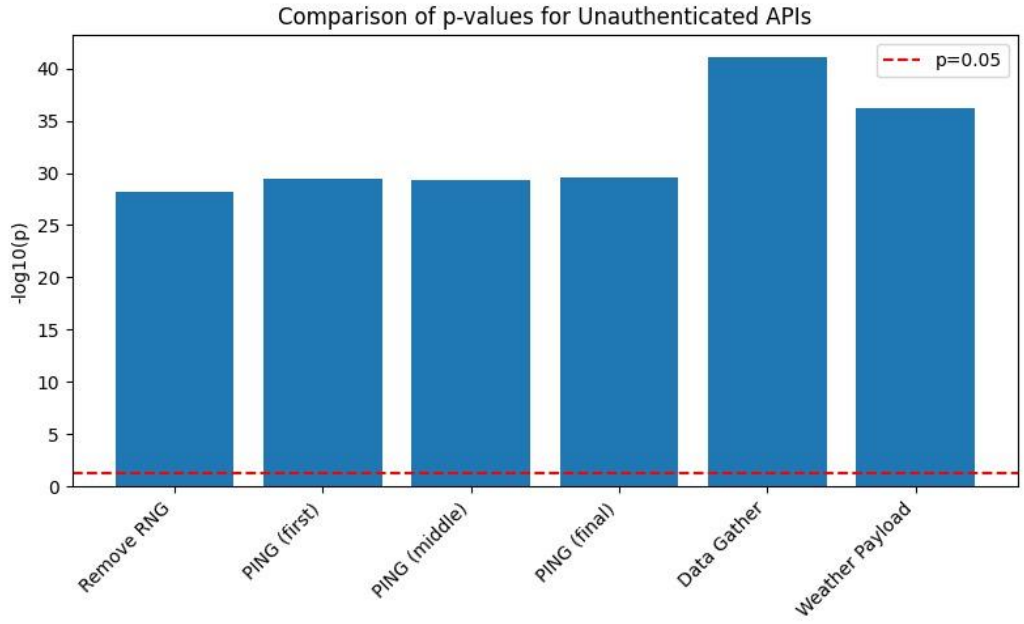


Figure 44: Bar chart illustrating −log(p) values for each unauthenticated API test, highlighting statistically significant differences from the authenticated baseline.

5.3.2 ANOVA Test Results

In addition to distinguishing between authenticated and unauthenticated API calls using EM traces, this research explores an additional experiment to determine whether changes in the execution

order of one API call can be detected through EM traces. Specifically, this investigation examines whether variations in the sequence of API invocations influence the EM emissions.

To evaluate this hypothesis, the PING API was executed in different positions within the sequence of API calls to assess its impact on EM traces. A one-way ANOVA test was employed for statistical analysis, as the experiment involved three distinct samples(Montgomery, n.d.)

- PING API Prior to the first request. (PING first)
- PING API Midway through the authenticated calls (PING middle)
- PING API Immediately following the final request (PING last)

This experimental setup aims to determine whether the order of API execution introduces measurable differences in EM emissions.

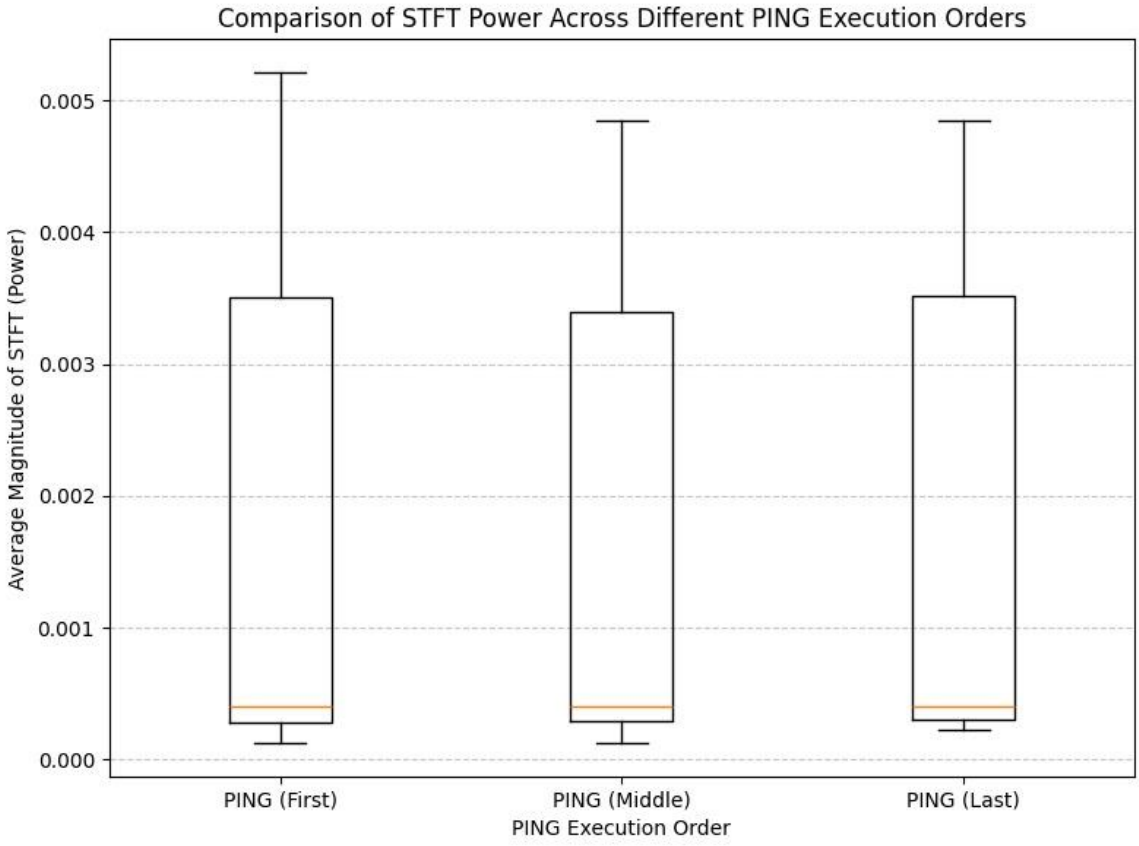


Figure 45: Box plot representing Average Magnitude of STFT for each PING API invoke position

This box plot illustrates the average STFT power measured across three distinct PING API execution orders: prior to the first request (PING first), midway through authenticated calls (PING middle), and immediately following the final request (PING last). Each box represents the distribution of power values within a given execution scenario, with the median indicated by the central horizontal line and the interquartile range (IQR) represented by the box itself. The whiskers extend to the lowest and highest values within 1.5 times the IQR, excluding outliers, thereby capturing the overall spread of the data.

The accompanying ANOVA results report an F-statistic of 2.6018 and a p-value of 0.0742. While the median power values appear relatively small, the moderate variability observed (as reflected in the box height and whisker range) suggests that PING API calls at different execution stages may induce distinguishable EM patterns. However, the p-value (~0.074) is slightly above the conventional significance threshold ($\alpha = 0.05$), indicating that the observed differences in STFT power are not statistically significant at the 5% level.

5.4 Support Vector Machine Model

To assess the statistical separability of unauthenticated API calls from legitimate ones, a t-test was performed, revealing a highly significant distinction. To further analyze classification performance, a SVM classifier was employed to evaluate the accuracy of differentiating unauthenticated API calls. Prior to training the SVM model, Linear Discriminant Analysis (LDA) was utilized to examine the linearity of the dataset. The results indicated that while authenticated and unauthenticated API calls exhibited some degree of linear separability, the data also contained non-linear characteristics. Therefore, to enhance classification performance, the Radial Basis Function (RBF) kernel was chosen for the SVM model to effectively model the underlying data structure.

Unauthenticated API / Statistics	Remove RGN API	PING API			Info Data API	Update Weather API payload
		First	Middle	Last		
Accuracy	0.91	0.89	0.81	0.98	0.85	0.98

Table 3: SVM result of authorized and unauthorized API for binary classification

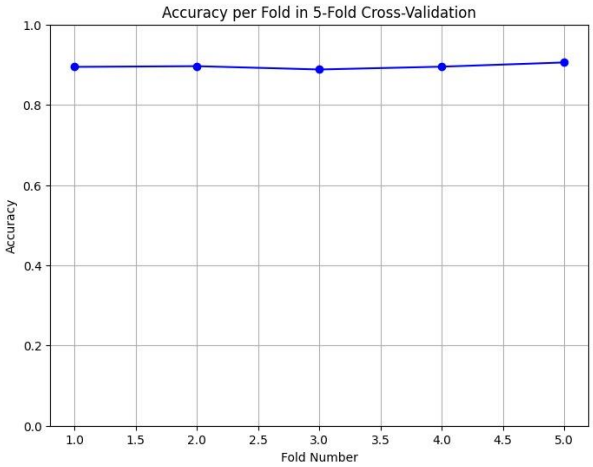


Figure 46: Mean accuracy of 0.8959 for authenticated API vs remove RGN API

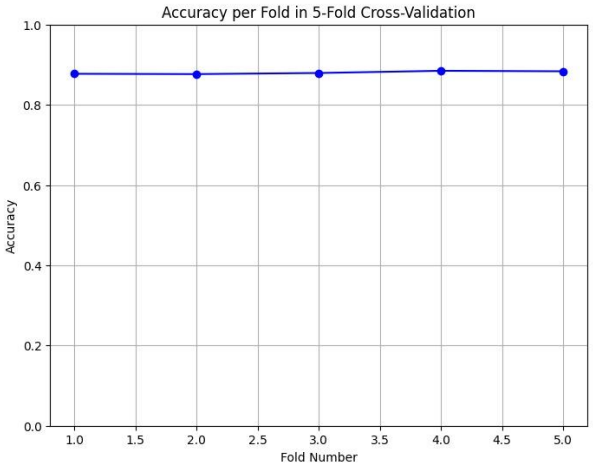


Figure 47: Mean accuracy of 0.8802 for authenticated API vs PING first API

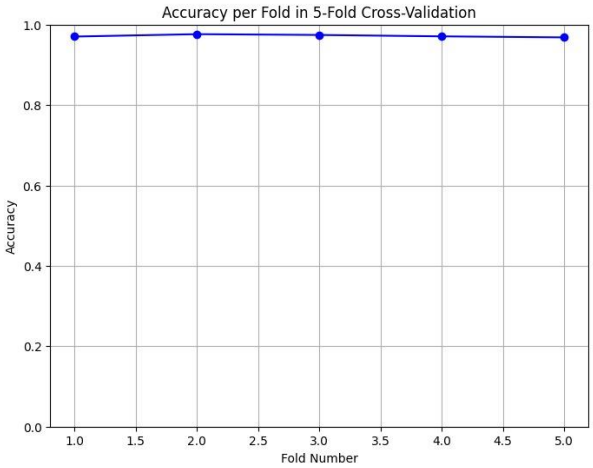


Figure 48: Mean accuracy of 0.9719 for authenticated API vs PING middle API

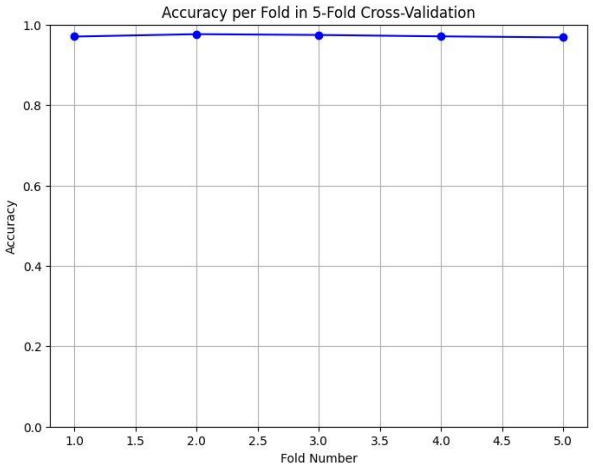


Figure 49: Mean accuracy of 0.9719 for authenticated API vs PING last

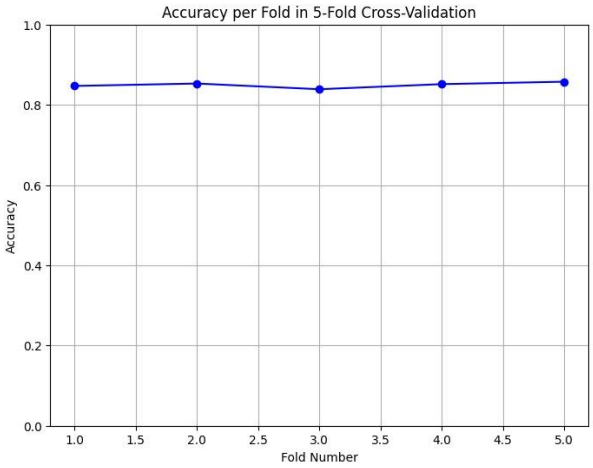


Figure 50: Mean accuracy of 0.8496 for authenticated API vs data gather API

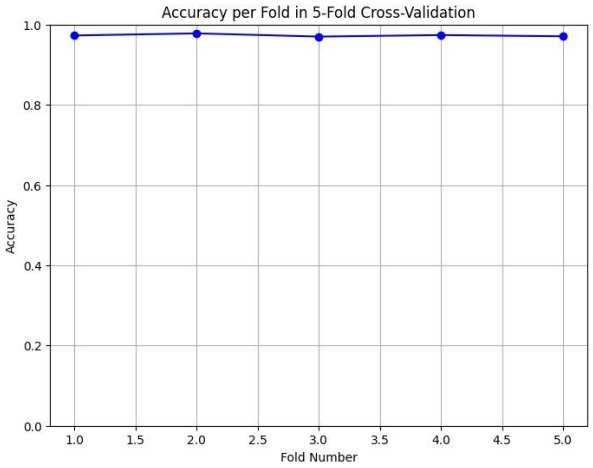


Figure 51: Mean accuracy of 0.9730 for authenticated API vs update payload of weather API

As summarized in Table 3, the model achieved consistently high accuracy, exceeding 81% in binary classification for each unauthenticated API against authenticated APIs. To ensure the robustness and generalizability of these results, a 5-fold cross-validation strategy was employed (refer Figure 46, Figure 47, Figure 48, Figure 49, Figure 50, Figure 51), systematically partitioning the dataset into five subsets to mitigate potential biases. The classifier's performance accuracy is illustrated in subsequent figures across all cross-validation folds. These findings reinforce the efficacy of SVM-based classification in distinguishing unauthorized API calls through their EM signal characteristics.

This research further aims to quantify the degree of information leakage at a per-character resolution. To achieve this, the payload size of the Weather API is systematically modulated across three distinct levels low, moderate, and high data leakage as outlined in the Systematic Variation of Weather API Payload Size section. The resulting EM traces are then analyzed using a Support SVM classifier to determine the classification accuracy across these payload variations. This approach enables a precise assessment of the granularity at which data exfiltration can be detected via EM-SCA, thereby substantiating the efficacy of side-channel analysis in identifying and quantifying unintended data exposure.

Information Leakage / Statistics	No information Leakage	Low information Leakage	Moderate information Leakage	High information Leakage
F1-score	0.83	0.77	0.79	0.75
Precision	0.80	0.81	0.77	0.76
Recall	0.85	0.74	0.80	0.74

Table 4: SVM classification results for Data leakage

The statistical metrics used to quantify information leakage are presented in Table 4, with an overall classification accuracy of 79% in distinguishing between leakage levels. To further ensure the robustness and reliability of these results, a 5-fold cross-validation procedure was conducted as

Figure 53. Which has the mean accuracy of 78.28%. Also confusion matrix represents for SVM classification as Figure 52.

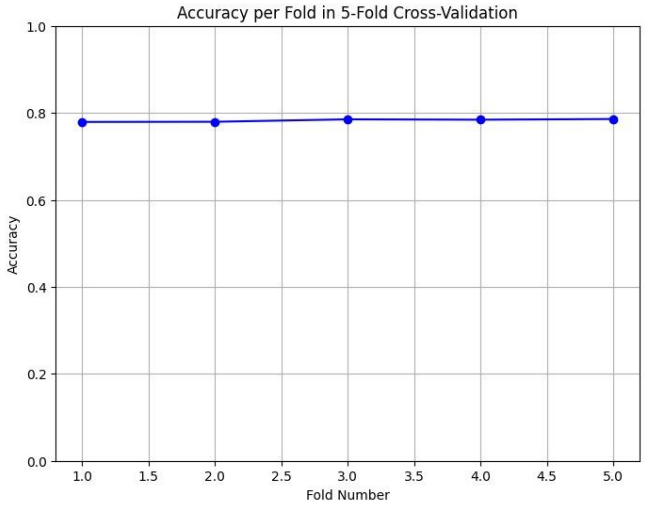


Figure 53: Cross validation for data leakage classification

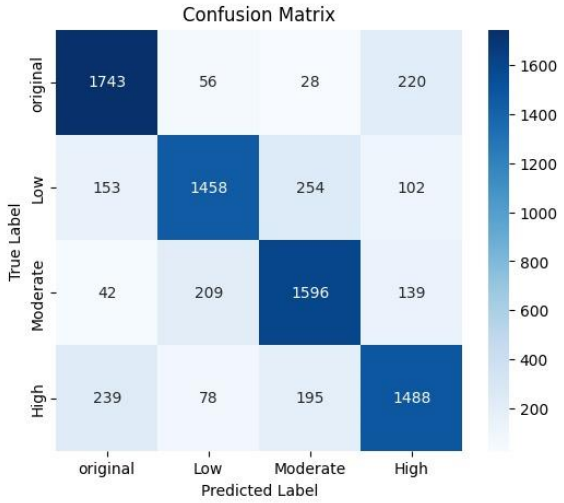


Figure 52: Confusion matrix of the SVM classification

5.5 Neural Network Model (NN model)

To enhance the accuracy of the information leakage model, a NN approach was employed. A crucial step in optimizing neural network performance is Hyper Parameter Tuning, which significantly influences model efficiency.

These findings highlight the critical role of model selection and hyper parameter tuning in neural network design. Specifically, smaller and less complex architectures may offer an optimal trade-off between representational capacity and generalization, ultimately leading to improved mean CV scores for this particular task.

The integration of these architectural optimizations and training methodologies yielded a neural network classifier that exhibited robust performance. As summarized in Table 5, the model achieved an accuracy of 82% in correctly categorizing information leakage levels, surpassing the classification accuracy of the SVM model. This result underscores the effectiveness of the neural network in capturing subtle patterns within the EM traces, demonstrating its capability for precise leakage detection.

Information Leakage / Statistics	No information Leakage	Low information Leakage	Moderate information Leakage	High information Leakage
F1-score	0.84	0.82	0.83	0.80
Precision	0.85	0.79	0.85	0.80
Recall	0.83	0.86	0.80	0.80

Table 5: NN model classification for information leakage statistics

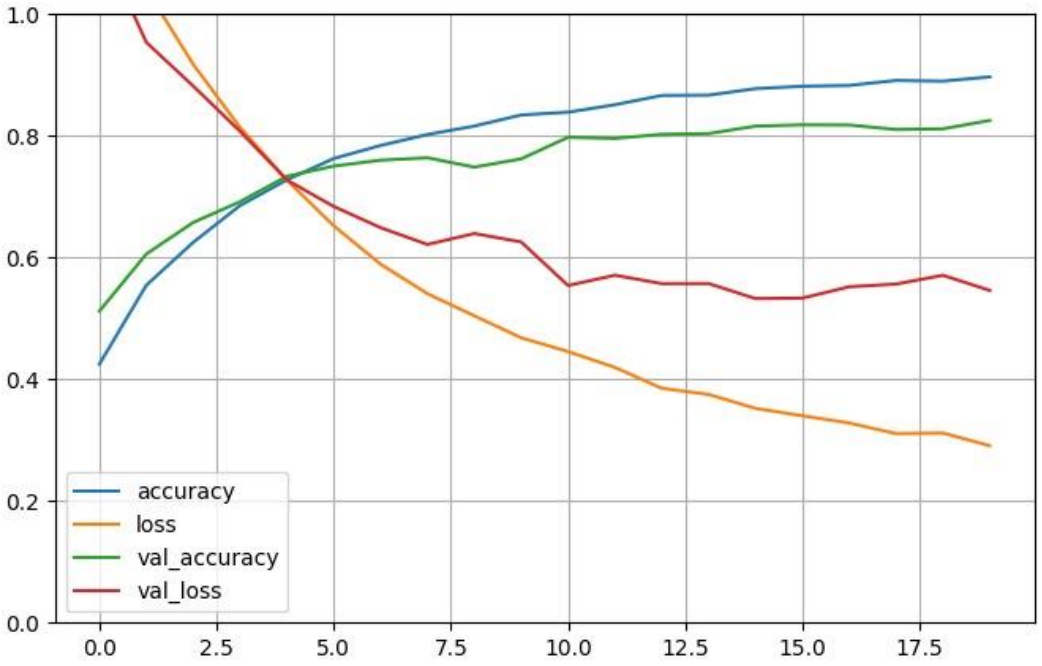


Figure 54:Training and Validation Loss Curves for implemented model

The training curves (refer Figure 54) demonstrate a stable learning process, as evidenced by the steady increase in both training and validation accuracy alongside a corresponding decrease in loss. Notably, the absence of divergence in validation loss indicates that the model does not exhibit obvious signs of overfitting. Further supporting these observations, the confusion matrix reveals that the majority of samples are correctly classified, as reflected by the concentration of values along the diagonal. However, some off-diagonal misclassifications particularly between the "Moderate" and "High" leakage classes suggest a degree of overlap or similarity in their feature distributions. This indicates that while the model generalizes well to unseen data, further

refinements, such as additional regularization or feature engineering, may enhance class separability.

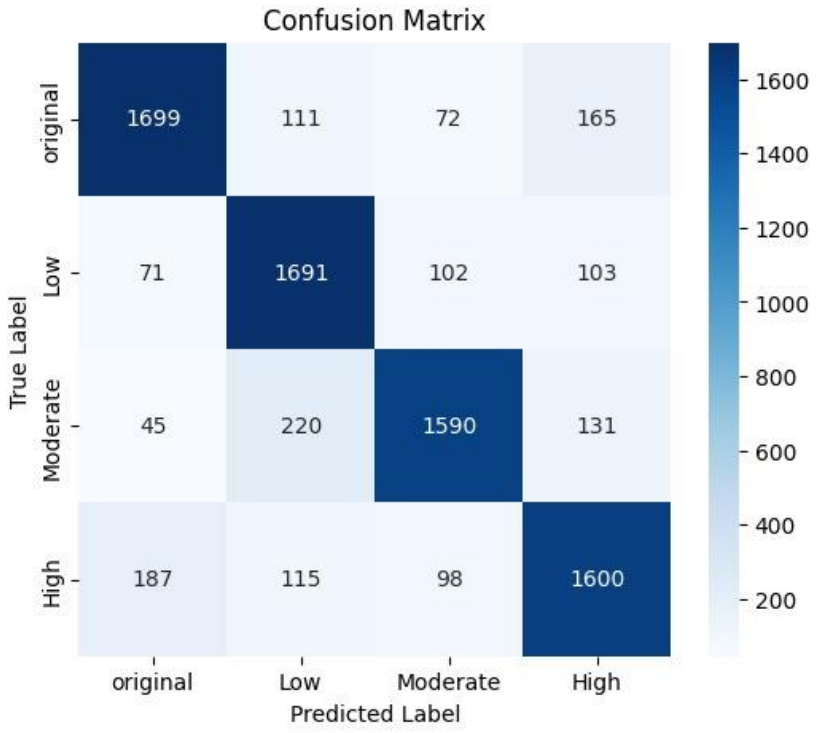


Figure 55: The confusion matrix of classifying information leakage using neural network model

Moreover, these findings underscore the practical applicability of the proposed model in real-world scenarios, particularly in digital forensic investigations involving NodeMCU devices. The model's ability to accurately distinguish subtle variations in electromagnetic traces highlights its potential for detecting and analyzing unauthorized data leakage for API related activities in embedded systems.

5.6 Chapter Summary

This chapter investigates EM-SCA data from NodeMCU devices to distinguish authenticated from unauthenticated firmware behaviors and to assess different levels of information leakage. Statistical tests revealed significant differences between legitimate and tampered API calls, demonstrating that changes such as the removal of random number generation or an increase in payload size result in identifiable EM signatures. An ANOVA test assessing the execution order

of a PING API call returned a p-value of 0.0742, suggesting that positional changes are observable but not statistically significant at the 95% confidence threshold.

Expanding on these findings, a binary SVM classifier achieved over 81% accuracy in identifying unauthorized API calls. For information leakage classification, the SVM attained a 79% success rate, with 5-fold cross-validation confirming its robustness. Further optimization using a neural network model, tuned via RandomizedSearchCV, improved classification accuracy to 82%. These results demonstrate the feasibility of EM-based methods for both detecting firmware manipulation and quantifying information leakage, reinforcing their potential application in IoT forensic analysis.

CHAPTER 6 - CONCLUSION AND FUTURE WORK

This chapter offers an overview of the research, a synthesis of the conclusions of the entire research work, the limitations of the research study, and an outline of further improvements.

6.1 Conclusion

The overall research outcomes strongly support the use of EM-SCA for both detecting firmware tampering and quantifying information leakage in IoT devices. Statistical tests, particularly the t-tests, produced extremely low p-values for unauthorized API modifications, which clearly indicate significant differences between the EM signatures of authenticated and tampered firmware. In addition, the classification experiments further corroborate these findings. Specifically, a SVM model achieved a classification accuracy of 79% in differentiating between various data leakage levels. In contrast, a neural network model, after hyper parameter tuning, reached a higher accuracy of 82%, demonstrating its superior ability to capture subtle distinctions in the electromagnetic traces.

Moreover, an ANOVA test was conducted to evaluate whether the order of API execution influenced EM emissions. Although the ANOVA yielded a p-value of 0.0742, which is marginally above the conventional 0.05 significance threshold, the results indicate observable variations in the emission patterns due to changes in API call sequence. This further emphasizes that while the positioning of certain API calls may not be statistically significant in isolation, their cumulative impact can still be instrumental in detecting unauthorized activities.

Overall, the study confirms that EM-SCA is a robust non-invasive tool for digital forensic analysis in embedded systems. The rejection of the null hypotheses regarding both the detection of firmware tampering and the quantification of information leakage illustrates that even subtle modifications in API interactions result in measurable differences in electromagnetic emissions. This research not only validates the practical application of EM-SCA but also highlights the advantages of advanced machine learning approaches particularly neural networks for accurately classifying security breaches in IoT devices.

6.2 Limitation

This study is not without its inherent limitations, and a primary constraint stems from the challenges associated with conducting data analysis on virtual machines. The restrictive nature of virtual environments, particularly in terms of available random access memory (RAM), poses a significant obstacle to the seamless execution of model fitting processes. The intricate computational demands of model fitting, exacerbated by the limitations of virtual machine configurations, hinder the comprehensive exploration of the data, and may lead to suboptimal results. Consequently, the findings derived from this study should be interpreted within the context of these computational constraints, acknowledging the potential impact on the overall analytical outcomes.

Another noteworthy limitation pertains to the physical characteristics of the NodeMCU chip, specifically its metal shield. The research primarily focused on the electromagnetic radiation emanating from the shield, recognizing it as a crucial aspect of the side-channel analysis. However, it is imperative to acknowledge that the presence of the metal shield introduces complexities in capturing and interpreting the full spectrum of electromagnetic signals. The shielding effect can potentially attenuate or modify the emitted radiation, influencing the accuracy and completeness of the forensic insights obtained. Consequently, the scope of this research is delimited by the challenges inherent in precisely characterizing the electromagnetic emissions from within the shielded environment of the NodeMCU chip.

6.3 Future Works

Future work in this area can build upon the promising results of this study by exploring several new directions. One potential avenue is the expansion of hardware and device testing. Future studies should investigate a broader range of IoT devices and microcontrollers beyond the NodeMCU to assess whether EM-SCA can reliably detect unauthorized API activities across diverse hardware architectures. In addition, employing more advanced antennas and higher-end SDRs may capture subtler or lower-level EM emissions, thereby enhancing the accuracy of anomaly detection.

Another critical area for future research is the improvement of signal processing techniques. While this study employed the STFT for time-frequency analysis, alternative methods such as wavelet

transforms could be explored to offer enhanced resolution or robustness in noisy environments. Additionally, the development of advanced noise reduction and filtering algorithms would help address the variability of environmental conditions, ensuring more consistent EM trace analysis in real-world scenarios.

Advancements in machine learning and statistical models also represent a promising direction. Future work could investigate the application of more complex deep learning architectures, such as convolutional neural networks combined with recurrent layers, to better capture the intricate temporal and spatial patterns inherent in EM emissions. There is also a need to develop real-time anomaly detection frameworks using online learning techniques, which would allow for proactive identification of firmware tampering as it occurs. Moreover, assessing the robustness of classification models against adversarial manipulations would further strengthen the detection system against deliberate attempts to obscure unauthorized API activities.

Integrating EM-SCA with broader security frameworks is another compelling direction. Future research should focus on the integration of EM-SCA techniques with existing digital forensic and intrusion detection systems to create a multi-layered defense strategy against IoT-based cyber threats. Automated countermeasures, such as systems that isolate compromised devices or alert network administrators when unauthorized API calls are detected, could provide real-time protection against emerging threats.

In addition to technical enhancements, future research should focus on expanding the dataset and standardizing benchmarking practices. Collecting larger and more diverse datasets that include longer-duration traces, varied operational conditions, and different types of firmware modifications would help refine machine learning models and improve their generalization across different environments. Moreover, establishing standard protocols and benchmarks for EM-SCA in IoT security would facilitate comparative evaluations of different methods and encourage community-wide advancements. Lastly, engaging with regulatory bodies to develop guidelines for EM leakage assessment could provide manufacturers and policymakers with a framework to enhance the overall security of IoT ecosystems.

APPENDICES

Appendix A

A.1 Electromagnetic Core Concepts

Maxwell's mathematical explanations for static electricity, magnetism, how currents in wires affect each other, and how currents are created in nearby wires. These laws imply that changing magnetic fields induce electric fields and vice versa, forming the basis for EM wave propagation exploited in side-channel attacks.

Gauss's Law $\nabla \cdot E = \varepsilon 0 \rho$

Gauss's Law represents electric fields (E) originate from electric charges, with the flux through a closed surface proportional to the enclosed charge.

Gauss's Law for Magnetism $\nabla \cdot B = 0$

Gauss's Law for Magnetism represents magnetic monopoles do not exist, and magnetic field lines (B) always form closed loops.

Faraday's Law of Induction $\nabla \times E = -\frac{\partial B}{\partial t}$

Faraday's Law of Induction represents a changing magnetic field induces a circulating electric field, forming the basis of electromagnetic induction

Ampère's Law (with Maxwell's Correction) $\nabla \times B = \mu_0 J + \mu_0 \varepsilon_0 \frac{\partial E}{\partial t}$

Ampère's Law represents the magnetic fields are generated by electric currents (J) and time-varying electric fields.

A.2 Hamming Theory

Hamming distance was first introduced in Information Theory. In this theory, hamming distance measures the number of different characters between two strings of the same length. In simpler terms, it tells us how many characters need to be changed to turn one string into the other. Typically, we achieve this by using XOR between each corresponding position in the two strings and counting the number of times we get a "1" to find the Hamming distance (Jiemin et al., 2014). The calculation formula of the Humming Distance Model is as follows.

$$C = \alpha . HW(D \oplus R) + b \tag{6}$$

We can use symbols to represent this relationship: C stands for electromagnetic radiation, D for the original data, R for the resulting data, HW for Hamming weight, b for noise, and $\oplus$ for the XOR operator. Then, $H\ (D \oplus R)$ represents the Hamming distance between D and R. The Hamming Weight Model is a specific version of the Hamming Distance Model, but in this case, the resulting data is set to zero (Jiemin et al., 2014).

A.3 Signal Theory

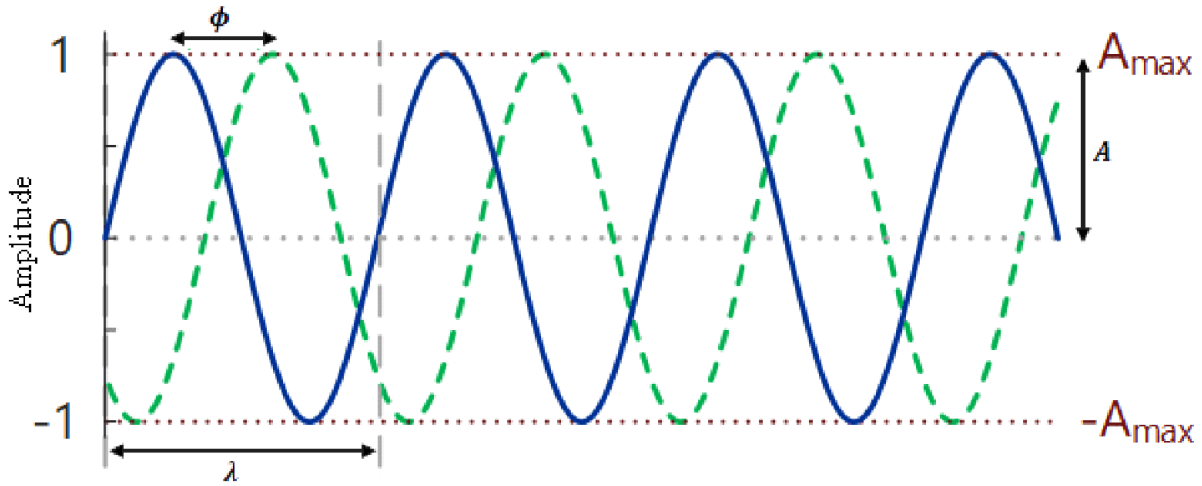


Figure 56: Properties of a periodic signal from (The Open University, n.d.)

Frequency (f), measured in Hertz (Hz), represents the count of recurring patterns within a one-second time span in a signal. Wavelength (λ), measured in meters (m), refers to the distance between two consecutive peaks in the signal. The value of the signal's dependent parameter at a specific moment in time is referred to as its amplitude (A), and this can be quantified in decibels

(dB). Phase (ϕ) represents the current displacement of the signal from a specified reference point. In a periodic signal with a frequency f= (1λ). Harmonics are generated as additional signals at positive integer multiples of f. These harmonics are labelled based on the integer multiple, such as the 1st harmonic (which is the fundamental frequency f), the 2nd harmonic (2f), the 3rd harmonic (3f), and so on (Smith, 1997).

A.4 Analog-to-Digital Conversion (ADC)

ADC involves two main steps they are sampling and quantization. In the field of signal processing, sampling refers to the process of converting a continuous-time signal into a discrete-time signal. When dealing with functions that change over time, consider a continuous function $S(t)$, often referred to as a "signal," that sample is needed. Sampling involves measuring the value of this continuous function at regular intervals of T seconds, known as the sampling interval or sampling period. The sampling frequency, denoted as fs, is determined by the number of samples taken in the time interval, and it can be calculated as fs= 1T, with the unit of measurement being samples per second, which is also commonly expressed as hertz. In digital signal processing, quantization refers to the procedure of transforming input values, often drawn from a large and sometimes continuous set, into output values from a smaller, countable set, typically composed of a finite number of elements. The distinction between an input value and its quantized counterpart, often resulting from rounding off, is known as quantization error.

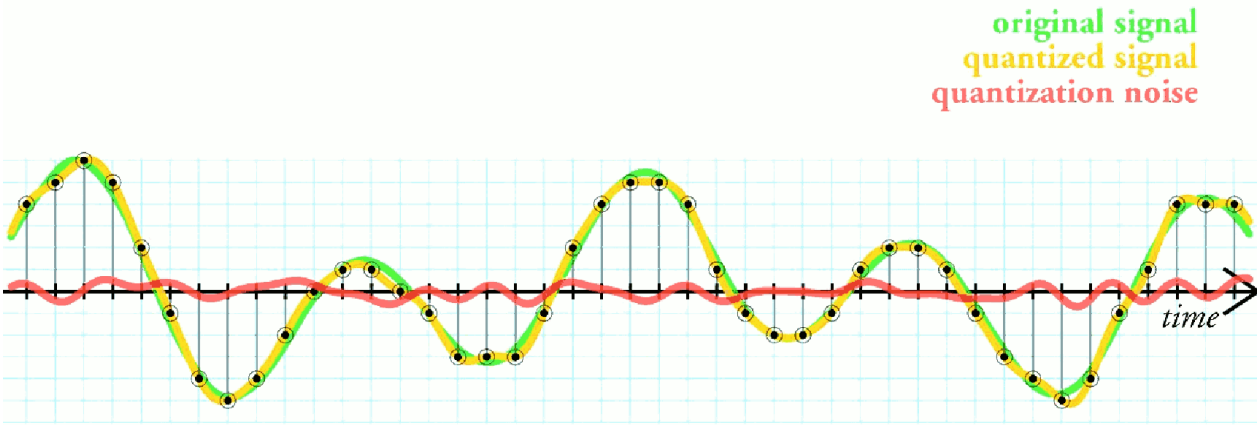


Figure 57: this illustration, represents the initial analog signal (depicted in green), the quantized signal represented by black dots, the signal reconstructed from the quantized data (shown in yellow), and the variance between the original signal and the reconstructed one (highlighted in red). This discrepancy between the original and reconstructed signals constitutes the quantization error. Adopted from ("Wikipedia," n.d.)

A.5 In-phase/Quadrature-phase (I/Q) Data

A sine wave is shown below, where 'A' stands for amplitude, 'f' for frequency, 't' for time, and 'ϕ' for phase. This equation states that the combination of amplitude, frequency, and phase is frequently employed to describe a signal

$$A \cos (2\pi f t + \phi)$$

However, amplitude (A) and phase (ϕ) alone can be sufficient to depict a signal's spontaneous state when needed, particularly when shown in a polar coordinate system. Using 'A' and 'ϕ' in a polar coordinate system, this depiction of a particular point in a signal may be converted into a Cartesian coordinate system, which uses coordinates on two axes to express the same information.

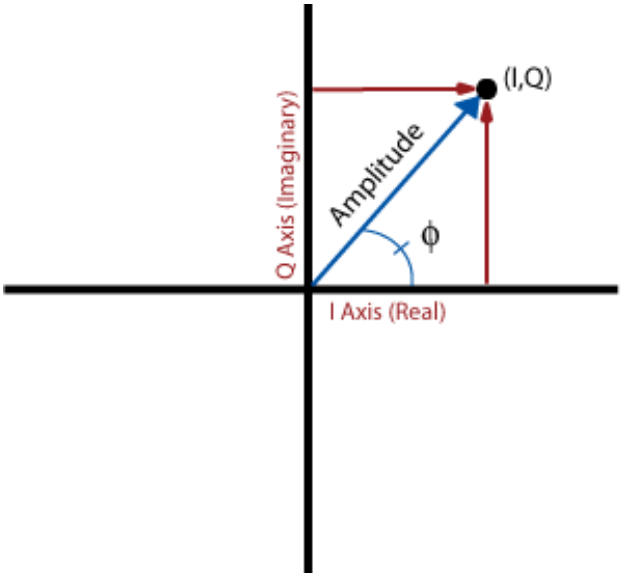


Figure 58: Cartesian coordinate representation of an SDR data sample. Adopted from ("I/Q Data for Dummies," n.d.)

The horizontal axis in this newly created Cartesian coordinate system is called the In-phase (I) axis, while the vertical axis is called the Quadrature-phase (Q) axis (refer to Figure 58). As a result, two coordinates (I/Q) may be used to represent a signal sample that the SDR has recorded, they are typically handled and stored as complex numbers. This complex number has an imaginary component that corresponds to the 'Q' coordinate and a real component that corresponds to the 'I' coordinate ("I/Q Data for Dummies," n.d.).

Appendix B

B.1 Resources

URL for Code

https://github.com/Rasika666/master_reseach

Diagrams

https://drive.google.com/drive/folders/1MECBw0NPQgtchGJt19jbp4mI2CzQe6sb?usp=sharing

REFERENCES

Adnan Khan, H., Alam, M., Zajic, A., 2018. Detailed tracking of program control flow using analog side-channel signals: a promise for iot malware detection and a threat for many cryptographic implementations. Presented at the International Society for Optics and Photonics, Cyber Sensing.

Aldowah, H., Rehman, S.U., Umar, I., 2019. Security in Internet of Things: Issues, Challenges, and Solutions. Adv. Intell. Syst. Comput. Switz. Springer 396–405. https://doi.org/10.1007/978-3-319-99007-1_38

Alieyan, K., Almomani, A., Abdullah, R., Almutairi, B., Alauthman, M., 2019. Botnet and Internet of Things (IoTs): A Definition, Taxonomy, Challenges, and Future Directions, in: Security, Privacy, and Forensics Issues in Big Data.

Amrouche, A., Boubchir, L., Yahiaoui, S., 2022. Side Channel Attack using Machine Learning. Presented at the 2022 Ninth International Conference on Software Defined Systems (SDS), IEEE. https://doi.org/10.1109/SDS57574.2022.10062906

Bari, N., Mani, G., Berkovich, S., 2013. Internet of Things as a Methodological Concept. Presented at the 2013 Fourth International Conference on Computing for Geospatial Research and Application, San Jose, CA, USA, 2013, IEEE.

Bartkewitz., T., 2016. Leakage prototype learning for profiled differential side-channel cryptanalysis. Presented at the IEEE Transactions on Computers, IEEE, p. 65:1761-1774.

Ben-Hur, A., Weston, J., 2010. A User's Guide to Support Vector Machines. Presented at the Data mining techniques for the life sciences, Springer, pp. 223–239.

Bhunia, S., Tehranipoor, M.M., 2018. Hardware Security: A Hands-on Learning Approach 1st Edition.

Biggs, S., Vidalis, S., 2009. Cloud Computing: The impact on digital forensic investigations. Presented at the Internet Technology and Secured Transactions, 2009. ICITST 2009, IEEE Xplore. https://doi.org/10.1109/ICITST.2009.5402561

Brier, E., Clavier, C., Olivier, F., 2004. Correlation power analysiswith a leakage model, in: Lecture Notes in Computer Science (LNCS). pp. 16–29.

Callan, R., Behrang, F., Zajic, A., Prvulovi, M., Orso, A., 2016. Zero-overhead profiling via EM emanations. Presented at the ISSTA 2016: Proceedings of the 25th International

Symposium on Software Testing and Analysis, pp. 401–412. https://doi.org/10.1145/2931037.293106

Che, Y., Zhang, X., 2020. On the Limitations of Traditional Machine Learning for Electromagnetic Side-Channel Analysis. Presented at the IEEE Transactions on Information Forensics and Security, IEEE, pp. 1298–1311.

Cheng, B., Titterington, D.M., 1994. Neural Networks: A Review from a Statistical Perspective. Inst. Math. Stat. 9, 2–30. https://doi.org/10.1214/ss/1177010638

Chevallier-Mames, B., Ciet, M., Joye, M., 2004. Low-cost solutions for preventing simple side-channel analysis: Side-channel atomicity. Presented at the IEEE Transac- tions on Computers, IEEE, p. 53:760-768.

Cortes, C., Vapnik, V., 1995. Support-vector networks 20, 273–297. https://doi.org/10.1007/BF00994018

Das, D., Boro, D., 2023. Electromagnetic Signal Analysis: A Vulnerability to Edge Computing Driven IoT Devices. Presented at the 2023 4th International Conference on Computing and Communication Systems (I3CS), IEEE, Shillong, India. https://doi.org/10.1109/I3CS58314.2023.10127273

Ding, A.A., Eisenbarth, T., 2016. Simpler, Faster, and More Robust T-Test Based Leakage Detection. Presented at the Constructive Side-Channel Analysis and Secure Design, pp. 163–183. EM Side Channels in Hardware Security: Attacks and Defenses, 2022. . IEEE Des. Test 39, 1–111. https://doi.org/10.1109/MDAT.2021.3135324

Espressif IOT Team, n.d. ESP8266EX Datasheet.

Farhan, L., Kharel, R., Kaiwartya, O., Quiroz-Castellanos, M., Alissa, A., Abdulsalam, M., 2018. A Concise Review on Internet of Things (IoT) -Problems, Challenges and Opportunities. Presented at the 2018 11th International Symposium on Communication Systems, Networks & Digital Signal Processing (CSNDSP), Budapest, Hungary, 2018, IEEE, pp. 1–6.

Garg, V.K., 2007. Fourth-Generation Systems and New Wireless Technologies, in: Wireless Communications & Networking.

Gartner, n.d. Gartner Research. IoT Secur. Primer Chall. Emerg. Pract. URL https://www.gartner.com/en/doc/iot-security-primer-challenges-and-emerging-practices

Gers, F.A., Eck, D., Schmidhuber, J., 2002. Applying LSTM to Time Series Predictable Through Time-Window Approaches. Presented at the Neural Nets WIRN Vietri-01, Springer, pp. 193–200.

Goodwill, G., Jun, B., Jaffe, J., Rohatgi, P., 2011. A testing methodology for side-channel resistance validation. Cryptography Research Inc.

Gupta, A.K., Johari, R., 2019. IOT based Electrical Device Surveillance and Control System. Presented at the 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), Ghaziabad, India, 2019, IEEE, pp. 1–5. https://doi.org/10.1109/IoT-SIU.2019.8777342

HackRF Great Scott Gadgets, 2025.

Han, Y., Etigowni, S., Liu, H., Zonouz, S., Petropulu, A., 2017. Watch me, but don't touch me! contactless control flow monitoring via electromagnetic emanations. Presented at the ACM Conference on Computer and Communications Security, Association for Computing Machinery.

Hanley, N., Tunstall, M., P. Marnane, W., 2009. Unknown plaintext template attacks, in: Lecture Notes in Computer Science (LNCS). pp. 148–162.

Hermann, M., Pentek, T., Otto, B., 2016. Design Principles for Industrie 4.0 Scenarios. Presented at the 2016 49th Hawaii International Conference on System Sciences (HICSS), IEEE.

Hochreiter, S., Schmidhuber, J., 1997. Long Short-term Memory. Neural Comput. 9, 1735–1780.

Husamuddin, M., Qayyum, M., 2017. Internet of Things: A study on security and privacy threats. Presented at the 2017 2nd International Conference on Anti-Cyber Crimes (ICACC), Abha, Saudi Arabia, 2017, IEEE, pp. 93–97. https://doi.org/10.1109/Anti-Cybercrime.2017.7905270

Ibrahim, O.A., Sciancalepore, S., Oligeri, G., Pietro, R.D., 2021. Fingerprinting usb flash drives via unintentional magnetic emissions. ACM Trans. Embed. Comput. Syst. 20.

I/Q Data for Dummies, n.d. . Whiteboard Web. URL http://whiteboard.ping.se/SDR/IQ

ISA, P., 2023. Fourth Industrial Revolution (4IR). URL https://www.shankariasparliament.com/current-affairs/fourth-industrial-revolution-4ir

Jiemin, Z., Jinming, L., Haimeng, S., Jian, M., 2014. Analysis of the Relationship between Hamming Distance and the Electromagnetic Information Leakage. TELKOMNIKA Indones. J. Electr. Eng. 12, 269–274. http://dx.doi.org/10.11591/telkomnika.v12i1.3966

Joshi, S.J., Mamaniya, S., Shah, R., 2022. Integration of Intelligent Manufacturing in Smart Factories as part of Industry 4.0 - A Review. Presented at the 2022 Sardar Patel International Conference on Industry 4.0 - Nascent Technologies and Sustainability for "Make in India" Initiative, IEEE, pp. 1–5. https://doi.org/10.1109/SPICON56577.2022.10180471

Kadhim, Z.S., Abdullah, H.S., Ghathwan, K., 2022. Artificial Neural Network Hyperparameters Optimization: A Survey. Int. J. Online Biomed. Eng. IJOE 15, 59–87. https://doi.org/10.3991/ijoe.v18i15.34399

Kagermann, H., Anderl, R., Gausemeier, J., Schuh, G., Wahlster, W., 2018. Industrie 4.0 in aGlobal Context Strategies for Cooperating with International Partners (Acatech Study). National Academy of Science and Engineering, München, Germany.

Kagermann, H., Wahlster, W., Lukas, W.-D., 2011. Industrie 4.0: Mit dem Internet der Dinge auf dem Weg zur 4. industriellen Revolution. VDI Nachrichten 13, 2.

Kalousis, A., Prados, J., Hilario, M., 2006. Stability of feature selection algorithms: a study on high-dimensional spaces. springer 12, 95–116. https://doi.org/10.1007/s10115-006-0040-8

Khan, A.H., Sehatbakhsh, N., Nguyen, L.N., Callan, R.L., Yeredor, A., Prvulovic, M., Zajic, A., 2021. Intrusion detection through electromagnetic-signal analysis for critical embedded and cyber-physical systems. Presented at the IEEE Transactions on Dependable and Secure Computing, IEEE, pp. 1150–1163.

Lakshminarasimhan, A., 2011. Electromagnetic Side-Channel Analysis for Hardware and Software Watermarking.

Le, Q., Miralles-Pechuan, L.M.-P., Sayakkara, A., Le-Khac, N.-A., 2021. Identifying Internet of Things software activities using deep learning-based electromagnetic side-channel analysis. Forensic Sci. Int. Digit. Investig. 39.

Lerman, L., Fernandes Medeiros, S., Meuter, cedric, Bontempi, G., Markowitch, O., 2013. Semi-supervised template attack. Presented at the International Workshop on Constructive Side-Channel Analysis and Secure Design, Springer, pp. 184–199.

Maghrebi, H., Portigliatti, T., Prouff, E., 2016. Breaking cryptographic implementations using deep learning techniques, in: Lecture Notes in Computer Science, Lecture Notes in Artitcial Intelligence and Lecture Notes in Bioinformatics. pp. 3–26.

Majed, H., Noura, H.N., Chehab, A., 2020. Overview of Digital Forensics and Anti-Forensics Techniques. Presented at the 2020 8th International Symposium on Digital Forensics and Security (ISDFS), IEEE, Beirut, Lebanon. https://doi.org/0.1109/ISDFS49300.2020.9116399

Martoyo, I., Setiasabda, P., Kanalebe, H.Y., Uranus, H.P., Pardede, M., 2020. Software Defined Radio for Education: Spectrum Analyzer, FM Receiver/Transmitter and GSM Sniffer with HackRF One. Presented at the 2018 2nd Borneo International Conference on Applied Mathematics

and Engineering (BICAME), IEEE, Balikpapan, Indonesia. https://doi.org/10.1109/BICAME45512.2018.1570509150

Maxwell, J.C., 1865. A dynamical theory of the electromagnetic field. R. Soc. 155, 459–512.

Mcnames, J., Aboy, M., Crespo, C., Brahm, G., 2002. Harmonic Spectrogram for the Analysis of Semi-periodic Physiologic Signals. IEEE. https://doi.org/10.1109/IEMBS.2002.1134427 ·

Montgomery, D.C., n.d. Design and Analysis of Experiments 8th Edition. John Wiley & Sons, Incorporated, 2012.

Mukhtar, N., Kong, Y., 2018. Hyper-parameter Optimization for Machine-Learning based Electromagnetic Side-Channel Analysis. Presented at the 2018 26th International Conference on Systems Engineering (ICSEng), IEEE. https://doi.org/10.1109/ICSENG.2018.8638234

Naila, M., Kong, Y., 2018. Hyper-parameter Optimization for Machine-Learning based Electromagnetic Side-Channel Analysis. Presented at the 2018 26th International Conference on Systems Engineering (ICSEng), IEEE, pp. 1–7.

Nesimoglu, T., 2010. A review of Software Defined Radio enabling technologies. Presented at the 2010 10th Mediterranean Microwave Symposium, IEEE, Guzelyurt, Northern Cyprus. https://doi.org/10.1109/MMW.2010.5605145

NodeMCU ESP8266 Detailed Review, n.d. URL https://www.make-it.ca/nodemcu-details-specifications/ (accessed 12.30.24).

ONIK, M.M.H., KIM, C.-S., YANG, J., 2019. Personal Data Privacy Challenges of the Fourth Industrial Revolution. Presented at the International Conference on Advanced Communications Technology(ICACT). https://doi.org/10.23919/ICACT.2019.8701932

P. Fournaris, A., Pocero Fraile, L., Koufopavlou, O., 2017. Exploiting hardware vulnerabilities to attack embedded system devices: A survey of potent microarchitectural attacks. Electron. Switz. 6.

Paul, K., Joshua, J., Benjamin, J., 1999. Differential Power Analysis. Presented at the Annual International Cryptology Conference, Advances in Cryptology — CRYPTO’ 99, pp. 388–397.

Poussier, R., Grosso, V., Standaert, F.-X., 2016. Comparing Approaches to Rank Estimation for Side-Channel Security Evaluations. Presented at the International Conference on Smart Card Research and Advanced Applications. https://doi.org/10.1007/978-3-319-31271-2_8

Rathour, N., Monika, Kumar, V., Kundu, S.S., Gehlot, Y., Gurung, A., 2023. Sigma Home: An IoT-Based Home Automation Using Node MCU. Presented at the 2023 2nd International Conference on Edge Computing and Applications (ICECAA), IEEE. https://doi.org/10.1109/ICECAA58104.2023.10212124

Robyns, P., Martino, M.D., Giese, D., Lamotte, W., Peter, P., Guevara, 2020. Practical Operation Extraction from Electromagnetic Leakage for Side-Channel Analysis and Reverse Engineering. Presented at the In 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks, Linz Austria, pp. 161–172.

S. Messerges, T., A. Dabbish, E., H. Sloan, R., 1999. Investigations of power analysis attacks on smartcards. Presented at the Proceedings of the 1st Workshop on Smartcard Technology, Smartcard.

Sayakkara, A., Le-Khac, N.-A., Mark, S., 2019. Leveraging Electromagnetic Side-Channel Analysis for the Investigation of IoT Devices. DFRWS 2019 USA 29, 94–103.

Sayakkara, A., Le-Khac, N.-A., Scanlon, M., Miralles-Pechuan, L., Luis, L., 2020. Identifying Internet of Things software activities using deep learningbased electromagnetic side-channel analysis. Forensic Sci. Int. Digit. Investig.

Sayakkara, A.P., Le-Khac, N.-A., 2021. Electromagnetic Side-Channel Analysis for IoT Forensics: Challenges, Framework, and Datasets. IEEE Access 9. https://doi.org/10.1109/ACCESS.2021.3104525

Sehatbakhsh, N., Alam, M., Nazari, A., Zajic, A., Prvulovic, M., 2018. Spectral analysis for anomaly detection on medical IoT and embedded devices. Presented at the IEEE international symposium on hardware oriented security and trust (, IEEE, pp. 1–8.

Sikder, A.K., Petracca, G., Aksu, H., Jaeger, T., Uluagac, S., 2018. A Survey on Sensor-Based Threats and Attacks to Smart Devices and Applications. IEEE Commun. Surv. Tutor. https://doi.org/10.1109/COMST.2021.3064507

Smith, S.W.S., 1997. The Scientist and Engineer's Guide to Digital Signal Processing. California Technical Publishing.

Socha, P., Brejnéík, J., Barték, M., 2018. Attacking aes implementations using correlation power analysis on zybo zynq-7000 soc board. Presented at the 7th Mediterranean Conference on Embedded Computing, MECO, pp. 1–4.

TensorFlow, https://www.tensorflow.org/api_docs. TensorFlow. API Doc.

The Open University, n.d. Signals and modulation. URL https://www.open.edu/openlearn/digital-computing/exploring-communications-technology/content-section-1

Tirumaladas, V., Axelsson, S., Dougherty, M., Ahsan Rasool, M., Hamdy, M., 2020. Deep learning-based Electromagnetic Side-Channel Analysis for the Investigation of IoT Devices. Presented at the Proceedings of the Second International Conference on Inventive Research in Computing Applications (ICIRCA-2020), IEEE, pp. 150–156. https://doi.org/10.1109/ICIRCA48905.2020.9182814

Tiwari, A., Mehrotra, V., Goel, S., Naman, K., Maurya, S., Agarwal, R., 2021. Developing Trends and Challenges of Digital Forensics. Presented at the 2021 5th International Conference on Information Systems and Computer Networks (ISCON), IEEE, Mathura, India. https://doi.org/10.1109/ISCON52037.2021.9702301

Verma, P., Bharot, N., 2023. A Review on Security Trends and Solutions Against Cyber Threats in Industry 4.0. Presented at the 2023 Third International Conference on Secure Cyber Computing and Communication (ICSCCC), IEEE. https://doi.org/10.1109/ICSCCC58608.2023.10176999

Vosoughi, A., Köse, S., 2019. Combined distinguishers to enhance the accuracy and success of side channel analysis. Presented at the IEEE International Symposium on Circuits and Systems, IEEE.

Waghmare, P.A., Megha, J.V., 2018. Efficient Pattern Recognition in Time Series Data. Presented at the 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), IEEE, Coimbatore, India, pp. 436–441.

Wanga, X., Zhoua, Q., Harera, J., Browna, G., Qiua, S., Doua, Z., Wangb, J., Hintonb, A., Aguayo Gonzalezb, C., China, P., 2018. Deep learning-based classifcation and anomaly detection of side-channel signals. Presented at the Cyber Sensing 2018.

What Is GNU Radio, n.d. URL https://wiki.gnuradio.org/index.php?title=What_Is_GNU_Radio

Wikipedia, n.d. . Quantization Signal Process. URL https://en.wikipedia.org/wiki/Quantization_(signal_processing)

Williams, P., Dutta, I.K., Daoud, H., Bayoumi, M., 2022. A survey on security in internet of things with a focus on the impact of emerging technologies. Internet Things 19.

Wim van, E.∗, 1985. Electromagnetic radiation from video display units: An eavesdropping risk? Comput. Secur. 269–286.

World Economic Forum, n.d. . Fourth Ind. Revolut. What It Means Respond. URL https://www.weforum.org/stories/2016/01/the-fourth-industrial-revolution-what-it-means-and-how-to-respond/

Xu, J., M. Heys, H., 2018. Template attacks of a masked s-box circuit: A comparison between static and dynamic power analyses. Presented at the 16th IEEE International New Circuits and Systems Conference, IEEE, NEWCAS, pp. 277–281.

Yang, W., Jia, A., 2021. Side-Channel Leakage Detection with One-Way Analysis of Variance. Security and Communication Networks.