Hybrid security solutions for IoT devices

Arié Haenel

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2024IPPAS022

Thèse de doctorat

Hybrid security solutions for IoT devices

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Telecom SudParis

École doctorale n°626 de l'Institut Polytechnique de Paris (IP Paris)
Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Palaiseau, le 6 Novembre 2024, par

Arié Haenel

Composition du Jury :

Sophie Chabridon Directrice d'études, Télécom SudParis, FRANCE	Présidente
David Starobinski Professeur, Boston University, ETATS-UNIS	Rapporteur
Valérie Viet Triem Tong Professeure, CentraleSupelec - Rennes Campus, FRANCE	Rapporteure
Bracha Shapira Professeure, Ben Gurion University, ISRAËL	Examinatrice
Vincent Nicomette Professeur, LAAS Toulouse, FRANCE	Examinateur
Amit Dvir Professeur associé, Ariel University, ISRAËL	Examinateur
Maryline Laurent Professeure, Télécom SudParis, FRANCE	Directrice de thèse
Yoram Haddad Professeur associé, HDR, Jerusalem College of Technology, ISRAËL	Co-directeur de thèse

Remerciements

הודו לה' כי טוב, כי לעולם חסדו.

Je tiens à exprimer ma plus profonde gratitude à mes directeurs de recherche, Yoram HADDAD et Maryline LAURENT, pour leurs conseils et leur soutien inestimables tout au long de mon parcours de recherche. Leur sagesse et leur patience ont largement contribué à mon développement académique et à la réalisation de cette thèse.

J'adresse mes remerciements à mes amis et collègues pour leur camaraderie et leur aide tout au long de ce parcours universitaire.

Je remercie également mes étudiants pour leur contribution à mon apprentissage. "De tous mes maîtres j'ai appris, et de mes élèves plus que de tous les autres." מכל מלמדי השכלתי ומתלמידי יותר מכולן (*Traité des Pères*, 4, 1)

Un immense merci à ma famille, qui m'a apporté un amour, un soutien et des encouragements sans fin. À ma mère, dont la force et la résilience ont été une source d'inspiration constante. À ma femme, Myriam, et à nos enfants, Elhanan, Yair, Avigail, Dvir et Ittiel, merci pour votre sacrifice, votre compréhension et pour la joie que vous apportez à ma vie chaque jour. Votre soutien a été la pierre angulaire de ma persévérance et de ma réussite. J'aimerais également rendre hommage à mon défunt père ז"ל, qui, bien qu'il ne soit pas là pour assister à l'aboutissement de ce travail, a été une lumière dans ma vie. Ses enseignements et ses valeurs continuent de vivre en moi.

Cette thèse n'est pas seulement le reflet de mes efforts, mais aussi le symbole du soutien et des encouragements collectifs de toutes les personnes mentionnées et non mentionnées qui ont fait partie de ce voyage.

Résumé

Il y a quelques années, un incident de cybersécurité aurait pu facilement être le scénario d'un film hollywoodien : des attaquants ont exploité un point d'entrée inattendu : un thermomètre intelligent dans l'aquarium du hall d'un casino. Cet objet connecté avait accès au réseau du casino, et les pirates ont réussi à le compromettre. Ils ont ensuite utilisé ce point d'entrée initial pour accéder à d'autres parties du réseau et ont finalement exfiltré 20 Go de données sensibles, y compris des informations sur les clients "gros joueurs".

Cet exploit audacieux met en évidence la vulnérabilité des appareils de l'Internet des Objets (IoT) et l'importance cruciale de leur sécurisation dans notre monde de plus en plus interconnecté. Dans le domaine en constante expansion de l'IoT, la sécurisation des réseaux de capteurs sans fil (WSN) présente un défi unique. Ces réseaux, composés de nœuds de capteurs à ressources limitées, jouent un rôle vital dans diverses applications, agissant comme des sentinelles de notre environnement physique. Cependant, leur puissance de traitement limitée et leur durée de vie de batterie les rendent vulnérables aux cyberattaques. Garantir l'intégrité et la sécurité des données collectées par les WSN est primordial.

Cette thèse aborde ce défi en adoptant une approche à deux volets, traitant de la sécurité à la fois du point de vue du réseau et du développement logiciel.

Le premier facette présente un schéma d'authentification léger et novateur, conçu pour répondre aux exigences strictes des RSN contraintes par la puissance. En surmontant la charge computationnelle des techniques cryptographiques traditionnelles, ce protocole innovant assure une communication sécurisée tout en minimisant la consommation de ressources, améliorant ainsi considérablement l'efficacité et la fiabilité des réseaux de capteurs sans fil. Le schéma proposé adopte une approche hybride, intégrant l'empreinte radiofréquence (RFF) avec des

méthodes cryptographiques légères pour fournir un cadre de sécurité multicouche. Cette double approche renforce non seulement le processus d'authentification en combinant les techniques de la couche physique et de la couche réseau, mais elle offre également une adaptabilité permettant des améliorations futures de la technologie RFF sans perturber l'architecture globale.

La contribution est validée à travers une série de méthodes d'évaluation. Une implémentation en temps réel démontre sa praticité dans des environnements WSN opérationnels, tandis que des tests de précision confirment la robustesse du protocole, atteignant une précision d'authentification supérieure à 99,8 % dans des scénarios expérimentaux. Les évaluations d'efficacité énergétique soulignent son adéquation aux dispositifs à ressources limitées grâce à une faible consommation d'énergie. Des analyses comparatives avec des protocoles cryptographiques de base et des méthodes RFF autonomes établissent la performance supérieure du schéma, équilibrant efficacement sécurité, efficacité et utilisation des ressources.

La deuxième facette introduit Shmulik, un système basé sur le Deep Learning conçu pour détecter des vulnérabilités logicielles, en particulier pour les systèmes embarqués et à ressources limitées. Contrairement aux outils traditionnels, Shmulik exploite une représentation intermédiaire (IR) du code pour améliorer la compatibilité et réduire la complexité de l'analyse. Il utilise une extraction GIMPLE personnalisée via un plug-in GCC adapté, une analyse de flux de contrôle basée sur des graphes à partir de points d'entrée stratégiques et un réseau neuronal Bidirectional Long Short-Term Memory (Bi-LSTM) pour le traitement des données séquentielles. Ces innovations permettent à Shmulik d'identifier les vulnérabilités en capturant des dépendances contextuelles complexes dans le code.

Shmulik est conçu pour être convivial et évolutif, avec une interface graphique intuitive qui relie les prédictions de vulnérabilités au code source, permettant aux développeurs de naviguer et d'interpréter les résultats de manière efficace.

L'efficacité de Shmulik est validée par des tests expérimentaux sur un ensemble de données dédié, une analyse comparative avec des outils de pointe, une étude de cas ciblée sur la bibliothèque libtiff, et la détection de vulnérabilités zero-day. En intégrant des techniques avancées d'apprentissage profond avec des considérations pratiques d'utilisabilité, Shmulik établit une nouvelle référence pour la détection automatisée des vulnérabilités dans les bases de code en

C, assurant une analyse de sécurité robuste et efficace.

La justification de cette double approche réside dans la recherche d'un cadre de sécurité complet. Un schéma d'authentification robuste sécurise la communication au sein du WSN, tandis que Shmulik protège l'ensemble du firmware des vulnérabilités. Les schémas d'authentification traditionnels seuls pourraient ne pas être suffisants si le logiciel lui-même présente des faiblesses exploitables. À l'inverse, l'efficacité de Shmulik repose sur les canaux de communication sécurisés qu'un schéma d'authentification léger peut fournir. En abordant la sécurité au niveau du réseau et du logiciel, nous visons à créer un système de défense plus résilient.

Cette thèse comble le fossé entre les solutions de sécurité légères pour les réseaux à ressources limitées et les techniques d'apprentissage profond de pointe pour l'analyse des vulnérabilités logicielles. En explorant ces deux pistes, nous nous efforçons de contribuer à un avenir plus sûr et plus fiable pour les WSN, améliorant ainsi la fiabilité et l'efficacité de cette technologie en évolution rapide.

Mots-clés: Réseaux de capteurs sans fil, authentification légère, appareils à faible consommation d'énergie, apprentissage profond, analyse des vulnérabilités logicielles, systèmes embarqués.

Abstract

A few years ago, a cybersecurity incident could easily have been the scenario of a Hollywood movie: attackers exploited an unexpected entry point: a smart thermometer in the casino's lobby fish tank. This Internet of Things (IoT) device had access to the casino's network, and the hackers managed to compromise it. They then used this initial foothold to access other parts of the network and ultimately exfiltrated 20GB of sensitive data, including high-roller customer information.

This audacious exploit highlights the vulnerability of IoT devices and the critical importance of securing them in our increasingly interconnected world. In the ever-expanding realm of the Internet of Things (IoT), securing Wireless Sensor Networks (WSNs) presents a unique challenge. These networks, composed of resource-constrained sensor nodes, play a vital role in various applications, acting as the sentinels of our physical environment. However, their limited processing power and battery life make them vulnerable to cyberattacks. Ensuring the integrity and security of the data collected by WSNs is paramount.

This thesis tackles this challenge with a two-pronged approach, addressing security from both the network and software development perspectives.

The first facet explores the development of a lightweight authentication scheme specifically designed for power-constrained WSNs. By overcoming the computational overhead of traditional cryptographic techniques, this innovative protocol ensures secure communication while minimizing resource consumption, thereby significantly enhancing the efficiency and reliability of wireless sensor networks. This contribution is validated through a range of evaluation methods, including real-time implementation, accuracy testing, and energy efficiency assessments. These methods demonstrate the effectiveness and practicality of the proposed scheme.

The first facet explores the development of a lightweight authentication scheme specifically designed for power-constrained WSNs. By overcoming the computational overhead of traditional cryptographic techniques, this innovative protocol ensures secure communication while minimizing resource consumption, significantly enhancing the efficiency and reliability of wireless sensor networks. The proposed scheme adopts a hybrid approach, integrating radio fingerprinting (RFF) with lightweight cryptographic methods to provide a multi-layered security framework. This dual approach not only strengthens the authentication process by combining physical-layer and network-layer techniques but also offers adaptability, allowing for enhancements in RFF technology without disrupting the overall architecture.

The contribution is validated through a range of evaluation methods. A real-time implementation demonstrates its practicality in live WSN environments, while accuracy testing confirms the protocol's robustness, achieving over 99.8% authentication accuracy in experimental scenarios. Energy efficiency assessments highlight its suitability for resource-constrained devices by maintaining low energy consumption levels. Comparative analyses against baseline cryptographic protocols and standalone RFF methods further establish the scheme's superior performance, showcasing its ability to balance security, efficiency, and resource usage effectively.

The second facet introduces Shmulik, a deep learning-based system crafted to unearth software vulnerabilities, especially for embedded and resource-constrained devices. Traditional methods of fortifying these systems often come at a cost, requiring increased memory, processing power, or dedicated hardware, further straining their limited resources. Shmulik offers a compelling alternative. By leveraging deep learning, we aim to develop a system that can automatically analyze code and pinpoint potential security weaknesses early in the development process. Unlike traditional tools, Shmulik leverages an intermediate representation (IR) of code to enhance compatibility and reduce analysis complexity. It employs customized GIMPLE extraction via a tailored GCC plug-in, graph-based control flow analysis rooted at strategic entry points, and a Bidirectional Long Short-Term Memory (Bi-LSTM) neural network for sequential data processing. These innovations enable Shmulik to identify vulnerabilities by capturing intricate contextual dependencies in code.

Shmulik is designed for usability and scalability, featuring an intuitive graphical interface

that links vulnerability predictions to source code, allowing developers to navigate and interpret results efficiently.

Its effectiveness is validated through experimental testing on a dedicated dataset, comparative analysis with state-of-the-art tools, a focused case study on the libtiff library, and the detection of zero-day vulnerabilities. By integrating advanced deep learning with practical usability considerations, Shmulik sets a new benchmark for automated vulnerability detection in C-based codebases, ensuring robust and efficient security analysis.

The rationale for this dual approach lies in the pursuit of a comprehensive security framework. A robust authentication scheme secures communication within the WSN, while Shmulik safeguards the whole firmware from vulnerabilities. Traditional authentication schemes alone might not be sufficient if the software itself has exploitable weaknesses. Conversely, Shmulik's effectiveness relies on the secure communication channels that a lightweight authentication scheme can provide. By addressing security at both network and software levels, we aim to create a more resilient defense system.

This thesis bridges the gap between lightweight security solutions for resource-constrained networks and cutting-edge deep learning techniques for software vulnerability analysis. By exploring both avenues, we strive to contribute to a more secure and reliable future for WSNs, ultimately enhancing the trustworthiness and effectiveness of this rapidly evolving technology.

Keywords: Wireless Sensor Networks (WSNs), lightweight authentication, power-constrained devices, deep learning, software vulnerability analysis, embedded systems.

Acronyms

AES	Advanced Encryption Standard
AI	Artificial Intelligence
AMI	Advanced Metering Infrastructure
ANN	Artificial Neural Network
API	Application Programming Interface
Bi-LSTM	Bidirectional Long Short-Term Memory
C&C	Command and Control
CLC	Certificateless Cryptography
CN	Central Node
CNN	Convolutional Neural Network
CPPS	Cyber-Physical Power System
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CVE	Common Vulnerabilities and Exposures
CWE	Common Weakness Enumeration
DAG	Directed Acyclic Graph
DDoS	Distributed Denial of Service
DLT	Distributed Ledger Technology
DTLS	Datagram Transport Layer Security
ECDH	Elliptic Curve Diffie–Hellman
ECDL	Elliptic Curve Discrete Logarithm
EER	Equal Error Rate
eSTREAM	ECRYPT Stream Cipher Project
FAR	False Accept Rate
FRR	False Rejection Rate
GUI	Graphical User Interface
GW	Gateway
ID	Identifier
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
IoT	Internet of Things
LLM	Large Language Model
LOS	Line-of-Sight
LSTM	Long Short-Term Memory
MAC	Media Access Control
MAM	Masked Authenticated Messaging
MITM	Man-in-the-Middle
ML	Machine Learning
NIST	National Institute of Standards and Technology
NLOS	Non-Line-of-Sight
NLP	Natural Language Processing
PCS	Path Changing Switch
PEOS	Payload Encryption-based Optimization Scheme
PKI	Public Key Infrastructures
PQC	Post-Quantum Cryptography
PSK	Pre-Shared Key
PUF	Physically Unclonable Function
PoS	Proof of Stake
QKD	Quantum Key Distribution
RAM	Random Access Memory
RFF	Radio Frequency Fingerprinting
ROI	Return On Investment
RSSI	Received Signal Strength Indicator
SN	Sensor Node
SNR	Signal-to-Noise Ratio
TESLA	Timed Efficient Stream Loss-tolerant Authentication
TLS	Transport Layer Security
URL	Uniform Resource Locator
VANET	Vehicular Ad-Hoc Network
VSS	Verifiable Secret Sharing
WSN	Wireless Sensor Network

Contents

List of Figures

List of Tables

LIST OF TABLES

Chapter 1

Introduction

Contents

The Internet of Things (IoT) refers to the network of physical devices, vehicles, buildings, and other items embedded with sensors, software, and connectivity, allowing them to interact and exchange data with other devices and systems over the Internet. Wireless Sensor Networks (WSNs) are a subset of IoT, comprising spatially distributed autonomous devices that use sensors to monitor and record physical or environmental conditions. WSNs are a crucial component of the IoT, enabling various applications such as environmental monitoring, healthcare, and industrial control. However, WSNs face significant security challenges due to their resource-constrained nature, dynamic topology, and large scale. Traditional security approaches are often inadequate for WSNs, and novel security solutions are required to address the unique security challenges in WSNs.

1.1 The Internet of Things and Wireless Sensor Networks: Challenges and Security Concerns

The IoT represents a revolutionary paradigm in which the Internet extends into the physical world, encompassing a wide array of devices and objects. These entities, equipped with sensors, actuators, and communication capabilities, collaborate to achieve complex tasks, leading to an interconnected ecosystem that blurs the line between the physical and digital realms.

The incredible potential and impact of the IoT is a recognized fact. Its rapid growth in recent years makes the task of securing it only more challenging. However, IoT devices are abused in all sort of ways. They may be used as entry points in systems, like the casino that was reported in 2017 as breached through its lobby-connected fish tank and saw its database hacked and 10GB of data stolen [2]. Or they are maliciously controlled for large-scale attacks, as in the case of the Mirai malware [3]: it created a huge botnet that was used to take down several web sites through Distributed Denial-of-Service (DDoS) attacks. It was able to accomplish that by taking control of hundreds of thousands of IoT devices, especially IP Cameras. More frightening are probably the cyber-physical systems attacks (successful or not) on critical infrastructure lifeline sectors, e.g., water [4], healthcare [5], energy [6].

Several factors make IoT devices the preferred target for attackers, to cite a few [7, 8, 9, 10]. Many of them are cheap devices made by manufacturers who recently added connectivity to their products and have no experience in security; IoT equipment may be designed to stay in the field without a way to provide software or firmware updates, even if some vulnerability has been disclosed, and still, they are connected to the network; some are low-powered, low-resources, low-cost devices and do not implement modern Information Security (InfoSec) methodologies and techniques; hundreds of different platforms and manufacturers lead to a very fragmented market and complicate the design and development of security solutions; a lack of standardization hinders the ability to secure connected devices from different manufacturers.

Wireless Sensor Networks (WSNs), a critical component of the IoT, consist of spatially distributed autonomous sensors that monitor physical or environmental conditions. These networks collect data from the environment, process it, and transmit the insights to a central

location for further analysis and decision-making.

The importance of WSNs cannot be overstated, as they find applications across various domains, including but not limited to environmental monitoring, smart cities, healthcare, agriculture, and industrial automation. By leveraging WSNs, we can gain real-time insights, optimize processes, and improve quality of life and operational efficiency.

As we delve deeper into the IoT and WSNs, it becomes evident that these technologies are not just a convenience but a necessity for the advancement of modern society. This necessity, however, also brings a need to address the security threats associated with WSNs.

WSNs are inherently vulnerable to a myriad of security threats due to their distributed nature and the often unattended environments in which they operate. The security challenges in WSNs are multifaceted, encompassing the risk of data interception, unauthorized access, and the manipulation of sensor data.

For a tangible illustration of WSN applications, let's consider agriculture technology (agritech) as a prime example. In the last few years, agritech has seen some of the highest IoT adoption levels in comparison with energy, mining and transport [11]. The Office of Homeland Security published an assessment that precision agriculture technology is increasing cyber targeting against the Food and Agriculture (FA) sector, and advised the farming industry to increase awareness, protect their data, and follow some mitigation measures [12]. In domains like agritech, IoT nodes are often low-power, low-cost sensors and actuators. The efficiency factor is of great importance, because they can be deployed for very long periods, with technologies such as LPWAN providing long-range communications up to 40 km (with future expectations up to 1000 km) and more than a 10-year battery life [13]. However, low cost and low power are two limiting factors in the ability to create secure systems.

Traditional security mechanisms, designed for systems with abundant computational resources, are ill-suited for WSNs. These networks are characterized by their limited processing power, energy constraints, and minimal storage capacity. As a result, the computational overhead associated with conventional cryptographic methods can be prohibitive, leading to a need for security approaches that are tailored to the resource-constrained nature of WSNs.

The unique vulnerabilities and resource limitations of WSNs necessitate the development of

lightweight, efficient and robust security solutions. These solutions must ensure the confidentiality, integrity and availability of data while also taking into account the energy consumption and processing capabilities of the sensor nodes.

1.2 Government intervention in IoT security and the challenges of compliance

For a long time, governments refrained from involving themselves in IoT security policies and enforcement measures. However, it has become clear that this approach must change. Initially, governments issued security recommendations and codes of best practices. So far, these are largely voluntary, meaning non-compliance does not result in sanctions. This is the case in the United States, several European countries, Australia, Singapore, and others [14, 15, 16, 17, 18].

Compliance remains a significant challenge, as evident from a 2021 research study conducted by the Australian Department of Home Affairs. The study aimed to evaluate the effectiveness of their *"Code of Practice, Security of the IoT for Consumers"*[17]. The research revealed that manufacturers found the principle-based guidance, which was too high-level and lacked specific implementation details, difficult to translate into concrete actions. Moreover, they expressed a preference for internationally aligned standards, which would allow them to implement security measures that are widely recognized and beneficial across the industry. Although major manufacturers expressed good intentions, many failed to even implement high-priority, low-cost recommendations [19].

Therefore, given the limitations of voluntary guidance, governments are intervening in the market for connectable products to address a lack of economic incentives for manufacturers to prioritize security, as highlighted in [20]. Governments are taking measures to enforce compliance and penalties, with some already implementing regulations. In the UK, for instance, mandatory measures are being implemented, albeit with a limited scope: eliminating universal default passwords, implementing vulnerability disclosure policies, ensuring timely software updates, and providing transparency on security update duration [21]. The regulator will have the power to impose significant sanctions, including fines up to £10 million or 4% of the

company's annual turnover. The Bill came into effect on 29 April 2024.

The European Commission has published the "Proposal for a Regulation of the European Parliament and of the Council on horizontal cybersecurity requirements for products with digital elements and amending Regulation", also known as the European Cyber Resilience Act (CRA). A significant portion of the Act outlines mandatory risk management and governance requirements throughout the lifecycle of digital products. These requirements will primarily be the responsibility of manufacturers, but also extend to importers and distributors.

Annex I of the EU CRA outlines essential cybersecurity requirements, but these are defined at a high level, making implementation and enforcement challenging. In contrast, the UK's approach prioritizes basic requirements for initial enforcement. The EU Act aims to comprehensively regulate security requirements, including generic ones, such as: "Products with digital elements must be designed to ensure appropriate cybersecurity based on risk" or "Products with digital elements must be delivered free of known exploitable vulnerabilities". Companies developing secure systems have long advocated for these requirements, but their implementation has proven difficult. The effort required from small, inexperienced manufacturers and the likelihood of success are unknown. However, it is likely that implementing and effectively enforcing these regulations will take considerable time.

As of May 2024, the CRA has reached a significant milestone, receiving formal approval from the European Parliament in March 2024. Although this marks a crucial step forward, the CRA still requires formal adoption by the Council before it can come into force. Once adopted, manufacturers will have a two-year grace period to adapt and implement the new requirements. Considering this timeline, it is likely to take several years before the first fine (up to €15 Million or 2.5% of the offender's total worldwide annual turnover) is imposed for non-compliance.

The slow deployment of IoT security legislation can be attributed to the inherent complexity of implementing security measures. There are no straightforward solutions, and most manufacturers struggle to transform their product development processes to prioritize security. The regulatory landscape for IoT security is currently undergoing significant changes, and the journey ahead is intricate. Our research aims to develop innovative solutions that help

manufacturers improve IoT security and comply with regulations.

1.3 Motivation and Background

Building on the need for tailored security solutions, and acknowledging the challenges of legislation and compliance in IoT security, this section outlines the motivation and background for our research.

The advent of the Internet of Things and Wireless Sensor Networks has brought forth a new era of connectivity and data-driven decision-making. However, this technological leap also introduces significant security concerns that must be addressed to ensure the trustworthiness and reliability of these systems. The struggle of manufacturers to develop secure products and the slow deployment of legislation, as discussed earlier, underscores the urgency for effective and efficient security mechanisms.

The motivation for this research stems from the pressing need to secure WSNs against evolving threats while accommodating their inherent limitations. The background of this study is rooted in the observation that traditional security solutions are not viable for WSNs due to their high computational demands and energy consumption, which are incompatible with the resource-constrained nature of sensor nodes.

This research is driven by the goal of developing security mechanisms and systems that are both effective and efficient, capable of protecting WSNs without compromising their performance or lifespan.

1.4 Thesis Scope and Contributions

This thesis proposes innovative solutions to address security challenges in WSNs. The research objectives focus on how to achieve robust authentication in WSNs with minimal computational overhead, and how to enhance software vulnerability analysis using deep learning techniques.

By achieving these objectives, we contribute to the development of secure and sustainable WSNs, addressing a critical need in the IoT ecosystem.

The present work is divided into two main parts.

Part 1: Novel Lightweight Authentication Scheme

This part presents the following contributions.

Comprehensive Overview of Message Authentication Schemes A thorough analysis of contemporary message authentication schemes is provided, delving into the core strategies and recent trends shaping the field. By synthesizing a vast array of protocols and schemes, this analysis offers valuable insights into the evolution of authentication techniques within resource-constrained environments.

Novel Lightweight Hybrid Authentication Scheme This work introduces a novel and efficient lightweight authentication scheme specifically designed for WSNs. Recognizing the limitations of traditional cryptographic approaches on resource-constrained devices, this scheme prioritizes efficiency. It achieves robust communication security within the network by applying a hybrid approach that combines radio fingerprinting with lightweight cryptography. This combination minimizes resource consumption, resulting in a secure and efficient authentication protocol.

The validity of this contribution is confirmed through a robust evaluation framework, comprising a real-time implementation of the authentication system, accuracy assessments, and energy performance evaluations. These methods collectively demonstrate the effectiveness and efficiency of the proposed scheme, providing a comprehensive understanding of its capabilities and limitations in real-world scenarios.

Part 2: Shmulik - Deep Learning System for Software Vulnerability Analysis

This part presents the following contribution:

"Shmulik," a cutting-edge deep learning system designed to automatically analyze code and identify potential software vulnerabilities, is presented. While Shmulik can be applied to various C-based programs, its design prioritizes the specific needs of WSNs and Embedded Systems: WSNs rely heavily on embedded devices with limited processing power and memory to run their firmware. Shmulik, by design, can analyze firmware code on such resource-constrained devices and be integrated into the development process without overwhelming these systems.

This contribution is validated through a range of evaluation methods, including experimental evaluation on a deep learning-designed dataset, comparative analysis with state-of-the-art vulnerability detection tools, a focused case study on the popular open-source libtiff library, and the successful detection of zero-day vulnerabilities. These methods demonstrate the effectiveness and practicality of Shmulik in identifying software vulnerabilities, particularly for resource-constrained devices, which heavily rely on C code and benefit from Shmulik's specialized focus on this language.

Rationale and objectives

The rationale for incorporating both a robust authentication scheme and a deep learning-based software vulnerability analysis (referred to as Shmulik) within this thesis is rooted in the pursuit of a comprehensive security framework. A secure communication protocol within the WSN is paramount; however, it is not sufficient if the underlying software is riddled with exploitable vulnerabilities. Conversely, the effectiveness of Shmulik in identifying and mitigating software vulnerabilities is contingent upon the secure exchange of information that a lightweight authentication scheme can provide.

By addressing security at both the network and software levels, this thesis aims to establish a more resilient defense system against a broader range of threats. The authentication scheme ensures the integrity and authenticity of the communication within the WSN, forming the first line of defense. Meanwhile, Shmulik acts as a safeguard for the software itself, identifying potential vulnerabilities before they can be exploited.

While there are additional elements that could enhance the security of WSN solutions, this thesis adopts a pragmatic approach by focusing on the most fundamental aspects: secure communication and secure software. Authentication is the cornerstone of secure communication, ensuring that messages are exchanged between verified parties. Similarly, the integrity of the software is crucial, as it forms the operational backbone of the WSN.

1.5 Publications

Those contributions are supported by publications, published as peer-reviewed papers:

- Haenel, Arie, Yoram Haddad, and Zonghua Zhang (2019). "Performance Evaluation of Sensors Lightweight Security Mechanism". In: 2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems Workshops (MASSW).
- Haenel, Arie, Yoram Haddad, and Zonghua Zhang (2020). "Lightweight Authentication Scheme for Internet of Things". In: 2020 IEEE 17th Annual Consumer Communications Networking Conference (CCNC).
- Haenel, Arie, Yoram Haddad, Maryline Laurent and Zonghua Zhang (2021). "Practical Cross-Layer Radio Frequency-Based Authentication Scheme for Internet of Things". In: Sensors 2021, 21(12)
- Haenel, Arie, Maryline Laurent, and Yoram Haddad (2023). "Trends and challenges in Message Authentication Schemes for power-constrained WSN". Submitted, under review.
- Haenel, Arie, Yaakov Cohen, Yocheved Butterman, Polina Frolov, Yoram Haddad and Maryline Laurent (2023). "Shmulik: Enhancing Control-Flow Vulnerability Detection by Combining AI-Based Analysis and Human Expertise". Submitted, under review.

The following publications were also authored during the period of this thesis but are not directly related to the thesis work:

- Cohen, Yaakov*, Kevin Sam Tharayil*, Arie Haenel*, Daniel Genkin, Angelos D. Keromytis, Yossi Oren and Yuval Yarom (2022). "HammerScope: Observing DRAM Power Consumption Using Rowhammer". In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, pp. 547–561.
- Gonen, Gil, Ronen Haber, and Arie Haenel. "Secure captcha test." U.S. Patent No. 10,356,073. issued July, July 16, 2019.
- Hirschberg, Benyamin, Moshe Kravchik, Arie Haenel, and Hillel Solow. "Ransomware key extractor and recovery system." U.S. Patent 10,387,648, issued August 20, 2019.
- Ashkenazi, Oded, Moshe Kravchik, Arie Haenel, and Benyamin Hirschberg. "File-type whitelisting." U.S. Patent 10,540,509, issued January 21, 2020.

1.6 Thesis Structure

To provide a clear overview of the organization and flow of this thesis, we outline the structure and content of each chapter below.

This thesis is structured into the following chapters.

- Chapter 2 explores the trends and challenges in message authentication for power-constrained WSNs, including background information and the objectives of secure authentication schemes.
- Chapter 3 introduces a practical cross-layer radio frequency-based authentication scheme for IoT, detailing its design, evaluation, and security considerations.
- Chapter 4 examines various code analysis techniques that are pertinent to the security of WSNs.
- Chapter 5 discusses the enhancement of control-flow vulnerability detection by integrating AI-based analysis with human expertise, along with the associated challenges and evaluations.
- Chapter 6 provides a concluding overview of the thesis, summarizing the main findings, contributions, and implications for future research.

Part I

Novel Lightweight Authentication Scheme

Chapter 2

Trends and challenges in Message Authentication Schemes for power-constrained WSN

Contents

The Maroochy Shire sewage system attack was the first widely reported cyber attack on an industrial control system (ICS), in 2000. During this attack, a disgruntled former employee transmitted radio messages to the central station, spoofing their source address with the identification of a specific remote pumping station. It led to some massive release of raw sewage and dealt huge environmental damage [22]. Analysis showed that the system lacked basic security detection and protection mechanisms, the authentication being the most fundamental one. Sadly, more than twenty years later, not enough has changed since then, as shown in [23].

Authentication emerges as a pivotal focus within IoT security, encompassing message authentication and entity authentication. This chapter addresses recent advancements in message authentication relevant to power-constrained devices utilized as sensors in Wireless Sensor Networks (WSN). By consolidating and analyzing emerging trends in message authentication, this survey seeks to bridge existing gaps in understanding and provide actionable insights for securing WSN deployments in resource-constrained environments.

The chapter is organized as follows.

In Section 2.1, we will provide some background information on the challenges faced by the industry and the governments in securing IoT devices in general, with a specific focus on WSNs.

In Section 2.2, we provide an overview of the challenges and trade-offs in designing secure IoT devices, discuss the importance of understanding the context of a proposed solution, and present a comparative summary of the families of modern message authentication schemes.

In Section 2.3, we will provide an overview of the traditional approaches to message authentication, which are widely used and have been extensively studied.

In Section 2.4, we will review lightweight alternatives to legacy approaches, which have been developed to meet the growing demand for more efficient and cost-effective solutions for resource-constrained devices such as WSN sensors. This section presents trends in this domain, classified by categories.

Subsequently, in Section 2.5, we will propose guidelines for recommendations and adoption strategies by the industry.

2.1 Background

2.1.1 Main characteristics and specificities of Wireless Sensors

Wireless sensor networks (WSN) are communication systems connecting sensor nodes that have the purpose of facilitating the acquisition and monitoring of environmental information. The nodes are usually resource constrained, to keep their cost as low as possible, and to be deployed in number. For the sake of our study, some relevant characteristics of the nodes are:

- Resource constraint devices (limited computational power, low energy, small battery, small memory)
- Large Attack surface (both remote and physical)
- Lack of standardization

It is important to note that while the message authentication schemes proposed in this chapter may indeed have applicability beyond power-constrained wireless sensor networks (WSN), the focus was deliberately maintained on solutions suitable for resource-constrained devices. Despite their potential relevance to other scenarios such as 5G networks or IoT

environments, the selected schemes were chosen specifically for their compatibility with the constraints and requirements unique to WSN nodes. This emphasis on resource efficiency and suitability for power-constrained devices was a deliberate decision aimed at addressing the distinct challenges faced by WSN deployments. Therefore, while acknowledging the potential broader applicability, this work prioritizes solutions well suited to the specific needs of WSN environments.

2.1.2 Attacks on WSN

The security of these networks is a critical concern due to their vulnerabilities to various types of attacks. In this section, we will present the most common attacks that pose a threat to the security of WSN and how they impact the Confidentiality, Integrity, and Availability of the network.

- MitM (Man-in-the-Middle) Attack: In this attack, an attacker intercepts and alters the communication between two parties without their knowledge. This type of attack is particularly dangerous for WSN as it can result in compromised security and incorrect data transmission.

- Sybil Attack: This attack involves a malicious node creating multiple identities to gain control of the network. This can result in denial of service, false information propagation, and network partitioning.

- Wormhole Attack: This attack involves creating a virtual tunnel between two remote locations, allowing an attacker to intercept and manipulate data transmitted through the network.

- Sinkhole Attack: In this attack, closely related to the wormhole attack, a malicious node attracts and absorbs all the traffic in the network, resulting in denial of service and false information propagation.

- Spoofing: Spoofing involves faking the identity of another device, allowing an attacker to impersonate a legitimate node and access sensitive information.

- Replay Attack: In this attack, an attacker captures a valid transmission and retransmits it, either to interfere with the network or to gain unauthorized access.
- Cloning: This attack involves copying the identity of a legitimate node to create a duplicate, which can be used for malicious purposes.
- Physical Attacks and Tampering: Physical attacks on WSN nodes, such as theft, destruction, or tampering, can result in loss of data and compromise of network security.
- Denial-of-Service (DoS) attacks: DoS attacks aim to disrupt the normal functioning of a network by overwhelming it with excessive data or interference. Here are a few types of DoS attacks, relevant in WSNs:
  - Jamming attacks involve sending a high level of noise or interference to disrupt the normal functioning of the network.
  - Flooding attacks aim to overwhelm the network with a large amount of data, causing resource exhaustion and denial of service. The goal is to saturate the network's bandwidth and computational resources, making it unable to process legitimate requests.
  - Routing attacks involve the attacker manipulating the routing information, preventing legitimate traffic from reaching its destination.

Attack	Confidentiality	Integrity	Availability
MitM	✓	✓	
Sybil		✓	✓
Wormhole	✓	✓	✓
Sinkhole	✓	✓	✓
Spoofing	✓	✓	
Replay		✓	
Cloning	✓	✓	
Physical Attacks	✓	✓	✓
Denial-of-Service			✓

Table 2.1: Relationship between Attacks on WSN and CIA Model

Table 2.1 summarizes the relationship between the listed attacks and the CIA (Confidentiality, Integrity, Availability) security model.

2.1.3 Security Objectives

There may be multiple security objectives to WSN, and they don't have to be the same for different systems, but each project usually has to derive its own objectives based on the business functional and nonfunctional requirements. [24] lists fifteen commonly sought security objectives in modern WSNs. We can highlight the most basic ones:

- Confidentiality: Secret information is protected from unauthorized disclosure.
- Integrity: Data are verified for accuracy and completeness.
- Availability: Ensure timely and reliable access to the service and data.

  The three previous properties are often referred to as the *CIA triad*.

- Authentication: Verification of the identity of an sender (person, node, server, message, ...).
- Authorization: Verification of the rights of access or action by a certain party.

There are many more objectives that may be relevant to WSNs but are not in the scope of this research (non-repudiation, backward and forward secrecy, resilience, etc.).

It should be noted that [24] lists *Energy Efficiency* as a security property. Nonetheless, our approach considers it as a non-functional requirement or property that may impede the implementation of some security mechanisms. Thus, when designing a practical solution for WSNs, it is crucial to consider the potential impact of energy efficiency.

The main security objective that we will focus on in this contribution is Message Authentication. This objective can be achieved in various ways, which we will explore. For instance, a message can be either self-contained, including proof of its source identity and data integrity, or it can be part of a secured session. Throughout the study, we will discuss different approaches to message authentication in WSNs.

2.2 Challenges and Trade-offs in Designing Secure WSN Authentication Schemes

A perfect and universal security scheme doesn't exist and probably never will. Even a non-perfect acceptable *universal* scheme is improbable. The multiplicity of proposed solutions and

their evolution can be explained by the conflicting needs of different systems. Energy efficiency, bill of materials, performance, maintainability, and, as we have seen in Section 2.1.3, security objectives come into competition and are sometimes partly mutually exclusive. And each of these can have their own trade-offs, e.g. computation vs storage, space-time trade-offs, etc. As so, it's the responsibility of the architects to manage the conflicting demands, taking into account the requirements and specific features of each project.

Designing secure authentication schemes for Wireless Sensor Networks (WSNs) presents unique challenges. One of the most significant challenges is the resource-constrained nature of sensor nodes, which typically have limited processing power, memory, and battery life. This constraint requires the development of lightweight solutions that can operate efficiently within these constraints.

Traditional cryptographic algorithms, while offering robust security, often require high computational power and memory usage, making them unsuitable for WSNs. Therefore, a major challenge lies in designing lightweight authentication schemes that can balance security with resource limitations.

Furthermore, the need for schemes to be scalable and adaptable to changes in the network topology adds another layer of complexity. The wireless nature of WSNs also introduces additional challenges such as the risk of eavesdropping, necessitating the need for secure wireless communication protocols.

Finally, there is an inherent trade-off between achieving high levels of security and keeping the authentication protocol lightweight. Schemes with strong cryptographic primitives offer robust security but might incur higher computational overhead. In contrast, lightweight schemes might be faster, but potentially more vulnerable to certain attacks. Therefore, finding the right balance between security strength and resource consumption for a specific WSN application is crucial. This balance ensures that the chosen solution provides the most value and effectiveness for its intended use.

For the sake of our discussion, it makes very little sense to put in the same comparison all kinds of system with competing characteristics, e.g. a security system of maintained sensors connected to the energy grid, even if labeled "lightweight" [25, 26], and optionally running

strong processors and crypto accelerators, with cheap agri-tech battery-powered sensors deployed for years in the field to take measure of soil moisture. The security needs and requirements are not the same, and what is acceptable for one solution may be totally inadequate for the other. This is why we categorize the state-of-the-art research reviewed in this chapter into distinct families, allowing for a more meaningful analysis of the different approaches and techniques.

There is no silver bullet. This is why it's primordial to understand what is the context of a proposed solution, what are not only the problems it tries to solve, but also what are the underlying limitations, even when they are not always clearly stated. For this reason, some IoT related researches are not included, since they did not fit for lightweight solutions. A few exceptions are made for proposals of special interests and that can be adapted for power constrained environments.

In the following sections, we provide a comprehensive overview of modern message authentication schemes and highlight the main strategies and trends that have emerged over the years. A large number of protocols and schemes have been proposed, and we aim to outline the main strategies here. This review is not an exhaustive list, but a comprehensive analysis of the state-of-the-art research in this field.

2.3 Cryptographic Authentication Approaches for General Computing Systems

Cryptography based protocols are the most natural and legacy choice for message authentication. Widely used methods for cryptography based message authentication can be classified in two categories: based on symmetric or on asymmetric cryptography, and the differentiation is usually clear based on the way the keys are created and used. Symmetric key-based message authentication methods include message authentication codes (MACs) or authenticated encryption (AE), while asymmetric key cryptography allows digital signatures. In some cases, though, some asymmetric cryptography primitives can be also used for key establishment of keys that can be applied for MAC or AE, making the taxonomy not always "clean-cut".

In order to authenticate a message between two parties, one needs to first agree on some common key. There are several strategies to achieve this agreement. Types of key exchange strategies:

- Pre-shared key (PSK) - The key is provisioned in advance in the device, e.g. during manufacturing, or during some enrollment phase by the administrator. The disadvantage is that it is not easily replaceable. If the PSK is compromised, there is need to revoke it. PSK based systems usually use symmetric key cryptography schemes.
- Cryptographic key negotiation - Commonly accomplished on the basis of cryptographic key establishment protocols. The most widely used ones are based on the following:
  - Discrete logarithm cryptography, like the venerable Diffie-Hellman Key Exchange protocol [27] and its descendants, like the commonly used Elliptic Curve Diffie-Hellman protocol, and other variations[28]
  - Integer factorization cryptography, like RSA Key-pair generators[29]

  A caveat to avoid is that these protocols don't protect by themselves against man-in-the-middle (MITM) attack, and need to add some mechanisms to protect the identity of the parties as part of the key establishment. Ways to protect the identity usually involve public key infrastructures (PKI), e.g. use of certificates and add another "layer" of asymmetric cryptography to the key establishment protocol.
- Post-quantum cryptographic key establishment - Post-quantum cryptography (PQC, also known as "Quantum-safe" , or "Quantum resistant" cryptography) is intended to replace some algorithms of the "classical" cryptography endangered by the advent of quantum computers, especially asymmetric algorithms, very fragile against Shor's algorithm[30]. Although not widely used yet, especially in the context of WSNs, PQC is likely to see rapid growth in the coming years, with the help of standardization. As part of their Post-Quantum Cryptography Standardization process, NIST recently announced that they will standardize CRYSTALS-Kyber [31] as a post-quantum key establishment algorithm, with more algorithms which could be added in the coming months [32].
- Quantum key exchange - Known as Quantum Key Distribution (QKD), it makes use of some

unique properties of quantum mechanics, and that should guarantee eavesdropper detection. In theory, QKD could guarantee perfect security, but the difficult implementation of such systems and the fact that the QKD protocols do not provide authentication, the theoretical promises do not yet fully meet the reality. There are a very few deployed systems using QKD, most of them are proof-of-concept systems or for research. Recently, the enthusiasm around QKD has declined, as main governmental institutions recommended the use of Post-Quantum Cryptography (PQC) solutions and not of QKD, claiming that PQC solutions are more cost effective and more maintainable than QKD solutions [33, 34, 35, 36].

From a conceptual level, it's clear that for each of these strategies, a root-of-trust element must be present, and registered in advance, directly or indirectly, to be trusted by another party: in the case of PSK, this is a symmetric key provisioned or negotiated in a safe environment before deployment; for key negotiation, it's a proof of identity, usually a certificate based on public-private key pairs. There is no miracle and it's not possible to *securely* establish a symmetric key from thin air.

After a common key has been agreed upon, the most common approach to achieve authentication (message authentication or source identification) is by using some MAC legacy cryptographic primitives.

For message authentication, HMAC [37] is one of the most popular choice. It is providing integrity based on a keyed-hash message authentication code, the symmetric key being shared between the parties. Several underlying cryptographic hash functions are standardized. When confidentiality is required, encryption is used in addition of integrity mechanism, and for session based communication, protocols that implement both are usually used. TLS is probably the most widely used secured communication protocol, as it is the underlying security layer of HTTPS, and can be used with combinations of a vast number of cryptographic primitives [38].

When the underlying protocol is UDP instead of TCP, DTLS replaces TLS [39] as its datagram counterpart. It is used for securing media streams, as implemented in DTLS-SRTP [40], Voice-over-IP (VoIP), and in VPNs to avoid the TCP-over-TCP problem (a.k.a. "meltown problem") [41].

2.4 Message authentication strategies adapted for sensors

In the previous sections, we gave an overview of the legacy methods of message authentication. But these legacy protocols can be too demanding for resource-constrained devices. This is why there is a need for lightweight schemes.

The following sections are ordered as follows:

- "Classic" cryptographic approaches. We'll discuss cryptographic primitives and their use as part of lightweight authentication protocols (Section 2.4.1)
- Methods based on physical properties (Section 2.4.2)
- and a discussion around recent proposals based on distributed ledger technologies (e.g. blockchain) (Section 2.4.3)

2.4.1 Lightweight Cryptography protocols/schemes used for resource constrained devices authentication

Lightweight Cryptographic primitives

We first define what we mean in this work by "lightweight cryptography". Cryptography algorithms can be "versatile", in the sense that the same algorithm can be used to fulfill the security properties in a variety of scenarios. For example, twenty years ago, the cryptographic hash function MD5 [42] was widely used for file checksum, password hashing, keyed-hashing for message authentication, signature digest, electronic discovery, and more. Such ubiquitous algorithms are still present, e.g. in the category of hash functions, SHA-2 and SHA-3.

In order to deal with the trade-off between security and performance in cryptographic algorithms, solutions have been proposed in the form of lightweight or ultra-lightweight cryptography. By [43]'s definition, " ultra-lightweight cryptography deals with primitives fulfilling a unique purpose while satisfying specific and narrow constraints". For example, certain algorithms are designed for better performance, at a price of higher complexity and higher cost. Others are based on simpler implementation requirements, but need more rounds to get a sufficient level of security for specific use cases. The idea is not new: in the 90's, the preferred

stream ciphers algorithms were chosen to fit the bill of material, i.e. optimized for cheap HW like A5/1 used in GSM, or optimized for general usage CPU like RC4, which was optimized for 8-bit processors. But then, the standardization of some generic algorithms, well reviewed and well trusted, dictated a few algorithms suitable for almost all cases. For example, when NIST launched the AES competition to define a standard for block cipher, the goals included that the new algorithm should be efficient in software and hardware, support commonly used modes of operations, support several block sizes, etc. [44]. The winner was chosen by being the best overall, but it didn't have to be the best at anything in particular. It made AES a versatile symmetric cipher, able to cover even the needs of stream ciphers through stream cipher modes like Counter-Mode.

But the trend changed again, with more and more specialized categories of standardized algorithms. We will refer the readers to some of these standardization projects, like the ECRYPT Stream Cipher Project (eSTREAM) [45] or the NIST Lightweight Cryptography Standardization Process [46, 47].

In the domain of hash functions used as primitive for message authentication, we need to consider the required security properties, as explained in Section 2.2.

Regarding asymmetric primitives, the choice is more limited, and the standard one is usually based on elliptic curve cryptography (ECC). But even then, the most "lightweight" ECC-based system will require 10x-1000x more computation and time than symmetric algorithms. Of course, one should not just compare them this way, since different properties are covered.

We refer to [48, 49] for detailed reviews on the lightweight cryptographic primitives.

Schemes based on asymmetric primitives

The following researches are using asymmetric primitives as part of their authentication scheme.

[50] proposed an authentication scheme for WSN, based on ECDH, as an improvement on [51].

[52] introduces a revocable lightweight authentication scheme for resource-constrained devices in cyber-physical power systems (CPPSs). The scheme combines ECC and certificateless cryptography (CLC) to negotiate a secure session key with low computation and communication

costs. A real-time key update strategy is designed to improve security and theoretical analysis proves the security with respect to existential unforgeability against adaptively chosen message attacks (EUF-CMA). The experimental results confirm the feasibility and effectiveness of the proposed scheme, and the security analysis shows that the proposed scheme can satisfy security requirements of resource-constrained devices in CPPSs. However, the paper focuses only on the security authentication within a single domain, and future work is needed to extend the scheme to cross-domain security authentication.

[53] presents a model that provides integrity of data transmitted from individual nodes to a central node (CN), based on identity-based cryptography where the public keys are generated by a deterministic function and based on the node identity. Furthermore, the model uses an online/offline signature scheme, an idea first introduced by [54], where even if the scheme uses asymmetric cryptographic primitives, most of the computation is done on the powerful central node, while the sensor node has only few low computational needs.

More researches focused on using the concept of online/offline scheme. We refer to [55] for a literature review of lightweight, provably secure identity-based online/offline signature techniques (IBOOST). [55] themselves presented another new IBOOST-based design, using fractional chaotic maps, with an improvement over predecessors in the fact that the offline pre-stored information can be reused, with limitations, for more than one signature. The system architecture is presented as tailored for 5G-WSNs, but the principle is probably applicable to other types of resource-constrained WSNs.

A proxy multi-signature scheme is a scheme for which a proxy signer can sign on behalf of multiple signers [56]. [57] proposes an improvement on [58]for a more efficient authentication protocol, based on proxy multi-signature, adapted for sensors.

As explained in section 2.3, TLS is the de facto standard for secure communications, and DTLS is the variant that supports datagram-based communications, which is very interesting in the case of short messages in the context of IoT devices transmissions.

[59] proposes a Payload Encryption-based Optimisation Scheme (PEOS) for Advanced Metering Infrastructure (AMI) Sensor Networks, providing efficient and lightweight authentication for sensor devices and control messages. PEOS integrates and optimizes the important

features of Datagram Transport Layer Security (DTLS) in the Constrained Application Protocol (CoAP) architecture [60], and improves its efficiency by optimizing handshaking, removing duplicate features of the messages, propose improvements over AES like parallel execution of S-boxes and delayed Mix-column operations. PEOS also incorporates dynamic key generation and token processes to avoid DDoS and confidentiality attacks. The scheme was evaluated on the Contiki OS using the Cooja simulator and showed improved throughput compared to existing payload-based schemes and basic DTLS.

As noted in [61], the computational cost of DTLS software implementations is *"prohibitively expensive"*. Since general purpose microcontrollers found in most resource constrained embedded devices (for which their manufacturers tend to try to keep a low price) can't handle this kind of power-demanding processing, the authors propose a dedicated DTLS reconfigurable energy efficient custom hardware accelerator design to offload the heavy cryptographic computation. We can safely estimate that till the point that such a design is made very common, its cost will be a negative factor for its integration in the most majority of sensors. But such a purpose-oriented HW is probably the only way to propose asymmetric cryptography-based solutions for cheap low power devices.

So, while asymmetric cryptography is a classic approach for building authentication schemes, its high computational cost makes it unsuitable for resource-constrained devices like low-cost IoT sensors. This trade-off between affordability and processing power has led to the development of alternative methods based on symmetric cryptography.

Paper	Crypto Primitives	Key Elements	Validation	Energy Efficiency
[50]	ECDH	mutual authentication and key agreement protocol	Scyther automated security protocol verification tool; manual security review	188 bytes per message; 4 handshakes
[52]	ECC and certificateless cryptography	Revocable authentication; Real-time key update	Formal proof EUF-CMA (existential unforgeability against adaptively chosen message attacks)	304 bytes per signature transmission
[53]	Online/offline signature scheme	Key derivation from node identity	Formal proof EUF-CMA	400 bytes per signature transmission for 128-bit of security
[55]	Identity-based online/offline signature techniques	Based on fractional chaotic maps, originally designed for massive devices in 5G WSNs	Formal proof of unforgeability of online/offline identity-based signature under chosen message attack	480 bit (60 bytes) per signature (not including the whole transmission)
[57]	ECC algorithms; SHA-2 hash	Based on proxy multi-signature	Experimental analysis	150 bytes per signature communication
[59]	Datagram TLS (and underlying protocols)	Datagram Transport Layer Security (DTLS) in the Constrained Application Protocol (CoAP) architecture	Cooja simulator	120 bytes per message authentication overhead
[61]	DTLS (including underlying protocols ECDSA, ECDHE, AES-GCM, SHA-256)	Reconfigurable DTLS HW accelerator full	HW experimental implementation	High energy efficency through HW acceleration of standard protocols

Table 2.2: Asymmetric crypto based schemes

Schemes based on symmetric primitives

Symmetric cryptography algorithms are usually much computationally lighter than asymmetric ones. For this reason, solutions based on symmetric primitives are sought as the better alternative, when adequate. This may not be suitable in all contexts, and may need some pre-shared secrets, as explained in Section 2.3.

We present first two lightweight IoT protocols that have been widely adopted in recent years: LoRaWan and EnOcean. It may be noted that these communications protocols took security into account as part of their original design. Some recent research has pointed out some security gaps and in some cases proposed some changes to improve the protocol security.

LoRaWan

LoRaWan is the communication protocol used on top of the underlying LoRa physical layer [62]. LoRa is used for long-range (∼15 kilometers with line of sight) and ultra-low power communications. Though LoRa does not have intrinsic security, LoRaWan mandatory security features include authentication, integrity, and encryption. It is using symmetric-key cryptography, where session keys can be provisioned in advance in the device (ABP, "Activation by personalization") or negotiated (OTAA, "Over-the-air Activation").

A false sense of security can be created by the protocol promises, and as noted by the LoRa Alliance itself, LoRaWan may theoretically be secure, "the implementation matters" [63]. Several offensive security researches have been published to point out some possible vulnerabilities in the protocol, some practical attacks demonstrated, and some mitigations proposed [64, 65, 66]. Specifically related to message authentication, we can point to:

- Bit-flip attack [67]. Due to the malleability of the CTR mode used for the encryption, it is possible to modify the encrypted data without the key by knowing the position of the targeted part. The message authentication is based on the MIC (Message Integrity Code). It is only 4 bytes long and as so, relatively exposed to a brute-force attack. A shuffling of the transmitted data is proposed as a mitigation.
- Replay attack [68]. Because the frame counter can be reset, if the same session is used,

messages from previous sessions can be replayed. ABP-activated devices use static session keys and, as so, are very fragile against this attack. Keeping the counter in nonvolatile memory to avoid having it reset on system reset was proposed and implemented in version 1.1 of the protocol. OTAA activated devices may theoretically be vulnerable if the counter overflows, but it is far less practical as an attack. A rekey should be mandatory in the case of a counter overflow.

- More possible attacks have been discussed, assuming some problematic but realistic assumption, like untrusted network server possibly leading to rogue data injection [69]

EnOcean

EnOcean is a standardized technology designed for self-powered wireless devices, using energy harvesting to enable battery-less sensors and controllers. Its communication range is ∼30 meters indoor, and ∼300 meters with line-of-sight [70, 71].

It claims a security-by-design approach and is based on symmetric primitives, namely AEC-CMAC for integrity and authentication and AES-CBC or Variable AES (VAES) for confidentiality. Each device has a chip ID, and a pre-shared key (PSK). A rolling code is used to avoid replay attacks. To ease the process of replacement of a device in a network (through the "teach-in" process), a base ID can be generated and is used to set the replacement device. The initialization vector being fixed, the AES-CBC mode is considered insecure, and VAES is the recommended mode of operation for encryption [72].

It has been noted that the teach-in process involves the PSK, usually found on a sticker on the device. [73] found that an attacker could launch a key compromise impersonation attack (KCIA) after obtaining the PSK (e.g., simply reading it off the sticker). They proposed an improvement on the original protocol by adding a trusted third-party server, which would send new communication keys.

The TESLA family

The Timed Efficient Stream Loss-tolerant Authentication (TESLA) protocol, introduced in 2002, is an authentication protocol that provides source and message authentication. It

achieves some asymmetric properties using only a symmetric cryptographic primitive (a one-way hash function), a notable advantage compared to the traditional approach that relies on computationally expensive public-key cryptography. The asymmetry properties are obtained by using loosely synchronized clocks and delayed key disclosure. TESLA's goals are source authentication, message integrity, and optionally, confidentiality, in untrusted wireless networks [74]. However, even if the messages achieve these asymmetric properties with only the use of hash functions, the protocol still requires some bootstrap of the session setup to ensure trust establishment of the source identity, for example, through legacy digital signature.

Although the original TESLA protocol offered significant benefits, it also had limitations, particularly with respect to resource consumption. To address this, Perrig et al. proposed μTESLA, a variant designed for environments with severe resource constraints [75]. However, μTESLA authors recognized some shortcomings, such as susceptibility to compromised sensors, denial-of-service attacks, lack of non-repudiation, etc.

Several research efforts have focused on refining and extending TESLA's capabilities. Studer et al. introduced TESLA++ to reduce memory requirements by separating message authentication codes (MACs) from the messages themselves [76]. This variant, particularly suited for vehicular ad hoc networks (VANETs), also introduced a certificate verification mechanism to mitigate denial-of-service attacks. Liu et al. proposed a multilevel key chain scheme to improve μTESLA's scalability and offer a degree of protection against denial-of-service attacks [77].

More recently, [78] addressed edge cases in TESLA's operation, such as key chain expiration and authentication of final packets, by proposing a method for on-the-fly key chain regeneration, without the need to restart full synchronization. Additionally, [79] presented a mechanism to extend TESLA for enhanced authentication and non-repudiation, even in the presence of a single compromised node. Even if their method does not work for a higher number of rogue nodes, the authors note that it is an improvement on classical group key management schemes, broken with even one rogue node.

TESLA's influence has extended beyond academic research. It has been proposed for secure communication in VANETs [80, 81, 82] and has been adapted in the Open Service Navigation

Message Authentication (OSNMA) service for the Galileo satellite system [83].

Recent original academic approaches

Some recent academic research explores new approaches using symmetric primitives for lightweight message authentication schemes.

[84] made some critiques (insider and session-specific random number leakage attacks, lack of perfect forward secrecy) on [50] and proposed a new scheme called WSN-SLAP, that provides perfect forward secrecy and mutual authentication, and uses hash functions and XOR operations for low computational overhead.

[85] proposed a family of lightweight message authenticated encryption schemes, requiring only one pass for both authentication and encryption. The authentication part is based on a simplified keyed hash function using a dynamic key, one round of the hash function, and using only simple operations (substitutions, permutations, and XOR). The encryption part is based on a "dynamic" CTR mode, using the authors' idea of dynamic key, where the cryptographic primitives are updated for each new input message. This is an early-stage cryptographic primitive, and as such, it should undergo more scrutiny.

[86] suggested to use the CCM authenticated-encryption algorithm (counter with cipher block chaining message authentication code) [87] with additional flexibility based on the use of a token, the expiration time of which is determined by a trust value, and a variable sleep time based on the energy level of the sender.

[88] proposed an efficient message authentication protocol for short message ("RAM-MAC"), which replaces traditional MACs which would take a large part of the communication of short packets with a scheme using state chaining, random access messaging, and tag truncation to save data transmission. They claim to improve on previous MAC solutions seeking lower overhead in comparison of classic MACs (CuMAC [89], Mini-MAC [90], and ProMAC [91]).

Paper	Crypto Primitives	Key elements and security properties	Maturity Level	Evaluation
[62]	AES (CMAC, ECB and CCM modes)	Activation by personalization or Over-the-air Activation	Very mature and deployed	Formally (Scyther, ...) and experimentally verified
[70][72]	CMAC; AES	Highly energy efficient; Only VAES mode recommended (not AES-CBC)	Mature and deployed	Formally (ProVerif) and experimentally verified
[73]	CMAC; homomorphic hash; VAES	Based on [72], adding a TTP and the use of a homomorphic hash. Defense against Replay-Attack, tamper attack, KCIA	Research. Doesn't take the communication considerations into acountm based on immature primitives (homorphic hash).	Formal analysis (CPN, Dolev-Yao attacker)
[75]	μTESLA; SNEP (Secure Network Encryption Protocol)	Authenticated broadcast based on TESLA, for severely resource-constrained environments; SNEP for other security properties (confidentiality, data freshness, ...)	Seminal research, though not widely deployed	Experiment symulation and formally verified
[77]	TELSA; Multilevel key chain scheme	Some defense against DoS	Base of number of protocols, and proposed in various domains (VANET, Satellite communication , ...)	Experimental simulation
[76]	TESLA; ECDSA	Based on TESLA with reduced memory requirements; designed for VANET. Some defense against DOS.	No known wide deployment.	Experimental simulation
[78]	TESLA; New offset key-chains regeneration	Improved performance of the authentication generation time	No known wide deployment.	Experimentally verified
[79]	TESLA; SHAKE56; AES	Add non-repudiation, even in the case of a rogue node	No known wide deployment.	Formal security analysis and experimental performance evaluation
[84]	Hash functions and XOR	Add Perfect forward secrecy and mutual authentication	Protocol only. No implementation.	Formal (AVISPA, BAN, ROR model) and informal security analysis
[85]	Authentication by simplified keyed hash function using a dynamic key	Message Authentication Encryption, using one pass only	New cryptographic primitive requires further analysis. Additional steps are needed to finalize the scheme for inclusion in a complete protocol.	Preliminary cryptanalysis
[86]	CCM AE	Improved flexibility by additional use of trust value, expiration time and sleep time, based on energy level of the sender	Authentication and encryption protocol based on existing cryptographic primitives. Potentially forming part of a future protocol.	Experimental simulation
[88]	MAC optimized for short-messages	State chaining, random access messaging, and tag truncation	MAC construction, improving previous schemes. Potentially forming part of a future protocol.	Informal security analysis

Table 2.3: Symmetric crypto based schemes

2.4.2 Physical characteristics based

Biometric authentication has been used for user authentication, relying on unique physical or behavioral traits to create authentication mechanisms. Fingerprint, facial, palm, and iris recognition systems are widely used to unlock computer systems or doors. Inspired by the same principle of using physical characteristics, Machine-to-Machine (M2M) communications like Sensors-to-Gateway can also use physical properties or characteristics to establish a trust anchor for authentication.

By binding the identity of a device with unclonable physical characteristics, these methods provide alternatives to traditional methods such as PSK or certificates. This makes it more difficult to clone or spoof a device, even if the attacker were to extract information from it. These characteristics can be used for identification, authentication, and key establishment. In this section, we will review the primary methods that use this approach, highlighting recent trends in the field.

Physical Unclonable Functions

Physical Unclonable Functions (PUFs) are functions based on unclonable physical characteristics which constitute a set of properties that is seen as a unique fingerprint. The main applications of PUFs are authentication and cryptographic key generation. There are different types of PUF hardware designs, and different strategies can be used together in the same device. We refer the reader to the recent review [92] that describes the different types of PUF architecture and discusses PUF-based user authentication and key agreement for WSNs.

- [93] proposes a SRAM-PUF-based entity authentication scheme for resource-constrained IoT devices to ensure their trustability. The proposed scheme uses challenge-response pairs generated from reordered memory addresses and the corresponding SRAM cells startup values. The experimental results show that the scheme can efficiently authenticate devices with low computation overhead and small memory capacity, and the stability of the startup values was tested under different environmental conditions. The scheme's requirements can be satisfied by resource-constrained devices, and future work includes further testing on the

SRAM cells and research to find a module to predict cell stability under different conditions. It uses SRAM-PUF to generate random, stable and tamper-resistant fingerprints, and to authenticate remote devices based on challenge-response pairs (CRP). The novel aspect of this scheme is that it is lightweight and computationally efficient and can be deployed in IoT devices with low computational power and memory capacity. The SRAM-PUF concept has been tested under different environmental conditions, and its stability and resistance to tampering have been confirmed.

- The authors of [94] proposed a DRAM-based PUF method to simplify the key generation process. The method addresses some crucial security needs of the PKI scheme, such as node registration, strong random number generation, and defining trust relationships among the different "thing" nodes. Additionally, the proposed method uses the concept of session keys to address the limitations of certificate validation and revocation, incorporating ECC for the generation of asymmetric key pairs. Experimental results show it is more efficient with 4-7 times lower energy consumption compared to state-of-the-art methods.
- [95] leverages cryptographic XOR operations, hash functions for secure communication, and physically unclonable functions (PUFs) for the generation of unique device-dependent identity and a lightweight security solution to prevent physical attacks. We can see that the "XOR operations" defined by the authors are some kind of stream cipher, using the PUF as the function generating the stream. It requires a one-time enrollment phase by a trusted party (similar to PSK). The protocol performs device-to-device and device-to-server authentication without requiring additional communication and computation resources, eliminating the need for multiple protocols. The security of the protocol has been analyzed against adversarial attacks and bad PUF model-based attacks and verified using the Scyther verification tool. The proposed protocol has been implemented as a prototype in a smart street light monitoring system and demonstrated to be robust and secure against different adversarial attacks and physical attacks found in practical scenarios.
- In [96], the authors present a PUF-based authentication and key establishment protocol. The proposed scheme enhances the reliability of PUF by incorporating error correction in the server and removing cryptographic hashing. The scheme has undergone formal verification

and has been proven to be resistant to masquerade, brute-force, replay, and modeling attacks, providing a reliable authentication solution 99%. The proposed protocol also reduces hardware overhead by implementing a lightweight challenge-response pair (CRP) obfuscation mechanism and stream authentication scheme inside the device. It also provides 60-72% reduction in look-up tables and register count compared to recently proposed approaches.

- [97] presents a PUF-based Lightweight Group Authentication and Key Distribution Protocol (PLGAKD) to secure IoT applications. The current group authentication and key management protocols use asymmetric ciphers which are too computationally expensive for IoT devices. The PLGAKD protocol employs PUF, a factorial tree, and the Chinese remainder theorem (CRT) to achieve group authentication and key distribution. It is designed to be more efficient than current protocols by reducing computation and communication overhead while ensuring the authenticity, integrity, and confidentiality of the data. The use of a factorial tree and CRT reduces the number of keys stored and the number of communication messages during the key renewal process, making it more efficient as the number of members increases.
- [98] proposed solution is an end-to-end mutual authentication and key exchange protocol that combines the use of PUF and certificateless public key cryptography (CL-PKC) on elliptic curve (ECDL and ECDH). This protocol requires only "three handshakes" without the need for real-time server participation, significantly reducing communication overhead. Their security analysis suggests that the proposed protocol is secure against various attacks, including certain types of physical attacks, and provides perfect forward secrecy. Furthermore, performance analysis shows that the protocol outperforms existing related protocols in terms of security features, protocol rounds, and communication cost.

  [99] propose authentication and key sharing scheme for wireless sensor networks (WSN) that uses a PUF as the low-cost hardware security primitive for resource-limited sensor nodes. The authors claim that their proposed scheme, which integrates Pedersen's verifiable secret sharing scheme (Pedersen's VSS) and Shamir's secret sharing scheme (Shamir's SS) with PUF, offers the desired security with low overhead and provides mutual authentication, presents resistance against impersonation attack, replay attack, echo attack, MitM attack

and against some types of DoS scenarios.

Radio Frequency Fingerprint

In Radio Frequency Fingerprint (RFF) systems, features of the transmitted waveform due to imperfections in the transmitter circuitry are analyzed, and "fingerprints" of the transmitter are inferred. This physical layer security method aims to differentiate the transmitters by only analyzing the transmission from the receiver. Still, it requires some enrollment phase, some feature extraction capabilities by the receiver, and some effective classification methods. To create such a system precise and practical enough is still a challenge, even if the idea is pursued for more than 20 years. Cheaper equipment and advancements in the domain of deep learning help make the research around RFF systems more reachable to the IoT world every year. Since no additional hardware, data, or software is needed at the transmitter side, this method is totally adapted for the creation of authentication techniques, ultra lightweight for a resource-constrained device. Here are some of the most promising recent research in the domain:

- [100] presented a conceptual development of a device authentication of transmitter in IoT networks ("RF-PUF"), based on the extraction and analysis of multiple features of the received signals (instead of being preamble based, or transient mode based only). The identification is carried out by a non-linear multidimensional classifier, implemented using an Artificial Neural Network (ANN). They reached the accuracy 99% in the simulation.
- [101] presented an implementation of an RFF identification framework using the preamble part of the message sent to extract features, with various improvements to the robustness of the system. They included an extensive experimental evaluation of LoRa devices. A k-NN classifier is used for classification of the devices (rogue or authenticated). Their results showed an accuracy between 75.80% and 98.50%. The authors suppose that the gap in the results may depend on the type of DUTs and the size of the training set, and they recommend training the system on larger devices sets.
- Recently, [102] advocated for a symbiotic protocol combining RFF and PUF circuitry to achieve mutual authentication with key exchange. A complete RFF-PUF protocol is yet to

be developed.

- In the next chapter, we will present our novel authentication scheme that addresses the security needs of battery-powered, limited-resource IoT devices. This scheme leverages Radio Frequency Fingerprinting (RFF) technologies and lightweight cryptographic authentication algorithms to create a hybrid authentication mechanism that reduces energy consumption while maintaining a defined level of security.

Paper	Root of Trust	Key Elements	Validation	Addressed attacks
[93]	PUF	SRAM-PUF based Challenge-response pair (CRP) authentication scheme, using a hash function to mask the values of SRAM cells	Informal security analysis; Experimentation and measurements (Arduinos)	Man-in-the-Middle (MitM); Sybil Attack; Spoofing; Replay Attack; Physical Attack
[94]	PUF	DRAM-PUF-based CRP for authentication; Uses ECC for asymmetric key pair generation without need for certificate	Formal verification (RoR and AVISPA); Informal security analysis	MitM; Replay Attack; DoS; Physical Attack
[95]	PUF	Path Changing Switch (PCS)-based arbiter PUF based CRP and session key establishment	Formal verification (Scyther); Performance analysis (FPGA)	MitM; Spoofing; Replay Attack; DoS; Physical Attack
[96]	PUF	3-1 Double Arbiter PUF (DAPUF) with masking function authentication; ECDH key establishment attacks	Formal verification; Prototype in FPGA	MitM; Spoofing; Replay Attack; Physical Attack
[97]	PUF	PUF, factorial tree, and the Chinese remainder theorem (CRT) based group authentication and key distribution model	Informal analysis	MitM; Spoofing; Replay Attack; Physical Attack
[98]	PUF	Ring Oscillator PUF and certificateless ECDL and ECDH based mutual authentication and key exchange protocol	Formal and Informal verification	MitM; Spoofing; Replay Attack; DoS; Physical Attack
[99]	PUF	PUF and Pedersen's Verifiable Secret Sharing (VSS) based CRP authentication scheme; Group Key distribution using Shamir Shared Secret; assume a strong PUF	Informal security analysis	MitM; Sybil Attack; Spoofing; Replay Attack; DoS; Physical Attack
[100]	RFF	Conceptual development of a PUF based on RF properties; Features extraction by receiver; ANN for identification	Simulation and SDRs based experimental setup; informal security analysis	Spoofing; Physical Attack
[101]	RFF	Signal-preamble based RFF; feature extractor by CNN; k-NN device classifier	Experimental evaluation (LoRa devices and SDRs)	Spoofing; Physical Attack
[102]	RFF & PUF	Conceptual cooperative mutual authentication and key establishment based on RFF&PUF based CRP	Concept	MitM; Sybil Attack; Spoofing; Replay Attack; Physical Attack
[103]	RFF & PSK	Hybrid scheme using RFF as main authentication method and Lightweight crypto if RFF result under a threshold; Embodiment using anchor nodes RSSI measurements with k-NN regressor for RFF authentication and Chaskey keyed-hash.	E2E implementation and performance analysis; Informal security analysis	MitM; Sybil Attack; Spoofing; Replay Attack

Table 2.4: Physical characteristics based schemes

2.4.3 Distributed ledger technology

Distributed ledgers are a type of database that is spread across multiple sites, nodes, institutions, etc. rather than being centralized in one location like a traditional database. These ledgers allow records to be kept and managed in a distributed manner. Distributed ledger technology (DLT) encompasses the frameworks and protocols that enable the use of a distributed ledger. It includes all the necessary components to implement a distributed ledger system. Blockchain is a type of Distributed Ledger, with a certain set of properties and features. Blockchain-like protocols have been discussed already 40 years ago, but they gained tremendous interest due to their first widely used application: Bitcoin cryptocurrency. While Bitcoin brought distributed ledgers into the limelight, DLT itself offers properties that can significantly benefit message authentication schemes. [104] note that despite its primary adoption in digital currency, the characteristics of blockchain — such as auditability and verification of actions between multiple parties — make it appealing for data management tasks (e.g., supply chain management)). These same attributes hold significant promise for enhancing the security and integrity of message authentication protocols within diverse systems and networks.

In the subsequent sections, we will explore the desirable attributes of DLT for WSN message authentication, discuss challenges related to resource-constrained solutions, and highlight recent innovative research proposals.

Distributed ledger technology and WSNs

In regard of WSN authentication, DLTs claim certain appealing characteristics.

- Decentralization: DLTs are distributed across a network of nodes, rather than being stored on a central server. This means that no single node has complete control over the data and that the network is resistant to attacks that target a single point of failure.
- Immutability: Once data are added to a DLT, it cannot be altered or deleted. This ensures that the data remains tamper-proof, and prevents malicious actors from modifying the data to their advantage.
- Transparency: DLTs provide a transparent and auditable record of all transactions that

take place on the network. This allows users to easily verify the authenticity of the data and ensure that it has not been tampered with.

- Resilience: Since the data are distributed across multiple nodes in the network, the loss of a single node does not affect the overall availability of the network. This means that even if some nodes fail or are taken offline, the network can continue to function and maintain its integrity.
- Smart Contract: DLT can enable the creation of smart contracts for WSNs, allowing for the automation of certain processes and decision making based on the data collected by the sensors.

Some properties depend on the type of DLT. For example, "Public DLTs" are permissionless and decentralized, while "Private DLTs" are not. "Federated DLTs" are in-between regarding some properties (e.g. require permissions, but usually partially decentralized).

Challenges specific to DLTs and WSNs

However, there are important challenges that need to be addressed to leverage DLTs in the context of IoT in general, and more specifically, for resource-constrained devices.

- Computation: Actively participating in the DLT/Blockchain requires some computation. In the case of PoW, this computation is very important, but even in the cases of more "lightweight" alternatives, the computation cost is still more substantial than the legacy cryptographic methods.
- Storage: DLTs typically store a complete record of all transactions that have occurred on the network, which can result in a large volume of data being stored over time. The fact that the ledger is decentralized means that there is a need for storage at multiple (if not all) locations.
- Communication: Much more messages are required in a DLT than in a centralized system. DLTs usually rely on a network of nodes to validate and record transactions, which can result in a large volume of communication between nodes.

- Energy consumption: DLTs can be energy-intensive, as they require significant computational power to maintain the network and process transactions. This can be a problem for resource-constrained devices, which may not have the necessary energy resources to support a DLT.
- Latency: The distributed nature of DLTs can also result in higher latency, or the time it takes for a transaction to be processed and added to the ledger. This can be an issue for applications that require real-time processing, such as those found in IoT and WSNs.
- Regulation: DLTs are a relatively new technology, and there is still a lack of clear regulation and standards around their use. This can create uncertainty and challenges for organizations that want to use DLTs in their applications.

Challenges of the consensus mechanism

One of the most essential components of DLTs is the consensus mechanism. It is a fault-tolerant mechanism that enables the network to reach agreement on the state of the ledger without the need for a central authority.

The most prominent criticism of early blockchains is that they are very computationally intensive, which is the opposite of what is needed for resource-constrained devices such as those in a WSN. This is because the most common consensus mechanism used by Blockchains, called proof-of-work (PoW), requires nodes to perform complex calculations in order to verify and validate new blocks added to the chain. These calculations are designed to prevent a single adversary from tampering with the chain. However, this also makes PoW very wasteful, as the calculations performed by nodes are discarded after they are used. This is a problem for WSNs, where devices have limited resources and cannot afford to waste energy on complex calculations. According to [105], it is estimated that the global energy footprint of the Bitcoin, which uses a PoW as its consensus mechanism, is around 150 terawatt-hours of electricity annually. To put this number into perspective, this is approximately the same amount of electricity that countries like Sweden, Norway, or Egypt consume in a year.

Specific to resource constrained WSN elements, there are two main problems with PoW:

- Even if the distributed characteristics of Blockchain make it well suited for the IoT, its

consensus mechanism, PoW, is computationally very heavy and not adequate for resource-constrained devices. The issue is that these PoWs are computational and memory intensive, and time consuming. This means that it will be costly to have sensors capable of such computation and energy inefficient, which is a huge disadvantage of this approach. Not all the nodes have to be "miners", but that would mean that only central parts of the network can actively participate to the creation of new blocks. All the other nodes (including the power-limited ones) are then only verifying the integrity of the chain. That makes such a DLT only partly decentralized.

- The other problem is that a powerful adversary may be able to overcome the challenge of matching the computing power of a certain WSN: the *raison d'être* of the PoW is to avoid that an adversary can take control of the consensus process. Since most genuine participants in a WSN (the SNs) are typically power-constrained, if the network does not have a very large number of nodes contributing to the chain, it is not far-fetched that a powerful adversary, or a collusion of adversaries, could overcome the combined computational power of all the genuine mining nodes.

These severe limitations led to the research of new consensus mechanisms and alternatives to PoW have been developed. Some popular ones, adopted by the industry, include:

- Proof-of-stake (PoS): In proof-of-stake, the node that adds the next block to the Blockchain is chosen based on its stake, or ownership, of the tokens on the network. The more tokens a node owns, the more likely it is to be chosen to add the next block. In 2022, Ethereum, the 2nd most popular cryptocurrency after BitCoin, switched from proof-of-work to proof-of-stake, citing improved energy efficiency as the main reason for the change. The reported reduction in energy consumption was 99. 98%, but the move led to some disadvantages (less transactions per second; higher fees; etc.) [106].
- Delegated proof-of-stake (DPoS): This is a variant of proof-of-stake in which token holders can delegate their voting power to other nodes, which are then responsible for adding new blocks to the Blockchain. It is used by EOS, Steem, BitShares, ... [107, 108]
- Practical Byzantine fault tolerance: This consensus mechanism is designed to tolerate faulty or malicious nodes on the network. It uses a voting system to reach consensus on the state

of Blockchain, and requires a certain number of nodes to agree on the next block before it can be added to Blockchain. This mechanism is used by IBM's *Hyperledger* [109].

There is a trade-off made when switching from Proof of Work (PoW) to Proof of Stake (PoS) or other consensus mechanisms, in the form of a decrease in decentralization. In some cases, the benefits of a particular consensus mechanism may outweigh the potential loss of decentralization, but it depends on the priorities of the system. [110] use PoS and DPoS as examples of poor decentralization and claim that today there is still a gap between achieving good decentralization in the consensus protocol and not relying on a trusted third party.

There are other consensus mechanisms investigated for future use in DLTs, and new mechanisms are being developed all the time. Some other examples include proof-of-activity, proof-of-importance, proof-of-capacity, proof-of-elapsed-time, ... We refer the readers to these recent reviews: [111] for a multipoint taxonomy,[112] for the comparison of energy consumption, [113] for the scalability review, and [114] for their suitability study.

DLT based lightweight authentication systems

Most proposed solutions so far propose to use DLT in the context of IoT for several reasons (decentralization, interoperability, openness, resiliency, ...), but it doesn't seem that enablement of resource-limited devices is one of them. For example, in [115], the authors present a decentralized IoT system, using Ethereum as their public Blockchain, and meeting a list of requested security requirements. But this system is using ECDSA for authenticating each message, it is not more efficient (in fact, it is probably less efficient) than the asymmetric systems presented in 2.4.1. This is also the case for the papers covered in their "related work" section.

[116] addresses the opportunity of DLT/Blockchain converging with IoT ("Blockchain-IoT") and presents some taxonomies regarding types of DLTs and of validation process. The authors list some benefits of using DLT/Blockchain to solve some of the shortcomings of legacy IoT model approaches, including centralization, scalability, interoperability, adaptability, coordination, and more. They discuss some adoption considerations and challenges. The authors emphasize that most IoT devices lack the resources to handle the needs of directly being part

of the Blockchain. So, even if a system is claiming to use DLTs, these devices will not be able to handle the required computations, storage of the ledger or even the usually increased number of message transmissions (in comparison to legacy protocols).

Authors of [117] call the state of Blockchain based authentication systems as being in "exploratory stage". Some aspects are analyzed, but usually leaving aside considerations related to the communication cost itself. They too remark that the limitations of the IoT equipment make it impossible to meet the requirements of the Blockchain, and that's why researchers have to make use of gateways between IoT devices and Blockchain models. They present a multi-WSN authentication scheme based on a hybrid private and public Blockchain model. To handle the challenge of the constrained resources of the "ordinary nodes", they introduce in their system some *cluster head nodes* that handle the additional computation, storage and transmissions.

Although it may not be practical to implement DLT directly on resource-constrained sensors due to their limited computational and storage capabilities, it can still be useful for WSNs by being implemented on the more powerful gateways or in the cloud. This allows DLT to provide the benefits mentioned above without overburdening the edge devices.

IOTA Tangle

To overcome the limitations and challenges of Blockchains for WSNs, some have proposed to use IOTA Tangle. IOTA Tangle is the name of the Directed Acyclic Graph (DAG) data structure used by the IOTA distributed ledger technology to support transactions and smart contracts. The Tangle is an alternative to the traditional Blockchain data structure, which uses blocks to store transactions. Instead of using blocks, the Tangle allows transactions to be added directly to the ledger, which enables fast, low-cost transactions and makes it well-suited for use in the Internet of Things (IoT). In this model, each new transaction is validating two previously non-validated transactions. This means that there is no need for "miners" or "validators" like in Blockchains, since each node that adds a transaction participates in the ledger validation.

There are several important differences between IOTA and traditional Blockchains, for

example, better scalability, no transaction fees, ... But one thing makes the original IOTA not fully decentralized: It utilizes a central service called "Coordinator" (*Coo*) to help secure the network by preventing double spending. This is needed since the network does not have enough computational power for hashes calculation to secure itself.

IOTA's reliance on a central coordinator service, run by the IOTA Foundation, means that it is not fully decentralized. This could potentially allow the IOTA Foundation to manipulate transaction priorities and is also a limiting factor to the scalability of the network. However, this centralization is only intended to be a temporary measure.

The use of the Coo in IOTA also introduces a potential single point of failure, which goes against the basic idea of having a decentralized mechanism. In the context of DLT, this practical inability to be at once scalable, secure, and decentralized is called the "scalability trilemma", and IOTA is not an exception to it.

Still, the IOTA Foundation plans to remove the Coordinators from the IOTA network, an effort called "Coordicide", also known as IOTA V2. The ultimate goal of Coordicide is to create a fully decentralized and self-sustaining IOTA network that is able to secure itself without the need for a bootstrap centralized coordinator, but also brings more scalability, Sybil protection, smart contracts, and support for digital assets [118]. There are still some open research questions around IOTA 2.0, but is one of the most promising fast, cheap, and scalable DLT solutions.

IOTA Tangle based message authentication protocols

The IOTA foundation develops some cryptographic protocol frameworks around IOTA. IOTA Stream [119], based on the now-deprecated Masked Authenticated Messaging (MAM) [120], allows for secure and private communication of data streams over the Tangle, by encrypting and authenticating messages using a unique message key, derived from a seed. This allows for data streams to be selectively shared and accessed by authorized parties, while keeping the data private and secure from unauthorized access.

LASII [121] is an IOTA-based authentication scheme designed for IoT devices and services. The scheme uses the concept of "virtual zones" or "bubbles of trust" introduced by [115], where

each device can only communicate with other devices in the same zone. It uses the Masked Authenticated Messaging (MAM) protocol to create extensible authenticated zones in the IoT environment.

[122] propose a WSN architecture that avoids centralization of IoT storage in the cloud by storing device identities in IOTA. Their model provides data preservation and security by using the MAM protocol to store sensor information.

Although IOTA Streams has been improved to be more efficient, with a reduced memory size and less processing time required [123], it is still not suitable for all types of IoT devices due to constraints such as limited computational capacity, memory and energy consumption.

These limitations have been put forward by [124] and they designed L2Sec, a cryptographic protocol for secure data exchange over the IOTA Tangle, suitable for constrained IoT devices. Instead of Streams, they designed their protocol for sensors data model running on a micro-controller unit. L2Sec uses lightweight cryptographic protocols (EdDSA with Blake2b hashing for message integrity, ECDSA for authentication, XSalsa20 for encryption with Poly1305 MAC).

It is worth noting that while L2Sec employs asymmetric cryptography, data encryption is currently reliant on pre-shared keys. However, the authors have proposed adding the capability for data encryption using asymmetric cryptography as a potential improvement. They provide a full implementation using an evaluation board based on the ARM Cortex-M4 and a low-energy WiFi module. To achieve improved robustness, they also propose the adoption of hardware secure elements and using them in conjunction with a Trusted Execution Environment.

Paper	**Contribution**	**Consensus algorithm**	**Validation**	**Type and role of the DLT**	**Node authentication method**
[115]	"Secure virtual zones" where elements can identify each other	PoS	Implementation; informal security analysis	Public blockchain; Ethereum; Blockchain used for ledger	ECDSA for node authentication
[117]	Multi-WSN authentication scheme for IoT; use of *cluster head* nodes to offload computation from the *ordinary* nodes	Not defined	Informal security analysis	Hybrid private and public blockchain; public for *head node* registration; private for *ordinary* node	ECDSA for *head node* authentication; Implied trust of the registration data from the *ordinary node* to the *head node*
[121]	Extends the "Bubble of trust" from [115] using MAM channels	Tangle	PoC, implemented based on IOTA v1; message exchanges validated with AVISPA	IOTA; Authenticated zones extension	Masked Authenticated Messaging (MAM); implied trust inside each authenticated zone ("pre-authentication phase")
[122]	Lightweight identity authentication scheme; secured data storage in IOTA	Lightweight PoW; Tangle	Informal security analysis	MAM protocol; IOTA as ledger	MAM for data security and preservation and Cluster Head identification; MQTT for node to Cluster Head communication
[124]	Secure data exchange protocol over IOTA	Tangle	Performance analysis on PoC	IOTA as decentralized ledger	EdDSA for integrity signature; ECDSA for authentication signature; XSalsa20 for confidentiality; IOTA Chrysalis for message encapsulation

Table 2.5: DLT-based authentication systems for WSN nodes

2.5 Facilitating the Selection and Adoption of WSN Authentication

As discussed in Section 2.2, the diverse nature of authentication solutions requires a structured approach to comparison. Each solution has its unique characteristics and trade-offs, making direct comparison challenging. To address this, we have organized the solutions into different families based on their underlying principles and mechanisms. By understanding the nuances within each family, we can better evaluate their suitability for different application scenarios and make informed decisions regarding their adoption.

To assist readers in selecting among the various families of solutions, we provide a comparative summary of their strengths in Figure 2.1. This informal representation presents the general characteristics of each family and should be viewed as a preliminary guide rather than an absolute reference.

It uses six criteria: Device Cost, Security, Scalability, Logistics Complexity, Memory Requirement, and Computational Cost. The outermost points represent the best results for each family.

The Software Cryptography-based family (purple polygon) has relatively good results in all criteria, usually seen as a good middle ground, with the lower deployment cost (logistics and device cost). The Cryptography-based family with Hardware accelerators has lower device costs and better computational costs than the software-based family, but its security and scalability are not as strong, as represented by the red polygon. The Radio Frequency Fingerprinting family (green polygon) excels in Energy and Memory requirements, but its security and scalability results are generally lower or more challenging. The PUF-based family (cyan polygon) has a higher Device Cost, but relatively can provide higher security for a lower computational costs. Lastly, the Blockchain-based family (orange polygon) is best in Scalability, but its results are relatively weak in most of the other criteria used in this graph.

Within each family of authentication schemes, individual solutions can prioritize different characteristics, as highlighted in the key elements of each proposal discussed in the relevant subsection of Section 2.4. Therefore, the choice of a specific solution should carefully consider

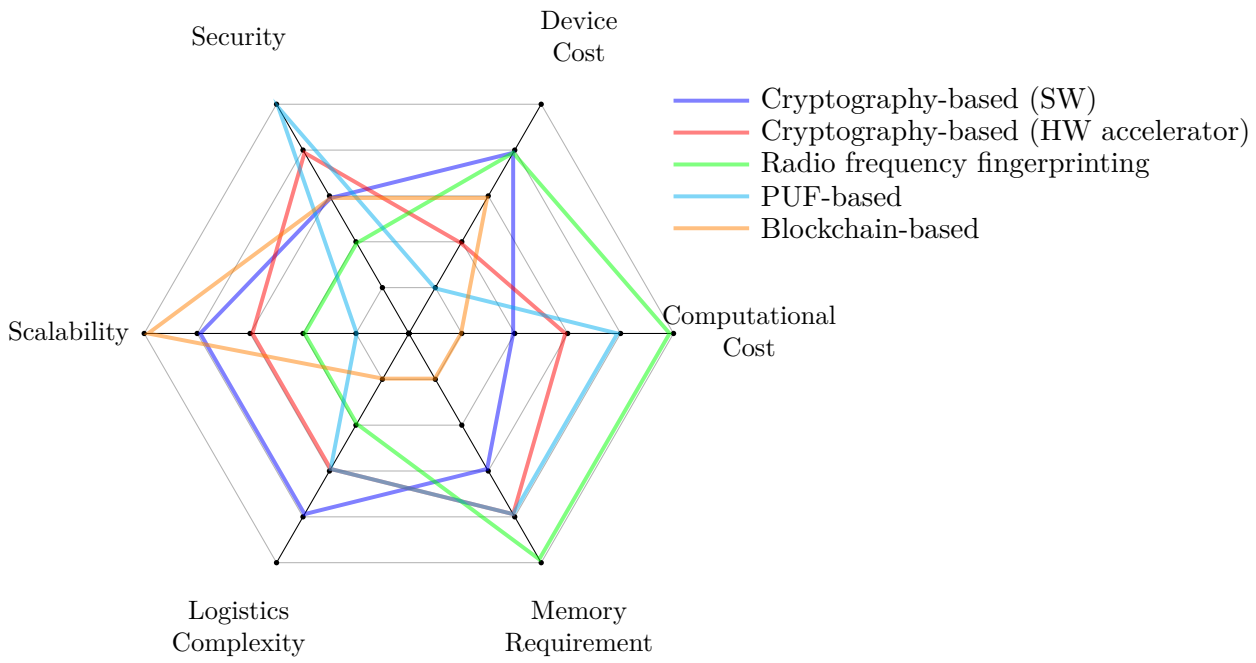


Figure 2.1: Characteristics by solutions families
(best results are represented by the outermost points)

the project needs, the level of maturity of the scheme, the scalability of the solution, its compatibility with existing systems, support of the community, and other relevant factors.

When it comes to adoption, manufacturers are typically reluctant to invest in a solution without the assurance of its standardization. In a global market, it's logical for them to prioritize investment in technologies with the highest return on investment (ROI). Therefore, while ongoing research in the domain is beneficial for both science and industry, the need for standardization is crucial to facilitate the integration of security into products across wide markets. As discussed in Section 2.1, we are currently witnessing the first concrete measures taken by governments, and there is hope for an eventual synchronization to establish common standards.

Other factors play a crucial role in the adoption of a solution. Projects like LoRaWAN have succeeded in rapid adoption because they prioritized the early development of open standards, received commercial support, and addressed logistical issues such as regulatory compliance (e.g., by operating in unlicensed spectrum bands) as part of their proposed solution. Additionally, the absence of patents has played a significant role in the adoption of the authenticated

encryption mode AES-GCM over the patented OCB.

2.6 Conclusion

The field of message authentication schemes for power-constrained WSNs is rapidly evolving, driven by the increasing deployment of WSNs in various applications. This chapter presented an overview of the trends and challenges in this field, including asymmetric, symmetric, PUF, RFF and DLT-based solutions.

We have discussed the advantages and limitations of each approach and highlighted the key challenges that must be addressed to ensure the security and reliability of WSNs. These challenges include the need for efficient and lightweight authentication schemes that can operate with limited resources, the development of standardized protocols and architectures, and the integration of advanced security features such as privacy preservation and secure key management.

Although much progress has been made in this area, there is still much work to be done. This thesis takes a step further by proposing a practical cross-layer radio frequency-based authentication scheme for Internet of Things (IoT), which is presented in the next chapter. The proposed scheme aims to address some of the limitations and open challenges discussed in this review, and its design and evaluation are detailed in Chapter 3.

Chapter 3

Practical Cross-Layer Radio Frequency-Based Authentication Scheme for Internet of Things

Contents

3.1 Introduction

In this chapter, we propose a new cross-layer approach that combines existing authentication protocols and Physical Layer Radio Frequency Fingerprinting (RFF) technologies to provide hybrid authentication mechanisms that are practically proven efficient in the field. Even though several Radio Frequency Fingerprinting methods have been proposed so far, as a support for multi-factor authentication or even on their own, practical solutions are still a challenge. The accuracy results achieved with even the best systems using expensive equipment are still not sufficient on real-life systems. Our approach proposes a hybrid protocol that can save energy and computation time on the IoT devices side, proportionally to the accuracy of the Radio

Frequency Fingerprinting used, which has a measurable benefit while maintaining an acceptable security level. We implemented a full system operating in real time and achieved an accuracy of 99.8% for the additional cost of energy, leading to a decrease of only ~20% in battery life.

The organization of this chapter is as follows. Section 3.2 presents the challenges that are seen today for lightweight authentication of IoT devices in the context of radio communications. Section 3.3 introduces a wireless sensors network system and defines the scope of our research. We also define the threat model used for this system. In Section 3.4, we present our hybrid scheme, with the details of the communication protocol. Section 3.5 presents an informal evaluation that addresses each security property of the threat model. In Section 3.6, we present an implementation of the system, with a review of its accuracy and a precise evaluation of the energy performance. Finally, Section 3.7 concludes the chapter.

3.2 Definitions and Related Work

In the context described in the previous chapters, one of the looming concerns is the definition and adoption of lightweight IoT security mechanisms.

One of the difficulties is that IoT devices are often designed to accomplish very specific and limited tasks, but function as part of complex ecosystems. Their security depends on their ability to defend against targeted attacks in challenging and ever-changing environments, despite their limited resources. The lack of resources is, according to Curran [125], the reason why "the adoption of security support ecosystems, such as large databases of malware signatures, is impractical".

Therefore, a *pragmatic* approach to IoT security is necessary, one that balances security goals with resource constraints. This requires a deeper understanding of key concepts that will shape our solution: "good enough security", "lightweight cryptography", and "radio frequency fingerprinting". By exploring these elements, we can develop a hybrid authentication mechanism that effectively addresses the challenges of IoT security.

Good Enough Security

Full-fledged security protocols are not always required to achieve a level of "good enough

security", and as Sandhu stated, "Good enough always beats perfect" [126]. Not all IoT systems require the same strength of protection mechanisms and the same procedures to be considered secure enough. However, by just proposing *"complete"* security solutions, these become too heavy and are not adopted. For example, in the case of sensors delivering non-confidential information, encryption may not always be required. The additional complexity and increased cost due to the requirements for implementing strong encryption support are obstacles to the adoption by device manufacturers. As a general approach to system security, it is important to create an appropriate threat model in order to define the security goals. The security mechanisms implemented will be considered appropriate if they are expected to be effective in addressing those goals.

Lightweight Cryptography

In this regard, different approaches for lightweight message authentication have been proposed in a resource-starving environment. Symmetric cryptography solutions include Message Authentication Code (MAC) based on shared secret using block cipher constructions (like CMAC) and hash-based MAC using secure hash function with adequate construction (like HMAC-SHA-256), e.g., [127, 128]. However, even if these methods are usually more lightweight than asymmetric cryptography solutions, they are still computationally demanding. In their survey of symmetric lightweight algorithms [43], Biryukov and Perrin call specialized algorithms providing one function with high performance *ultra-lightweight*. This is the case of the lightweight message authentication codes SipHash [129] and Chaskey [130].

Radio Frequency Fingerprinting

Radio frequency fingerprinting (RFF) is the identification of a wireless transmitter based on the analysis of the signal received by a receiver. This identification is based on hardware differences between the transmitters (e.g., tiny imperfections due to the manufacturing process) [131] and on channel characteristics of the transmissions [132, 133].

Device RF Fingerprinting has been proposed for a long time to solve the problem of node forgery and impersonation, which "constitutes one major security threat facing wireless networks" [134]. As part of their survey on IoT authentication protocols, Ferrag et al. have reviewed 23 protocols that fully or partially addressing the impersonation (*spoofing*) attack [135];

among all these protocols, only [136] presented "a plan of cross-layer authentication using the hardware RF fingerprint to identify whether messages are from the same wireless device."

Cryptography-based security requires resources; in most cases, this is computing power. One of the main advantages of a practical protocol that uses RFF for the authentication of the IoT device is that it would not require computation on the lightweight (and sometimes battery-powered) transmitting device/sensor since the identification computation is done mainly on the receiver side, which is usually less power constrained.

However, such an efficient system is not easy to achieve. In the domain of device authentication, radio transmitter fingerprinting systems were proposed for more than two decades (e.g., [137, 138, 139]). The authors of [140] provided results of RFF comparisons based on multiple features and of several Machine Learning classification algorithms and reported that they achieved "an overall accuracy higher than 80%, which can be suitable to support multiauthentication of IoT devices". However, this is not enough to be a viable first (let alone *single*) factor of authentication. Some more recent researches look very promising, even if still in "conceptual development" stage, and the ones reporting to achieve 99% accuracy were so far only under ideal conditions [100]. In more realistic environments, accuracy degrades rapidly. For instance, lower Signal-to-Noise Ratio (SNR) and factors like Line-of-Sight (LOS) vs. Non-Line-of-Sight (NLOS) directly affect the precision of the identification systems [141, 142].

Therefore our challenge is to make use of the existing schemes, despite their relative low accuracy, and still benefit from the fact that they do not need computation from the transmitter, while at the same time, it does not compromise the system security.

3.3 Network System and Threat Model

In order to adhere with the "good-enough-security" principle as described in Section 3.2, we must define the set of required security properties that will respond to the security threat model of our system. We use a concrete context, and as an example of embodiment, we define our system as similar with current implementations of smart (or precision) agriculture systems. Examples of properties measured in this context are numerous, including temperature,

humidity, acoustic, proximity, acidity, motion, etc. These measurements are used to evaluate weather conditions, soil quality, pathogens, insect pest detection, crop's growth, drug residues, heavy metal, etc.

3.3.1 Wireless Sensor Network System Definition

We define the system as a wireless sensor network, consisting of wireless, spatially distributed, fixed-location sensor nodes, and a gateway. The purpose of the sensors is to transmit real-time measurements to a command and control (C&C) center, via the gateway (Figure 3.1). The sensors are battery-powered and should be as cheap as possible in order to be deployable in numbers. As such, they should implement only the minimal subset of required functionalities.

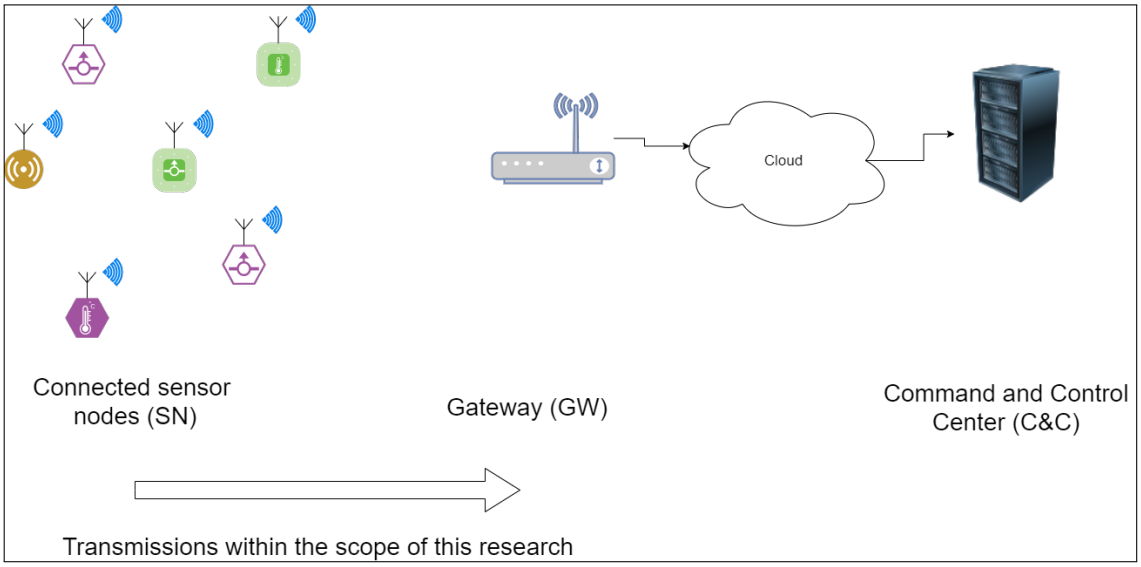


Figure 3.1: System model description.

We present our method using the case of a wireless sensor network (WSN) comprising unattended sensor nodes (SN) transmitting short messages to a gateway (GW). Each SN consists of at least a battery powered micro-controller, a transceiver and some sensors. Each sensor is initialized with an identifier (ID) and a pre-shared key (PSK), which is a secret value used for cryptographic authentication before being deployed. Both the ID and the PSK are known by the GW.

We assume that for such a system, only short messages are of interest, and therefore, the protocol is optimized for this matter.

3.3.2 Threat Model

There are a few security assumptions for this system:

- Key provisioning. We assume that a pre-shared key (PSK) has been serialized in each node and is known only by the gateway. The method for this provisioning and physical attacks on the nodes are out of the scope of this research.

- Physical attacks. Depending on the system, the nodes can be hardened or not accessible. Physical attacks, e.g., extracting the PSK from the node, are not in scope of this research and not part of its threat model.

- Practically infeasibility of purposely flipping certain bits by jamming. Jamming is usually aimed at radio signals to disrupt the reception of the original transmission by a receiver. In theory, it would be possible to purposely flip some bits of the transaction, but we consider such an attack *practically* unfeasible [143]. This also means that we can assume the equivalence of the authentication of a RF message source to the authentication of its content.

- Robustness of the hash function used. For our analysis of the scheme, we will consider the cryptographic hash function to be computationally secure against first pre-images, second pre-images and collisions.

The threats taken into account for this system, and thoroughly analysed in Section 3.5, are:

- Message forgery

- Message replay attack

- Message origin impersonation

- Man-in-the-middle attack

3.4 Hybrid Cross-Layer Authentication Protocol Scheme

3.4.1 Overview of the Scheme

In this work, we propose a hybrid cross-layer authentication protocol that tackles the aforementioned challenges, namely reducing the energy consumption of low-resource devices, by leveraging known RFF technologies together with known lightweight cryptographic authentication algorithms. The objective is to achieve this goal even if the actual RFF techniques as used today are not yet on par with the security level of other authentication techniques. The authentication of a single message may be achieved through the RFF or through cryptographic authentication. Using both approaches in the same protocol, not as multifactors of the authentication system but as complementary methods, we create a Hybrid Authentication mechanism: each message sent by SN to GW is authenticated *or* by RFF *or* by cryptographic authentication (in the case the RFF message authentication failed).

Our main goal is to transmit the messages in an authenticated way, with minimal impact on the power consumption of the SN. We define and propose using a hybrid cross-layer authentication protocol. This model will use a standard cryptography-based authentication mechanism, along with an RFF identification system, to achieve a lightweight authenticated communication.

3.4.2 Radio Frequency Fingerprinting Calibration

Like biometric-based authentication systems, RFF-based authentication systems need some metrics to be compared between them and calibrated. The probabilities of incorrect outcomes of an authentication session are known as False Reject Rate (FRR), sometimes called Insult Rate, and False Accept Rate (FAR), also called Fraud Rate. False Reject means that a rightful client was rejected during the authentication process (e.g., Alice was not authenticated as Alice). False Accept means that someone other than the rightful client was authenticated (e.g., Bob impersonated Alice). Biometric systems (or RFF systems) can be calibrated to be more lenient in authentication, thereby lowering the FRR. However, the negative side effect is then that the FAR will also increase. In contrast, raising the threshold so that the FAR

is reduced yields a higher FRR. The Equal Error Rate (EER) is obtained by calibrating the threshold so that FAR is equal to FRR. EER is usually used to compare the effectiveness of different systems. Still, this approach is adequate for comparison of mechanisms used for authentication. Rather than using an EER-based calibration, the RFF part of our method is calibrated using an accepted and sufficiently low FAR, based on the level of authenticity requested. As we have seen, this will yield a much higher FRR. In order to deal securely with these false rejects, the receiving side (the Authenticator) will "fall back" to our secondary authentication system.

In the following embodiment of the scheme, we use a lightweight Challenge—Response Algorithm, based on a keyed-hash using the secret stored in the SN as key. This will be detailed in the next sections.

3.4.3 Challenge–Response Authentication and Message Authentication

We first describe how a system answering our requirements and using some legacy handshake protocols would be built. In Figure 3.2, a legacy three-way handshake Challenge Response Authentication Protocol session is shown. As described in Section 3.3.1, both sides share a pre-shared key (PSK). When the client requests to be authenticated, the server (authenticator) sends it a challenge. The challenge itself is a cryptographic nonce, i.e., an arbitrary number used only once. It is generated using a deterministic random bit generator (DRBG) seeded with a monotonic counter to avoid a repetition of sequence.

Both sides calculate the response, which is the cryptographic hash of the challenge concatenated with the PSK. The client sends its response, and the server compares it to its calculation. If they match, the authentication is successful. Several well-known authentication protocols are based on this model, e.g., CHAP [144], EAP [145], and HOTP [146].

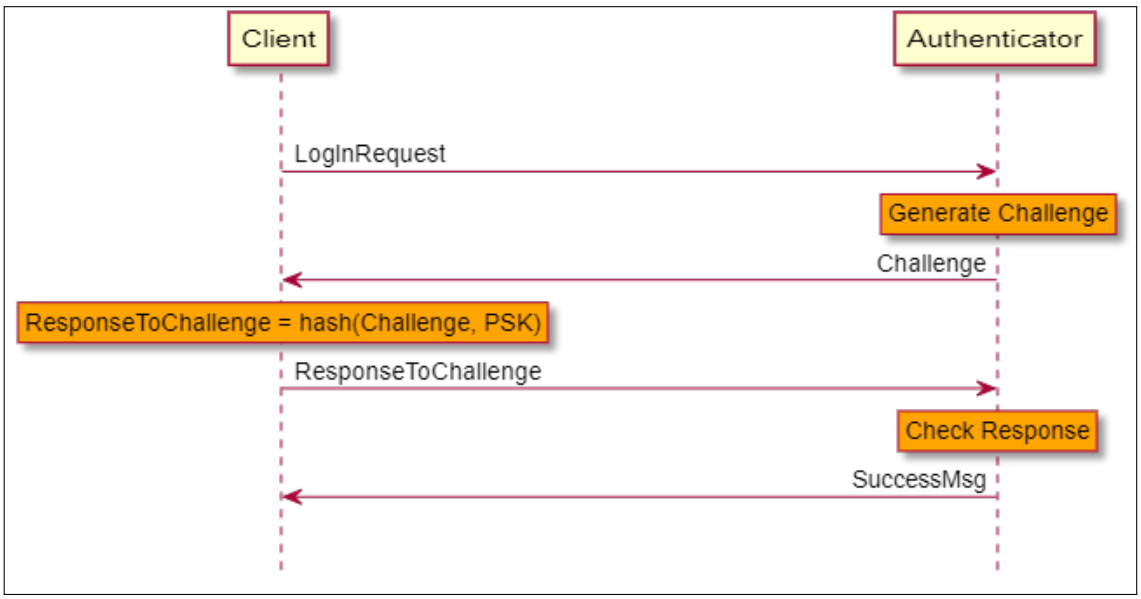


Figure 3.2: Legacy Challenge–Response authentication session.

As this type of protocol is based on some cryptographic hash, for short messages, we can also add information to be sent (e.g., the measurements taken by the sensor) as part of the transaction. Since each transaction and each computation requires time and energy, binding the message with the handshake will minimize the cost of Node Authentication and Message Authentication, in the case the Node (re)-Authentication is needed.

In order to bind the session to the client, a "session token" or "cookie" can be used to avoid some attacks based on stealing the session. This is common practice to avoid session hijacking attacks [147]. In order to be used as a stateless protocol, i.e., no session information is kept by the server, the system may use a token/cookie that binds some secret known by the client (like some cryptographic key) with the cookie, which may be encrypted by the server key.

In the following sections, we refer to such a protocol based on a cryptographic challenge-response handshake bind to the message and its authentication code as *MAC-only* solution (in contrast with our hybrid solution).

3.4.4 The Benefit of RFF Combined with a Challenge–Response Authentication

A legacy solution such as the one presented in the previous section leads to computation on both ends. In the context of lightweight IoT, we especially try to reduce the computation from the client side. By binding the session token with the perceived RF features from the server side, the client side does not need any new computation to prove its identity, since the binding is transparent and part of the communication itself, like the human voice of a known person.

In the case of sensors sending non-confidential messages, all the security properties may not be needed, and we want to check the authenticity of the sender but may also want to reduce the power consumption and complexity induced by cryptographic computation, handling of secret keys and sessions. As seen in Section 3.2, this may be considered "good enough security" and is more likely to be integrated into practical solutions.

Our approach has the advantage that the RF fingerprint can be used to authenticate the message to the identity of the sender without the need for cryptographic primitives. It makes the communication of short messages resistant against impersonation attacks ("spoofing attacks") with a minimum effort by the sender.

It has also the advantage of being forward-compatible with future advancements in the field of RFF: since the protocol is agnostic to the RFF method used, it is possible to replace the RFF part of the system by a more evolved one, without having to replace or even update the deployed sensors, since the RFF is integrally implemented on the receiver side. Furthermore, even if the system as a whole is made more power efficient by using a better RFF system, it can make a *practical* use of actual RFF systems, even with their relatively high EER.

3.4.5 Hybrid Authenticated Lightweight Communication

For a Hybrid Authenticated Lightweight communication based on RFF, we consider the case where a single transaction consists of the authenticated delivery of a message including at least some basic functional parts: the SN ID and the payload (for example, sensor values, like temperature, humidity, etc.).

We have to distinguish between the case where RFF was accepted as matching the stored RFF relevant to the SN ID and the case where it is not. As explained before, a system where it is possible to authenticate the messages without the need for some computation and additional information from the SN is preferred in order to reduce the SN energy consumption.

Successful Authentication by RFF

In Figure 3.3, the message was successfully authenticated by its RFF alone. No other message is needed from the SN.

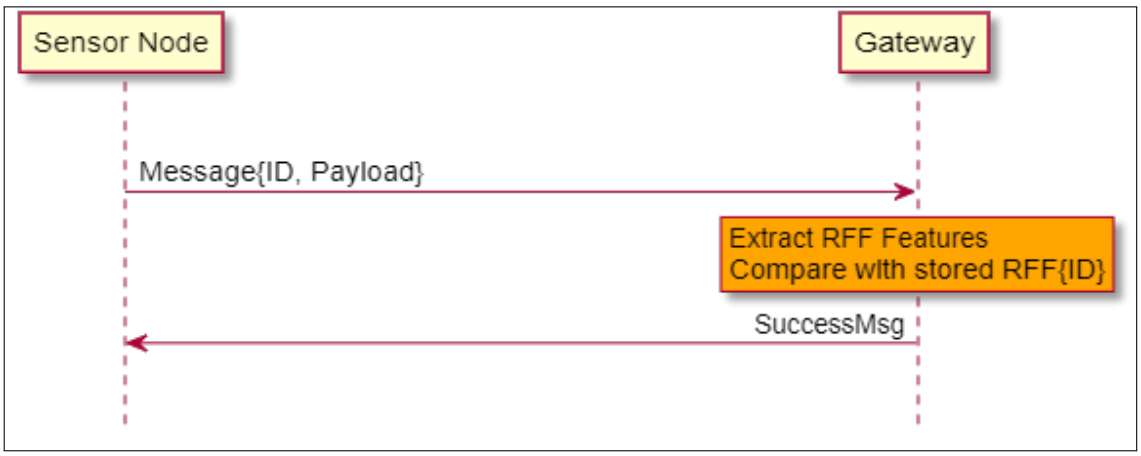


Figure 3.3: Successful authentication by RFF.

Unsuccessful Authentication by RFF and Fallback to Cryptographic Primitive

In Figure 3.4, the message was not successfully authenticated by its RFF. This can happen in several cases: the SN has never been authenticated before, and so, the GW never stored its RFF; the RFF was not matched; and of course, it may happen in the case of an adversarial (malicious) transmission.

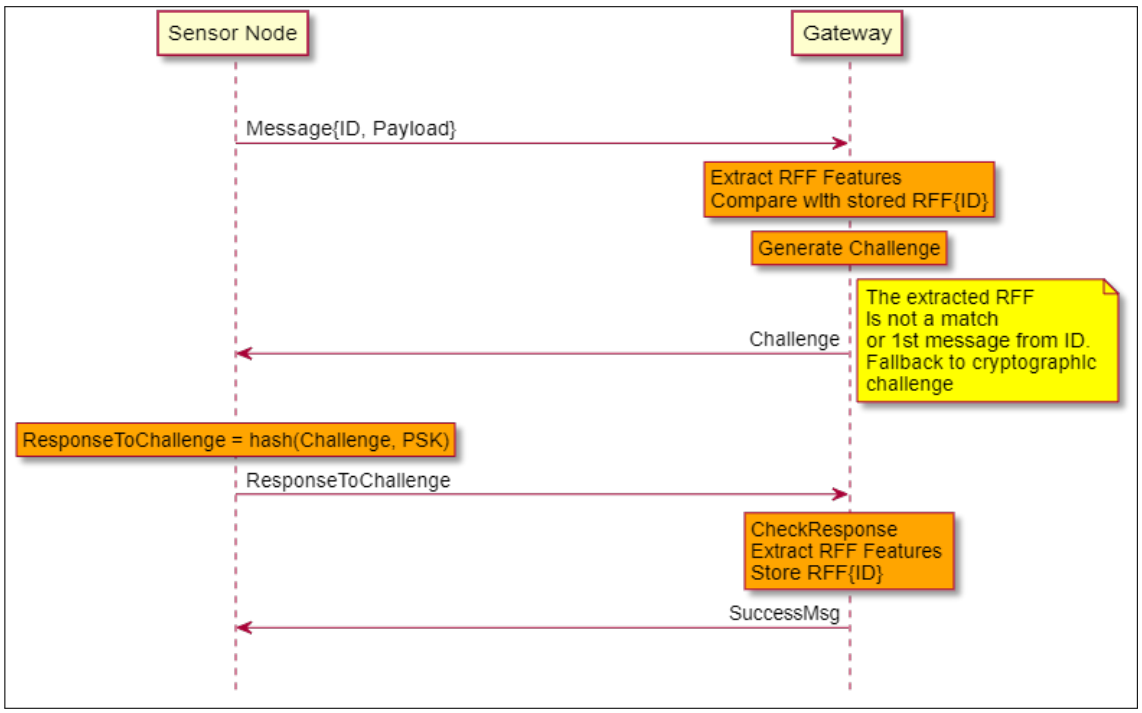


Figure 3.4: Unsuccessful Authentication by RFF and fallback to CHAP.

In order to differentiate between these cases, the GW will issue a challenge to the SN (usually, just a random number), as explained in Section 3.4.3.

The SN has to answer the challenge by sending the keyed-hash based message authentication code (MAC) of the challenge. The secret used as the key of the MAC is the shared secret as defined at the beginning of Section 3.4. In order to securely couple the message to the authentication part, the *ResponseToChallenge* message also includes the ID, the payload and the MAC calculated on all the fields to authenticate: challenge, ID and payload, with the Secret as the key.

If the *ResponseToChallenge* is successfully verified by the GW, the RFF features are extracted and the RFF is stored for this specific ID (RFF{ID} in the Figure 3.4). This scenario is a valid one even if not in the case of a malicious attack. It can happen even for static sensors for example, if the environment changes. This demonstrates the self-recovery property of the protocol.

3.5 Informal Security Evaluation

In order to evaluate that the security requirements as defined in Section 3.3.2 have been addressed, we discuss each threat and review how it is mitigated with our system for both MAC and RFF.

3.5.1 Message Forgery

Message forgery would mean that an attack is able to forge a single message and have it accepted by the gateway as genuine. This risk does not apply in our system for a single message, since, when using MAC, each message is authenticated, and we assume the equivalence of the authentication of a RF message source to the authentication of its content (see Section 3.3.2).

3.5.2 Message Replay

Message replay attack would mean that an attacker could resend a previous message sent by a sensor to the gateway. This risk is mitigated by the fact that using MAC, the message is coming along with a challenge–response authentication, and this does not allow a message to be replayed out of order. Furthermore, using the RFF, the gateway would detect that the message is not coming from the same source.

3.5.3 Message Source Impersonation

An attacker sending a single message in place of the GW is of no interest in our context (except for the case covered in the next section). Furthermore, regarding a sensor impersonation, in our system, a direct impersonation is not possible. Based on the PSK saved in the node, it would be computationally impossible for an attacker to fake the challenge–response based on the MAC (which would be equivalent to breaking the cryptographic hash function). Furthermore, the RFF is used to authenticate the sensor. As discussed, the configuration of the system has to be done so that the FAR is low enough to meet the security requirements of the given system, as is the case for any biometric authentication system. However, we discuss in the next section how a certain setup could lead to a sensor impersonation through an elaborate

attack, if not mitigated.

3.5.4 Man-in-the-Middle Attack

A man-in-the-middle (MITM) attack consists of an intruder relaying messages between both parties of a communication. In our case, an intruder capable of intercepting the messages between the SN and the GW during the CHAP cannot alter the messages, since the GW would detect that the response is not the one it calculated. However, if the intruder is capable of sending the messages in place of the SN, and of blocking the SN messages themselves from reaching the GW, then its RFF would be the one recorded by the GW as the valid one in place of the legitimate SN. The way to achieve this in our context would be for an attacker to jam the messages in a way that he can record the message without letting them reach their destination, and resend them as is from its own transceiver. This attack is similar to others found with systems sending one-way authenticated messages based on rolling codes. This was demonstrated in the so-called "RollJam" attack [148].

The "cryptographic way" to mitigate it is to cryptographically authenticate each message. However, this goes against the very essence of our requirements, which is to save as much of the computation as possible on the sensor side. We therefore would consider other system-level mitigations in place: e.g., first, CHAP executed in a safe environment to create an RFF baseline and/or use of an intrusion detection system (IDS) to detect signal jamming in the area of the system. Different implementations of jamming detection techniques have been published and analyzed with great success [149, 150, 151]. This would solve the MITM attack described here, but it is not in the scope of this work.

3.5.5 Security Advantage of the Hybrid Approach

A significant advantage of our method is the synergy of some of the methods used, in a flexible manner. Enforcing at the gateway side that one in every n message will request a full reauthentication challenge and check for methods at the same time increases the security bar significantly by mixing the advantages of both approaches. The periodicity of the forced reauthentication is flexible and is, of course, a trade-off between cost and security. For example,

to defend against anti-replay attacks, the cryptographic approach assumes that the node key was not extracted or lost and uses a nonce for the challenge, so that even if an attacker recorded all the previous *Challenge* and *ResponseToChallenge* messages, they cannot replay a valid one for a new challenge. This is a type of intrusion detection and prevention system. However, by having the same property of intrusion detection implemented by a different manner (RFF) and having the system check that both approaches are used at least once every n messages, we gain detection capability in the case some attacks find a way to bypass one of our protections, which gives more robustness to the system.

3.6 Scheme Experiment and Evaluations

In this section, we present a full implementation of the system, an accuracy evaluation, and an energy performance evaluation.

3.6.1 Evaluation System Description

From an end-to-end perspective, the evaluation system consists of several sensor nodes communicating with a computer. One of the nodes is a legit SN, which sends "sensitive" messages, while the others are "rogue" nodes, which send similar messages. The role of the computer ("GW", since it plays the role of the GW in our protocol) is to differentiate between the legit one and the rogue ones. The legit SN stays at a fixed location in an outdoor environment, as is the case for numerous scenarios, as explained in Section 3.3. The rogue nodes may be fixed or mobile.

Original Real-Time RFF Authentication System

In an effort to experiment and evaluate end-to-end the whole scheme on a *live* system, we implemented an authentication system based on Received Signal Strength Indicator (RSSI) values from different anchors. The choice was driven by the fact that most, if not all, modern RFF systems, such as the ones cited in Section 3.2, require post-processing.

We present here an original evaluation system fully working in real-time, inspired by pre-

vious works from different domains: using RSSI measurements to evaluate the location of a transmitter is a common technique based on radio triangulation, for indoor location tracking or for geolocation in GPS-degraded environments [152]; leveraging RSSI-based location fingerprinting as part of authentication schemes has been proposed in the past for smartphones or in the context of WLAN [153, 154]. It has also been proposed as a second factor of authentication, in the form of a proximity check [155], and for anti-spoofing mechanisms based on physical layers in V2X Networks [156]. Building on top of these methods to create a lightweight RFF-based authentication system, we were able to create an end-to-end evaluation system close to a real-life setup.

It should be noted that switching the RFF measurement part of this system to use another RFF method is straightforward: the sensor implementation, the gateway algorithms and interface control messages remain unchanged, and only the radio-based authentication differs between the systems. Furthermore, the RF-based authentication part does not affect the implementation of the sensors themselves, since it is only implemented in the GW (and optionally some processing can be carried out by the C&C). This is in itself one of the advantages of our scheme, as explained above.

As a side note, the authors would like to remark that the RSSI-triangulation based RFF mechanism used in this evaluation was developed as a simple one for a close-to-real-world evaluation of the full system end-to-end, including energy consumption comparisons. However, even if it was not the original intention of the authors, it turned out that this 'simple' system is quite efficient and rather precise. Anyway, we would clearly recommend for any manufacturer to evaluate more evolved and mature RFF systems for production, as our system was not intended to be production ready.

Selected Cryptographic Primitive

As the Message Authentication Code (MAC) of the cryptographic part of our system, instead of the HMAC construction using a generic hash algorithm like SHA256, we adopted the Chaskey keyed-hash algorithm [130], designed to be fast on short messages, as proposed for lightweight message authentication code by [43] (see Section 3.2). The Chaskey-12 variant was used, as

recently standardized by ISO/IEC [157], which has a very small memory footprint. Another valid choice would have been LightMAC, but with a higher memory cost [158].

For our hybrid solution, in the case of an RFF authentication reject (*Challenge* message), we could have switched back to the same method of just key-hashing the message, but since, in this case, another message was to be sent anyway from the GW to the SN, we chose to use a stronger Challenge–Response protocol based on the same MAC primitive, without any additional effort. In this way, we gained protection against Replay Attacks, without the need to keep a monotonic counter at both ends and without any more computational overhead.

3.6.2 Accuracy Evaluation

In this section, we evaluate how our hybrid system enhances an RFF scheme in order to achieve a much better accuracy.

Testbed of the Accuracy Evaluation System

The receiving side includes three RF anchors placed in different locations. They all receive the messages sent by the nodes and at the same time measure the RSSI. This value, along with the message, is sent to a computer. Each message sent by the sensors has a unique ID, so it is possible to synchronize the values received by the anchor nodes. Therefore, for each message received, the computer also receives three RSSI values. These values are used to create a simple RFF of the sender. A machine learning algorithm is then used to identify and validate the sender based on this RFF. In the case where the identity has not been validated based on the RFF, the computer sends a *Challenge* message and follows the protocol described in Section 3.4.5. A high-level schematic of the system is presented in Figure 3.5.

The nodes are all based on three different types of Texas Instruments ultra-low-power development boards, all powered by TI MSP430 RISC micro-controllers. The RF module is the CC110L transceiver, which uses the 868–870 MHz industrial, scientific, and medical (ISM) radio band. The CC110L transceiver is integrated using the Anaren CC110L RF BoosterPack, a low-power wireless transceiver extension kit compatible with the MSP-EXP430 (Figure 3.6a). The anchor points use the same radio modules as the sensor nodes, mounted on TI TM4C1294

Connected LaunchPad Evaluation Boards.

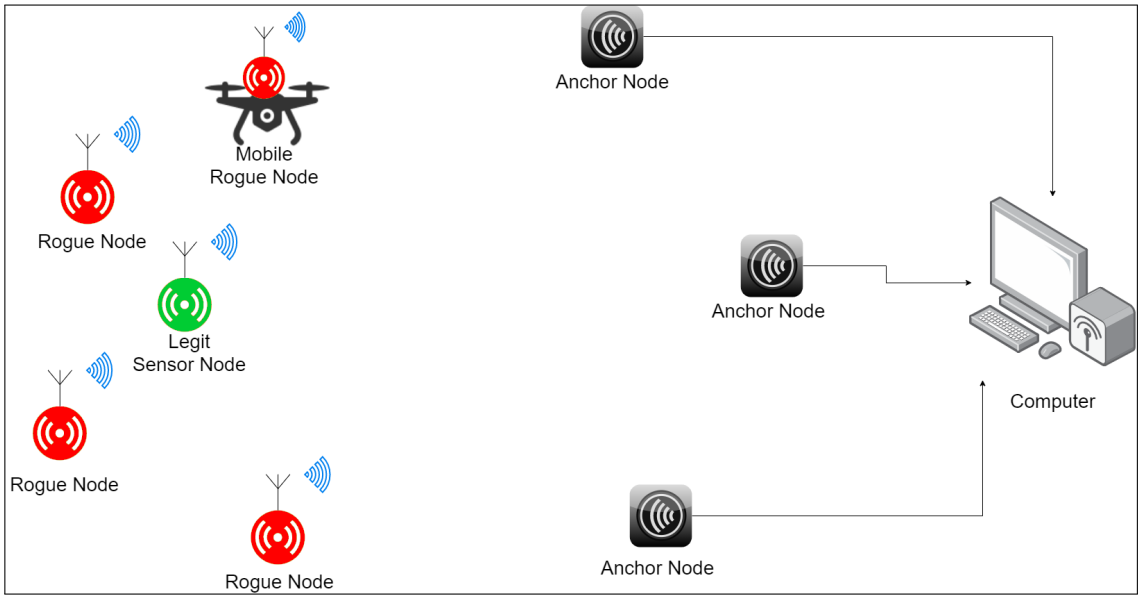
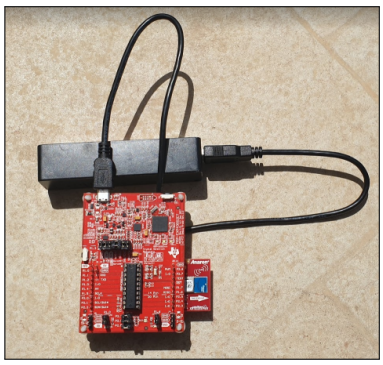
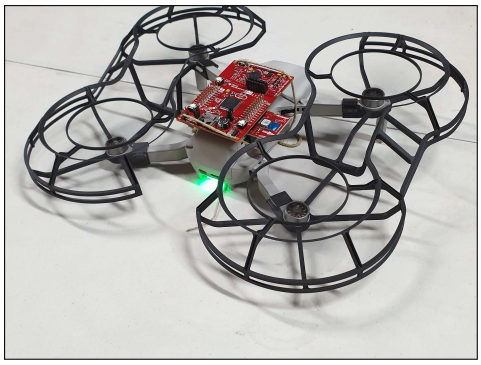


Figure 3.5: Evaluation system setup.



(a) Static node



(b) Airborne node

Figure 3.6: (**a**) MSP-EXP430G2ET, powered by an external battery pack. (**b**) MSP-EXP430FR5994, mounted on a Mavic Mini drone. Both use a CC110L RF BoosterPack.

The evaluation was conducted during the lockdown period of the COVID-19 crisis. As such, it was necessary to find a solution to the limitation of movement, but still to be able to take measurements from different positions. For this purpose, we used a drone that was able to carry some nodes and simulate rogue message attacks from multiple locations, even in three-dimensional space. This turned out to be a very efficient way to test the system.

The legit SN and two rogue nodes, including the drone-mounted one, each used an MSP-

EXP430FR5994. The choice was due to the fact that it could be easily mounted on the drone, since it can be powered by means of an on-board super capacitor instead of an external battery [159] (Figure 3.6b). Other rogue nodes used MSP-EXP430G2ET and were powered by an external battery pack (Figure 3.6a).

The legit SNs were positioned 12, 9, and 8 m away from the anchor nodes. The airborne rogue node was guided in different positions in the air, from 0.5 m up to 25 m from the legit SN, at different angles from the anchors. The other rogue nodes were placed in different fixed positions, the closest being 30 cm from the legit SN (Figure 3.7).

Accuracy Evaluation System Results

The RSSI value returned by the CC110L RF module of the anchor nodes is an estimate of the signal power level, based on the current gain setting in the RX chain and the measured signal level in the channel [160].

For each message sent by an SN and received by the anchor nodes, each anchor retransmits, via wired Ethernet connection to the computer, the received message along with the precise RSSI measured by its radio module. The computer determines the source of the packets' origin based on the RSSI triplets, as explained below.

In Figure 3.8, each point represents the triplets of the RSSI values measured by the anchors for a single message. We can see the different clusters of points colored for each transmitting node. Based on these values, we can use a classifier to identify the source of future messages.

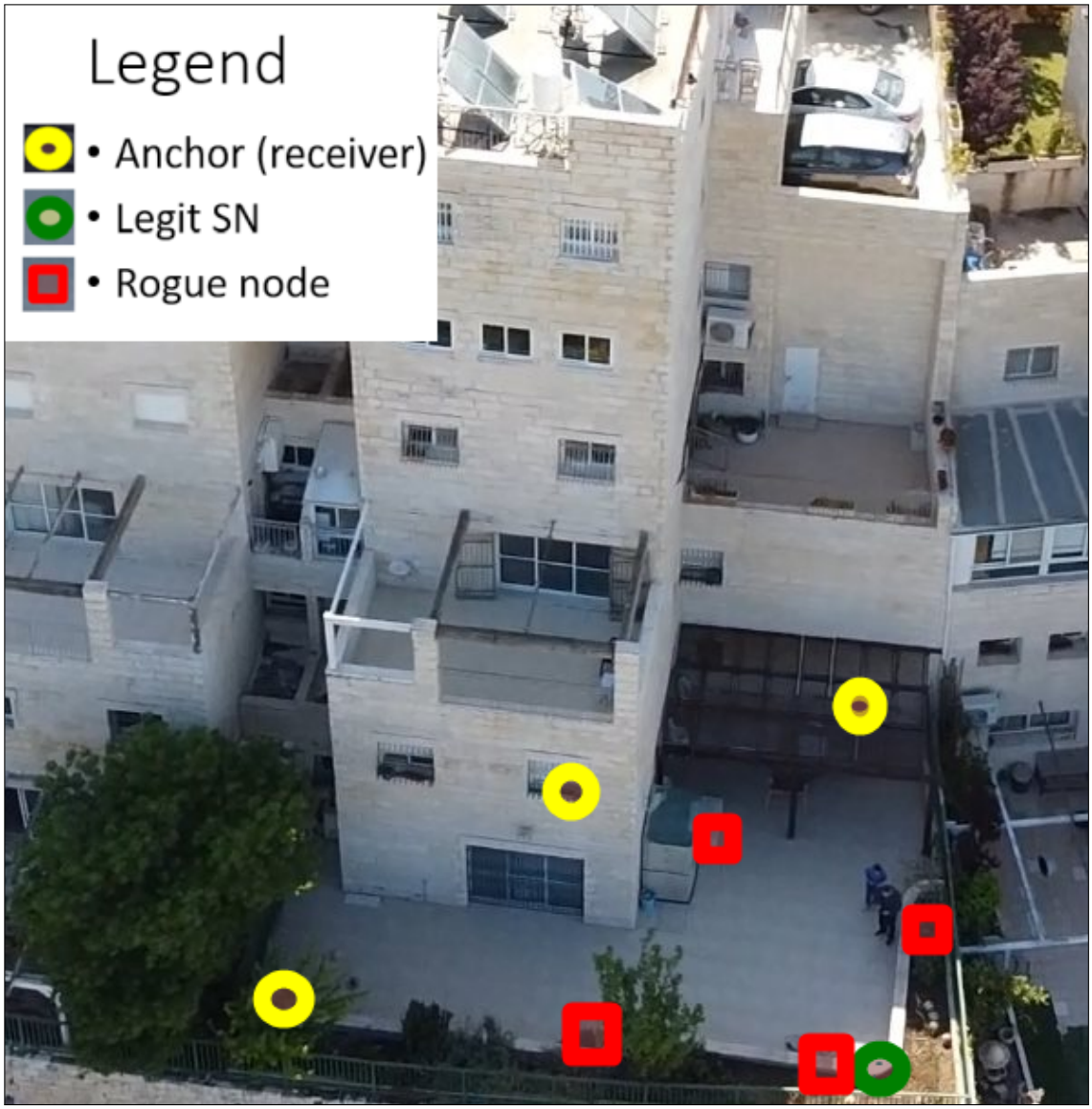


Figure 3.7: Photography of the live setup of the experiment, taken from the airborne rogue node.

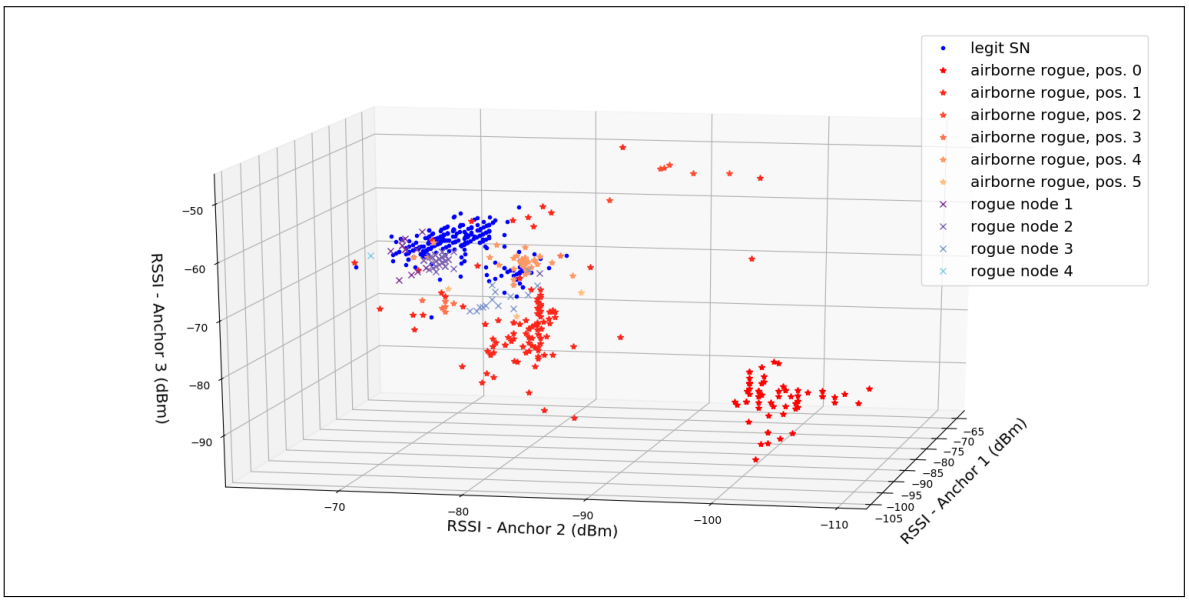


Figure 3.8: RSSI-triplets measured for 500 valid messages.

The authors of [161] discuss ML for RFF-based identification, reaching good accuracy using bagged tree and weighted k-nearest neighbors (KNN) algorithms. Based on the observations made about Figure 3.8, a natural choice for our evaluation would have been a weighted KNN classifier. Since our goal was to be able to change the configuration in order to reach different values of FAR and FRR, we use a regression KNN model, whose output value will be compared to a threshold value. If, for a certain input, the output of the KNN regressor is greater than the threshold, the message is considered as coming from the valid SN. The threshold value is chosen to have an acceptable FAR, as decided by the system requirements. For an RFF-only system, the accuracy is calculated as $Accuracy_{RFF} = 1 - (FAR + FRR)$, while for the hybrid system, since a reject falls back to a trusted cryptographic method, the total accuracy will be $Accuracy_{Hybrid} = 1 - FAR$.

For the evaluation, all nodes sent messages every second. For the training part of the KNN, the GW responded to all messages received by a *Challenge* response, thereby assuring the source of the messages even without the RFF. The first 500 valid messages (i.e., those with a valid CRC received by all three anchors) were used as the training set by the KNN regressor, with K set to 10, and from this point on, the GW followed the hybrid protocol. Figure 3.9 shows how the threshold influenced the FRR and FAR of the system for 2000 messages. By choosing

a threshold > 0.9, we reached a FAR $= 0.20\%$ and FRR $= 10.89\%$. Therefore, even though the bare RFF system gave 88.91% accuracy, the hybrid protocol reached 99.8% accuracy, much better than the other alternatives of pure RFF systems.

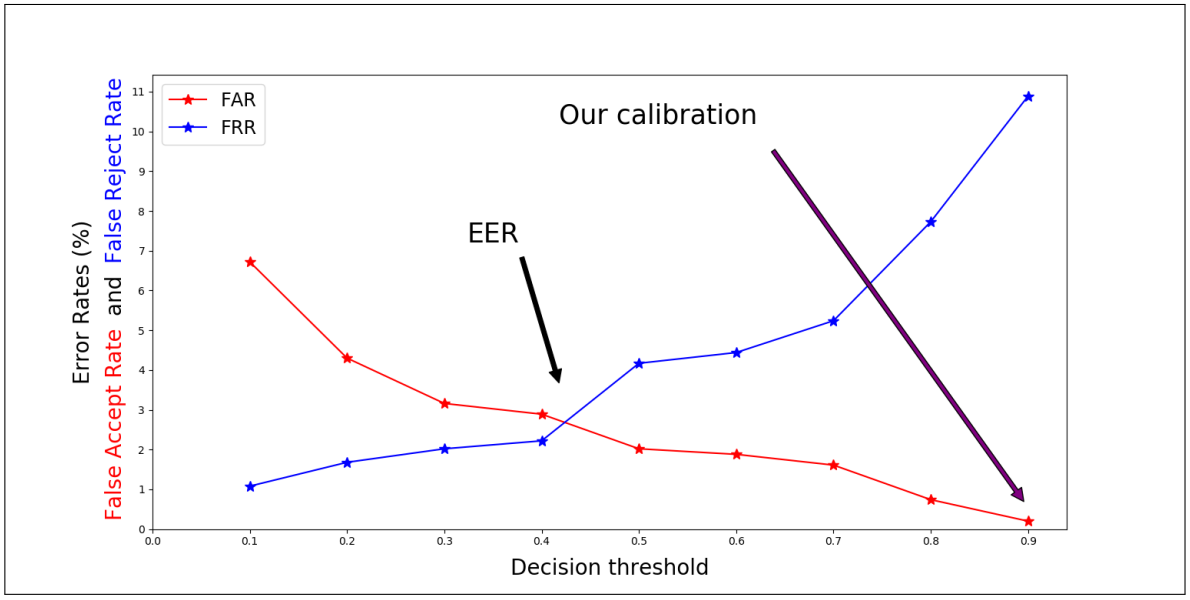


Figure 3.9: Evaluation RFF calibration by low FAR.

3.6.3 Performance Evaluation of Energy Efficiency

In the following, we present a performance evaluation that compares the hybrid scheme as described in Section 3.6.1 with two approaches: a cryptographic protocol using only a Keyed-Hash Message Authentication Code (*MAC-only*) to authenticate each message, similar to the legacy HMAC [37], and an RFF-only system.

Selected Protocols Description

The MAC-only nodes implementation makes use of the same Chaskey-12 algorithm defined in Section 3.6.1 in order to authenticate each message sent. The rationale behind the choice of using a MAC-based protocol is that the MAC does not require any other message (unlike a full CHAP) and can be seen as one of the most lightweight cryptographic-only authentication approaches.

In the case of a message received from the MAC-only SN, the GW checks only the MAC

validity and sends back the *SuccessMsg* message. In the case of a message from the hybrid protocol SN, a *Challenge* response or a *SuccessMsg* is issued in response to the *Message*, according to a simulated FRR of 10%. This value was chosen because it led to a FAR of 0.2% in our live experiment. It should be noted that according to recently published studies, an FRR as low as 10% would lead to a much better accuracy of our system and a FAR of $\sim 10^{-9}$ according to the results of recent RFF techniques, as demonstrated by [100]. To complete the evaluation and demonstrate how future improvements of RFF technologies can lead to better energy savings, we also present the results of the measurements with an FRR of 5% and, as a theoretical limit, the power consumption of an RFF-only system.

Testbed of the Energy Efficiency Evaluation System

To evaluate the energy efficiency of our authentication schemes as presented in this work with equipment simulating IoT SN, we used the Texas Instruments ultra-low-power microcontroller MSP430G2x and the sub-1 GHz RF transceiver CC110L, as in Section 3.6.

In order to achieve precise measurement, EnergyTrace (a Texas Instruments technology) was used, by means of the MSP-EXP430G2ET Launchpad Evaluation Kit and Code Composer Studio (CCS) [162]. In order to precisely calculate the energy profile, EnergyTrace makes use of an on-board DC–DC converter, which generates the power for the target. The pulses of the converter are counted by the software controlling the converter, and the measurements are acquired through CCS. Other methods for evaluating energy consumption have been proposed, such as the approach described in [163] which describes a methodology for measuring the power consumption of cryptographic functions on a Raspberry Pi. In contrast, our approach utilizes the specific capabilities of EnergyTrace technology on MSP boards to achieve high accuracy in measuring the energy profile of our authentication schemes.

The testbed consists of two identical kits. The first one runs the cryptographic-only baseline protocol (MAC-only). The second one runs a complete implementation of the hybrid protocol. The SNs transmit a data packet (the temperature) every 3 s to a GW; this is much faster and more energy-consuming than real-life scenarios. This also prevented the nodes from using the "deep-sleep" mode of the modules between two transmissions. However, it is appropriate to

compare the two models, given less time spent on the simulation. The energy consumption is measured continuously for each SN.

The GW runs on a similar platform but implements the server side of the protocol. Since our goal is to reduce the energy consumption of the SN, while the client side implements a real implementation of our protocol, the GW simulates the RFF, based on our results from Section 3.6 and the different values to be compared (simulated FRR of 10%, 5%; and 0% for the simulated RFF-only one). This test bed is shown in Figure 3.10.

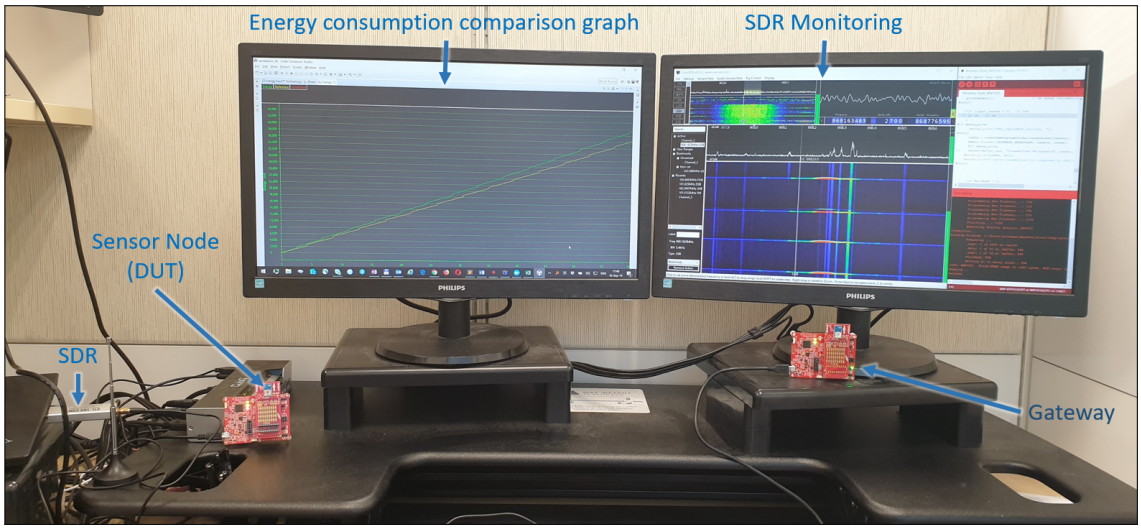


Figure 3.10: Performance evaluation lab.

Energy Efficiency Evaluation Results

Table 3.1 presents the comparison of the energy measurements for the MAC-only (baseline) protocol and the hybrid protocol, calibrated for a FRR = 10%, FRR = 5% and RFF-only (i.e., no fallback to cryptographic authentication). The profile used for the energy measurement was 3 V, 2400 mAh battery, equivalent to two standard AA batteries. The MAC-only protocol showed an increase in energy consumption of 24.4%, resulting in a decrease in battery life of almost 19.6% in comparison to the hybrid protocol configured for a simulated FRR = 10% as in Section 3.6, and which achieved a total accuracy of 99.8%.

Table 3.1: Energy measurement comparison.

System Name	**MAC-Only**	**Hybrid w/FRR = 10%**	**Hybrid w/FRR = 5%**	**RFF-Only**
Time	300 s	300 s	300 s	300 s
Energy	564.743 mJ	454.076 mJ	404.790 mJ	355.412 mJ
Power				
Mean	2.0922 mW	1.6342 mW	1.4625 mW	1.3449 mW
Min	0.0000 mW	0.0000 mW	0.0000 mW	0.0000 mW
Max	82.5882 mW	83.5259 mW	83.7475 mW	83.3862 mW
Voltage				
Mean	3.2798 V	3.2793 V	3.2793 V	3.2796 V
Current				
Mean	0.6373 mA	0.4980 mA	0.4457 mA	0.4111 mA
Min	0.0000 mA	0.0000 mA	0.0000 mA	0.0000 mA
Max	25.1640 mA	25.4552 mA	25.5250 mA	25.3826 mA
Battery Life (3 V, 2400 mAh)	5 months 7 days	6 months 15 days	7 months 9 days	8 months 2 days

Figure 3.11 shows the energy consumption graph of this setup over 5 min. Due to the first full authentication, the hybrid protocol was a little more energy-demanding at the start of the experiment. However, after only six transmissions, the total energy used by the MAC-only protocol was equivalent to that of the hybrid one, and after that, the advantage of the hybrid approach was apparent and grew almost linearly.

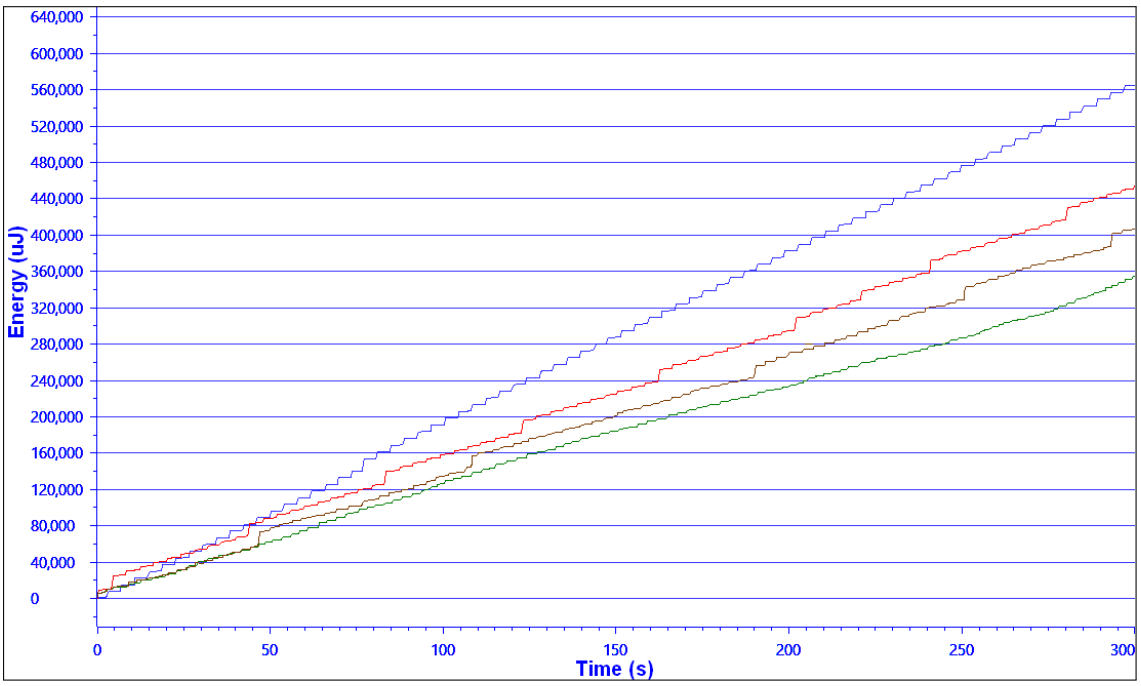


Figure 3.11: Energy measurement graph—From top to bottom: MAC-only (**blue**) vs. Hybrid FRR = 0.1 (**red**) vs. Hybrid FRR = 0.05 (**brown**) vs. RFF only (**green**).

These results clearly show the value of the RFF as a primary method of authentication as part of a hybrid protocol, as defined in the first part of Section 3.4. In this energy consumption evaluation, we tried the hybrid system with values of the FRR configured to 10% and 5% to simulate recent RFF methods; any progress in the RFF technology allowing the use of lower FRR value without negatively affecting the FAR value will immediately drive significant energy savings for a given security level.

3.7 Conclusion

In this work, we described the need for lightweight and "good-enough" security for resource-starving devices. We proposed a protocol scheme able to leverage modern RF physical-layer-based fingerprinting methods and lightweight cryptographic solutions and to create a flexible hybrid message authentication scheme, without compromising the security level required. This scheme can save time and resources by using current RFF solutions, even if their intrinsic level of precision is not on par with the cryptographic-only methods. We evaluated this approach

by precise energy level measurements, comparing the total energy consumption of a baseline cryptography-only authenticated protocol to a complete and fully real-time implementation of our hybrid scheme in IoT sensor nodes. The results provide a clear statement about the energy efficiency of this approach.

Part II

Shmulik - Deep Learning System for Software Vulnerability Analysis

Chapter 4

Code Analysis Techniques

Contents

Code analysis is a crucial step in ensuring the security and reliability of the IoT device firmware. This chapter focuses on the various code analysis techniques used to identify vulnerabilities and improve firmware quality. We will explore traditional methods such as static and dynamic analysis, symbolic execution, as well as machine learning-based and manual code review approaches. These techniques are complementary and each offers unique strengths and weaknesses. Even as they continue to evolve and improve, the reality is that firmware vulnerabilities remain a significant concern, highlighting the need for ongoing innovation and advancement in code analysis capabilities.

Software vulnerabilities pose significant threats to the security and reliability of computer systems, which necessitate the development of effective detection methods. Over the years, researchers and practitioners have explored various approaches to detect software vulnerabilities, with the aim of identifying and minimizing potential security weaknesses. As highlighted in a recent survey [1], the increasing threat surface of IoT devices has made firmware security

analysis an essential task to ensure the security and trustworthiness of these devices. Feng et al.'s approach categorizes firmware security analysis into four perspectives: binary code, firmware image, IoT network, and manual analysis (Figure 4.1).

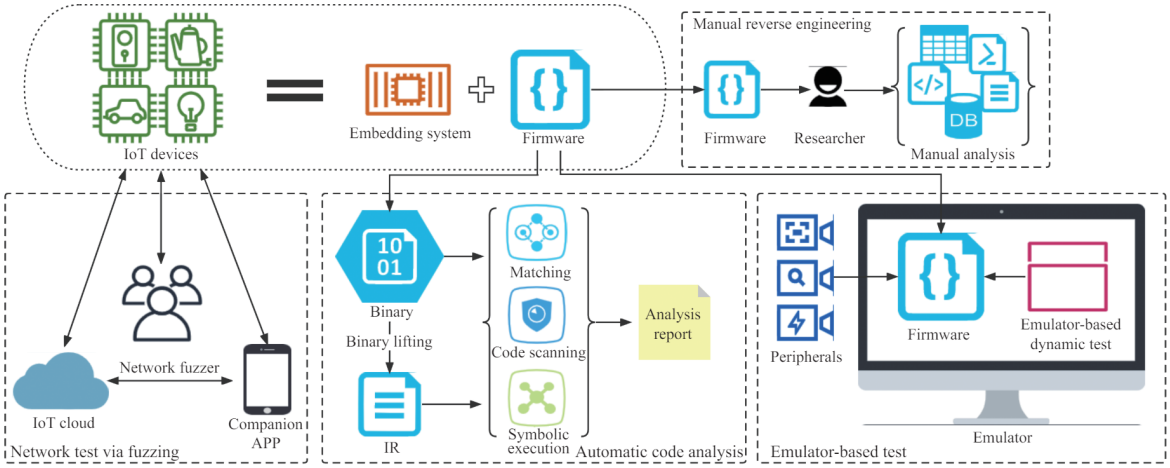


Figure 4.1: Firmware analyzed based on binary code, firmware image, IoT network and manual analysis [1]

In the following sections, we will focus on various aspects of software vulnerability detection, exploring methodologies that include:

- Static Analysis, involving an examination of source code or binary structures
- Dynamic Analysis and Fuzzing, encompassing the runtime execution of software and the use of fuzzing techniques to uncover vulnerabilities
- Symbolic Execution, a method that employs symbolic representation to traverse all possible code paths
- Machine Learning-based Vulnerability Detection, harnessing the capabilities of algorithms and statistical models to automatically acquire knowledge of patterns and attributes of vulnerabilities from datasets, allowing the identification of potential vulnerabilities
- Manual Code Review, providing nuanced insights and complementing automated approaches

It should be noted that while our work aligns with [1] categorization, we deviate from their approach in our network testing methodology. Specifically, we consider network sockets as just another interface for fuzzing, whereas they treat network testing via fuzzing as a distinct category. Our approach focuses on the methodologies employed, rather than the specific interface used.

Each section explores a different type of software vulnerability detection, delving into various methodologies and techniques used to identify and mitigate potential security weaknesses, providing a snapshot of the current state of the art in each area.

4.1 Static Analysis

Static analysis techniques involve examining software artifacts, such as source code or compiled binaries, without executing the software. These methods aim to identify vulnerabilities by analyzing the code structure, control flow, and data flow properties. Static analyzers employ a variety of algorithms and heuristics to detect common vulnerability patterns, such as buffer overflows, format string vulnerabilities, and injection attacks [164, 165, 166].

One common static analysis approach is *pattern matching*, where predefined vulnerability patterns or signatures are compared with the source code or binary representations [167]. These patterns capture known vulnerability patterns, allowing the detection of well-known security issues. However, obviously, pattern-matching approaches may struggle to detect previously unknown vulnerabilities.

Static analysis can be further categorized into *syntactic and semantic analysis*. *Syntactic analysis* focuses on the structure and syntax of the code, identifying issues such as syntax errors, violations of coding rules, and possible security risks based on code patterns. *Semantic analysis* intends to search for more complex issues, by examining data and control flows, and the relationships between variables and functions to identify potential vulnerabilities. [168]

Another static analysis technique is *static taint analysis*, which tracks the flow of user-controlled input throughout the code and identifies potential security risks when tainted data reach sensitive operations or data structures.[169, 170]

Furthermore, static analyzers can use *program slicing*, where relevant portions of the code that affect a particular vulnerability are extracted and analyzed. This technique reduces the complexity of the analysis and allows for more targeted vulnerability detection.[171, 172]

4.2 Dynamic Analysis and Fuzzing

Dynamic analysis is a broad category of vulnerability detection techniques that involves monitoring and analyzing the behavior of a software system during its execution. A popular technique within dynamic analysis is *fuzzing*, which is widely used to detect vulnerabilities. Fuzzing involves providing unexpected or malformed inputs to a target program and observing its response. This approach aims to trigger abnormal behaviors or crashes that may indicate the presence of vulnerabilities. [173, 174, 175]

Grey-box fuzzing, a variant of fuzzing, combines elements of both black-box and white-box approaches. It leverages partial knowledge of the system's internal structure with lightweight instrumentation to guide the fuzzing process effectively. By using dynamic analysis, fuzzing techniques can uncover vulnerabilities such as memory corruption issues, input validation errors, or unexpected behaviors that occur during program execution.[176]

During the fuzzing process, a fuzzer generates a large number of test inputs by mutating or generating data based on the program's expected input format or specification. These inputs are then fed into the target program, and the fuzzer monitors the program's behavior, such as crashes, hangs, or memory access violations. By analyzing the program's responses to these inputs, potential vulnerabilities can be identified.

Fuzzing has proven to be an effective method for uncovering known and unknown vulnerabilities. It has discovered numerous security issues in various software systems and has been instrumental in improving their overall resilience. Fuzzing can be applied to different software components, including network protocols, file parsers, web applications, and more.[174, 177]

Dynamic analysis techniques, including fuzzing, provide several benefits for vulnerability detection. They can uncover vulnerabilities that manifest during runtime and are difficult to identify through static analysis alone. Additionally, dynamic analysis allows for the detection

of vulnerabilities that are dependent on specific program inputs or execution paths, enabling a more targeted and realistic assessment of the system's security.

Despite its effectiveness, fuzzing also has limitations. It heavily relies on the quality and diversity of test inputs, and there is no guarantee that all vulnerabilities will be triggered during the fuzzing process. Additionally, the scalability and efficiency of fuzzing can be challenging, particularly for large and complex software systems.

Researchers and practitioners continue to advance dynamic analysis techniques, including fuzzing, by developing more sophisticated fuzzers, improving input generation strategies, and leveraging machine learning and evolutionary algorithms to enhance vulnerability detection capabilities. [175] These advancements aim to make dynamic analysis techniques more efficient, scalable, and effective in identifying software vulnerabilities.

4.3 Symbolic Execution

Symbolic execution is a technique that systematically explores all possible paths of a program by executing it symbolically, using symbolic values instead of concrete inputs. It allows for the generation of path constraints that represent the conditions necessary to reach specific program paths or trigger certain vulnerabilities.[178]

By solving these path constraints, symbolic execution can identify inputs that lead to potential vulnerabilities, such as assertion failures, uninitialized memory accesses, or privilege escalation. Symbolic execution-based tools can also generate test cases that exercise specific paths or trigger specific vulnerabilities.

However, symbolic execution suffers from the "path explosion" problem, where the number of possible program paths grows exponentially with the complexity of the code, leading to scalability issues. To mitigate this problem, researchers have developed techniques such as constraint pruning, path merging, and concolic execution, which combine symbolic execution with concrete execution to overcome the limitations of pure symbolic execution.[179]

4.4 Machine Learning-based Vulnerability Detection

Machine learning techniques have gained significant attention in the field of software vulnerability detection. These approaches leverage the power of algorithms and statistical models to automatically learn the patterns and characteristics of vulnerabilities from labeled data sets, allowing the detection of potential vulnerabilities in new code or binaries.[180]

In machine learning-based vulnerability detection, a model is trained on a dataset that contains examples of both vulnerable and non-vulnerable code. The model learns to differentiate between the two by extracting relevant features from the code, such as lexical, syntactic, or semantic information. These features can include the presence of specific function calls, API usage patterns, variable interactions, or code structure.[181]

Various systems based on this idea have been developed over the last few years. [182] introduced a hybrid technique combining N-gram analysis and feature selection algorithms for predicting vulnerable software components, achieving high precision, accuracy, and recall. [183] presented a data-driven approach to vulnerability detection for C and C++ programs, combining deep neural network models with tree-based models.

[184] introduced deep-learning based vulnerability detection at the slice level, where slices represent multiple lines of code with inherent semantic relations, such as data dependency or control dependency. In a subsequent work, [185], the authors leveraged intermediate code to accommodate additional semantic information, enhancing the identification of the vulnerability locations.

[186] addressed scalability and accuracy in large-scale source code vulnerability scanning with VulCNN, which converted the source code into images for efficient and precise detection. [187] proposed VulANalyzeR, a deep learning-based model for automated binary vulnerability detection, classification, and root cause analysis, offering explainability in vulnerability detection.

Recently, Large Language Models (LLMs) have shown substantial promise in software vulnerability detection. ChatGPT powered by OpenAI's GPT [188] has democratized access to LLM capabilities to the masses, and it led to a rush for LLM based solutions for vulnerability

detection [189, 190]. FalconLLM [191] is an open model with 180 billion parameters, and has been fine-tuned for cybersecurity applications, resulting in SecureFalcon [191], a C vulnerability distinguishing model tested using a generative AI trained dataset [192]. It also shows some hope in the domain of vulnerability repair.

The key advantage of machine learning-based approaches is their ability to learn various patterns of vulnerabilities in complex codebases. As the vulnerability patterns in the training set are more diverse, it enables the model to capture generalizable patterns of them. That even allows the model to detect previously unseen vulnerabilities in new code.

However, vulnerability detection based on machine learning also faces challenges. The quality and representativeness of the training dataset are critical factors in achieving accurate detection. The dataset must encompass a wide range of vulnerability types and cover different programming languages and application domains to ensure robustness. Additionally, handling imbalanced datasets, where the number of vulnerable samples is significantly smaller than non-vulnerable samples, is a common challenge that requires careful handling during training [172].

Furthermore, the interpretability of machine learning models is an important consideration. Understanding the reasoning behind the model's decisions and identifying the features that contribute most to vulnerability detection can help build trust and facilitate further analysis and improvement of the detected vulnerabilities.

To enhance the effectiveness of machine learning-based vulnerability detection, and in some cases, even vulnerability repair, researchers started exploring techniques such as transfer learning [193], ensemble methods [194], and active learning [195]. These techniques aim to improve generalization, handle data scarcity, and reduce the reliance on manual labeling efforts.

Machine learning-based vulnerability detection complements other techniques in the software security domain, such as static analysis and dynamic analysis. The combination of different approaches and their integration into comprehensive vulnerability detection frameworks can provide more robust and accurate results in identifying and mitigating software vulnerabilities.

LLMs and Real-Life Scenarios

Given the substantial academic and industry interest in LLMs, [196] recently emphasized

the considerable gap between the current capabilities of LLMs and the practical requirements for deploying them in security roles. Their research highlights the significant difference between testing LLMs on small code snippets, as traditionally done in benchmarks, and evaluating their performance on real-life code with complex vulnerabilities. The study reveals that while LLMs may show promising results on smaller datasets, they struggle with the intricacies and nuances of real-world vulnerabilities.

This gap underscores the importance of moving beyond simplistic testing scenarios towards more realistic evaluation environments that better reflect the challenges encountered in practical software development. By exposing the limitations of LLMs in detecting vulnerabilities in real-life code, their research emphasizes the need for innovative approaches and more comprehensive training strategies to bridge the disparity between testing on small code samples and real-world applications.

4.5 Manual Code Review

Another widely used method for software vulnerability detection is manual code review. Manual code review involves a detailed examination of the source code by experienced developers or security experts to identify potential vulnerabilities. It relies on human expertise and knowledge of common programming pitfalls and security best practices.[197]

During manual code review, the reviewers analyze the code for coding errors, insecure practices, and design flaws that could lead to vulnerabilities. They examine the code to identify potential issues such as input validation problems, insecure data storage, improper access controls, and potential injection vulnerabilities. Manual code review also allows for the detection of subtle logic flaws or vulnerabilities that may not be easily identified by automated techniques.

Although manual code review can be time-consuming and resource-intensive, it offers several advantages such as the following. It enables a deep understanding of the codebase and its specific context, allowing reviewers to identify vulnerabilities that may be unique to the system. Furthermore, manual review can uncover complex vulnerabilities and provide information

on improving overall code quality and security.[198]

However, manual code review has certain limitations. It is heavily based on the experience and expertise of the reviewers, which can introduce subjectivity and variations in the detection process. The effectiveness of manual code review also depends on the thoroughness and attention to detail of the reviewers, making it susceptible to human error or oversight. Moreover, manual review may not be scalable for large codebases or time-constrained projects.

To overcome these limitations, researchers have explored ways to augment manual code review with automated tools and techniques. These tools can assist reviewers by automating certain checks, providing vulnerability suggestions, and flagging potential code segments for further review. The combination of manual code review with automated assistance can improve the efficiency and effectiveness of the detection process [199, 200].

In general, manual code review remains an essential approach for software vulnerability detection, especially for complex or critical systems. It complements automated techniques by leveraging human expertise to uncover vulnerabilities that may not be easily detectable through automated means alone. And modern methods are continuously evaluated to improve the efficiency of code review.[201, 202, 203]

4.6 Conclusion

This chapter has presented a taxonomy of various code analysis techniques used to identify vulnerabilities and improve firmware quality. We have discussed traditional methods such as static and dynamic analysis, symbolic execution, as well as machine learning-based and manual code review approaches. Each technique offers unique strengths and weaknesses, and their complementary nature highlights the need for a comprehensive approach to code analysis. Despite advancements in code analysis, firmware vulnerabilities remain a significant concern, emphasizing the need for ongoing innovation and advancement in code analysis capabilities.

The symbiosis of AI and manual code review presents a promising direction in code analysis. Using the strengths of both, we can develop more effective and efficient vulnerability detection methods. However, challenges such as the effective scaling of AI models, integrating diverse

analysis techniques, and the need for continuous updates to address emerging threats remain significant.

Future research should focus on addressing these challenges, ensuring that techniques can keep pace with evolving security threats.

In the next chapter, we will introduce our method which combines AI-based analysis and human expertise.

Chapter 5

Shmulik: Enhanced Control-Flow Vulnerability Detection by Combining AI-Based Analysis and Human Expertise

Contents

Securing software against vulnerabilities is a critical concern, given potential risks such as compromised security, data breaches, and system crashes. Detecting these vulnerabilities during the design phase is crucial, particularly as modern codebases become more intricate. However, the vast and complex nature of contemporary software projects renders reliance solely on human efforts impractical. As projects grow, automated methods become essential for effective and thorough vulnerability detection.

Despite the recognized importance of automated vulnerability detection, current methods often fall short when applied to extensive projects. Issues such as false positives and other limitations restrain their effectiveness, creating a critical gap in ensuring some robust security of large-scale software systems. To address this challenge, we introduce Shmulik, a deep learning-based vulnerability detection system that operates directly on the intermediate representation (IR) of the compilation of software programs. Shmulik extracts relevant operations on relevant variables from the IR, enabling a comprehensive analysis of vulnerabilities.

Unlike previous similar approaches that relied on human-defined features and small code samples from corpora such as SARD (Juliet) [204] for both training and evaluation, Shmulik

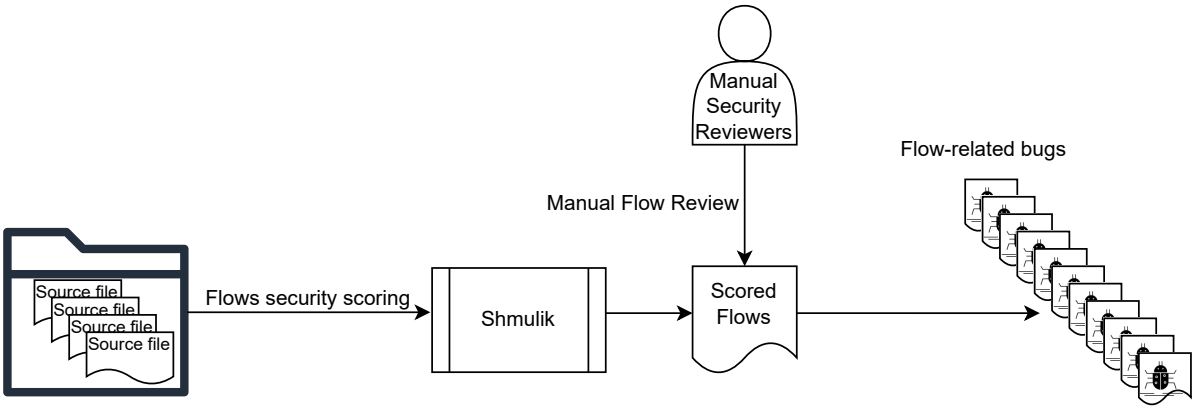


Figure 5.1: Generalized Shmulik usage flow

tackles the challenges posed by large-scale codebases. We observed that noise in the form of false positives rendered existing systems impractical for real-world usage. Our insight is that to address this gap, we need to collaborate efforts between automatic systems and researchers to create a hybrid method. For that purpose, we fine-tuned Shmulik to optimize code reviews conducted by researchers or developers, enhancing their effectiveness. By "coloring" different sections of code based on their risk level, Shmulik assists in identifying potential vulnerabilities, enabling review teams to prioritize their efforts effectively.

Furthermore, Shmulik is adept at the task of prioritizing areas of interest within the codebase, rather than focusing solely on detecting individual bugs within specific files or modules. It can indicate large parts of a project with the highest chances of having system vulnerabilities, enabling teams to focus their review efforts accordingly in extensive codebases. This approach streamlines the review process and helps allocate resources efficiently, ultimately bolstering code quality and security.

An overview of the Shmulik workflow is shown in Figure 5.1.

We evaluate Shmulik using a dataset specifically designed for deep learning approaches and test it on the widely used software library, libtiff. Remarkably, our evaluation uncovers several zero-day vulnerabilities in libtiff, showcasing the effectiveness of Shmulik in identifying previously unknown security risks.

By harnessing the power of deep learning and leveraging IR, Shmulik demonstrates its potential to improve software security by proactively detecting vulnerabilities, including zero-

day vulnerabilities. In addition, its focus on improving code review practices and providing prioritization guidance to review teams makes it a valuable tool for developers and researchers alike.

The subsequent sections of this chapter are structured as follows:

- In Section 5.1, we explain the problem statements and outline our contributions.
- Section 5.2 introduces Shmulik's design and the methodologies employed.
- Section 5.3 discusses our evaluations and results.
- In Section 5.5, we present the main challenges met by our approach, and some possible ways to improve it.
- Finally, Section 5.6 concludes this chapter.

5.1 Problem Statement and Contributions

5.1.1 Bridging AI and Human Expertise

Shmulik is designed to serve as a complementary approach to existing static code analyzers, especially in the detection of flow-oriented vulnerabilities.Tracing the program's flow aligns with the natural thought process of human developers and reviewers. In this context, Shmulik functions as an enabling tool, enhancing their efficiency in a similar way to how an augmented reality system enriches one's perception of the physical world.

It is also important to recognize that both Shmulik and previous research in this field have not achieved 100% accuracy in vulnerability detection within real-world projects. While some modern deep learning-based vulnerability detection systems may exhibit near-perfect accuracy in their evaluation measurements, real-world usage has so far shown that no system has been able to completely eliminate entire classes of vulnerabilities. As pointed out by [172], a notable issue with current approaches is their reliance on evaluation datasets that often have limited scope. These datasets are typically used to assess a model's performance, but may not offer a comprehensive view of how well the model can generalize to real-world examples, especially in

terms of false positives and false negatives. Moreover, vulnerabilities encountered in real-world situations tend to be significantly more intricate, requiring analysis of control flow, data flow, relationships, and various other interdependencies among code components.

In an effort to apply our technology in a real-world environment and benefit from it immediately, we propose an innovative hybrid approach that synergistically combines the AI-based Shmulik system with the expertise of human code reviewers. This method aims to capitalize on the strengths of both AI and human reviewers, thereby providing a more effective and comprehensive vulnerability detection process.

By using this combination of AI and human reviewers, we want to reduce the gap between the limitations of AI-based vulnerability detection systems and the ever-changing world of software security.

5.1.2 Prioritizing Resources for Effective Code Review

Shmulik can play a significant role in guiding the allocation of resources for code review, driven by several key factors:

- **Identifying Critical Areas in large codebases:** In large codebases, determining the most critical areas is highly valuable, even without specifying particular bugs. Shmulik assists code reviewers by identifying potential vulnerabilities within the codebase. This prioritization streamlines the review process, enabling reviewers to allocate resources where they will have the most impact.

- **Prioritizing Potential Exploitable Vulnerabilities**: Shmulik's approach to vulnerability detection is particularly tailored to identifying bugs and vulnerabilities that are pertinent to the control flow of a software program. This specialized focus allows Shmulik to excel in pinpointing vulnerabilities that have a direct impact on the program's security.

  It's important to note that Shmulik's methodology may not flag certain code mistakes that are typically caught by other static analyzers. The reason behind this is that such issues may not be readily triggerable through external inputs and consequently their

potential for exploitation by malicious actors is limited. Although these undetected issues still constitute bugs in the code, their practical significance may be less in the context of a project with finite resources for code cleanup.

Shmulik, in this regard, emerges as an invaluable tool. It distinguishes itself in prioritizing findings based on their potential for exploitability. This stands in stark contrast to the exhaustive review of all bugs or coding errors returned by most static analyzers. By categorizing and prioritizing vulnerabilities, Shmulik enables development teams to focus their attention and resources on the most critical areas of concern. This approach is not only pragmatic but also highly efficient, especially for projects with resource constraints, as it ensures that efforts are channeled towards addressing the most pressing security risks.

5.1.3 Enriched Learning Platform

We propose an innovative method of code evaluation that promotes a more enriching learning experience for reviewers, granting them the opportunity to cultivate a comprehension of various types of vulnerability and their manifestations within the code. Consequently, human reviewers can enhance their proficiency in identifying and addressing security risks, thus making a valuable contribution to the advancement of software security.

5.2 System Design

5.2.1 System overview

Shmulik is a deep learning-based vulnerability detection system designed to identify security-related bugs in C code, regardless of the code's complexity. The system operates on the IR of software programs' compilation using a customized version of the GCC compiler. Shmulik is designed to use flow as the minimal evaluation surface, making it highly useful for processing complex projects with massive flows that are challenging to follow manually, even for experienced researchers. Additionally, Shmulik provides a user-friendly Graphical User Interface (GUI) that enhances the interpretability of its output, offering an interactive visualization of

the identified vulnerabilities. This GUI allows users to intuitively navigate the project source code along the analysis results, contributing to a more effective and user-centric vulnerability assessment process.

Shmulik's execution flow can be summarized in the following high-level description in four steps:

1. Compilation: Shmulik compiles source code files in customized fashion to get their data in the form consistent for further processing

2. Code Flow Graph Creation: Shmulik generates a code flow graph to represent the code as a directed graph, with a primary emphasis on the control flow. The root nodes of the graph can be defined either at the API level of the module, or for functions that handle external inputs (such as `fread()`, `recv()`, ...). The flow progresses through intermediate-level representation (IR) operations, tracking modifications made on relevant arguments of interest.

3. Vector Extraction: After generating the code flow graph, Shmulik extracts valid sequences (vectors) from the graph and preprocesses (i.e. encodes) them. These vectors will be used as input for the neural network.

4. Model Training and Prediction Making: After the vectors have been extracted and preprocessed, a machine learning model is trained and used to generate predictions (*inference*).

This high-level overview of Shmulik's execution flow provides a foundation for understanding the system's design and functionality.

In the following sections, we delve deeper into the details of system input, execution flow, and output.

5.2.2 System Input

The input to the Shmulik system consists of two main components: training data and target project source code. Both types of inputs have specific requirements that must be met for the

system to function correctly.

Evaluation Dataset

Evaluation dataset includes all training, testing, and validation data, which are derived from a single comprehensive corpus of code. This corpus is divided into three parts: one for training, one for testing, and one for validation. The evaluation dataset should meet the following requirements:

- The training data code must be built using the GCC compiler. It does not have to be fully linked, as Shmulik uses an intermediate language representation, as explained later.
- Each function in the training data should be accompanied by a label that indicates whether it is malicious or benign. It allows the machine learning model to learn the patterns associated with vulnerable and non-vulnerable code. This labeling step is present only for training and not executed for inference (a.k.a. using Shmulik to find vulnerabilities).

Target Project Source Code

We define an *entry point* as a tuple of a method, an index of an argument within the method, and an indication of whether this function is an API function or not. The method refers to a function called in the source code that the Shmulik user suspects may have a potentially malicious argument, such as input from a file or a user. The API indication differentiates between API functions, which are tracked from the first line of the function, and non-API functions, which are tracked from the line following their invocation.

The target project source code should satisfy the following conditions:

- The source code must be compilable using the GCC compiler and contain a valid makefile with the CFLAGS variable specified. The code can be an application, library, etc.
- Shmulik requires a list of entry points to analyze the target project source code for potential vulnerabilities.

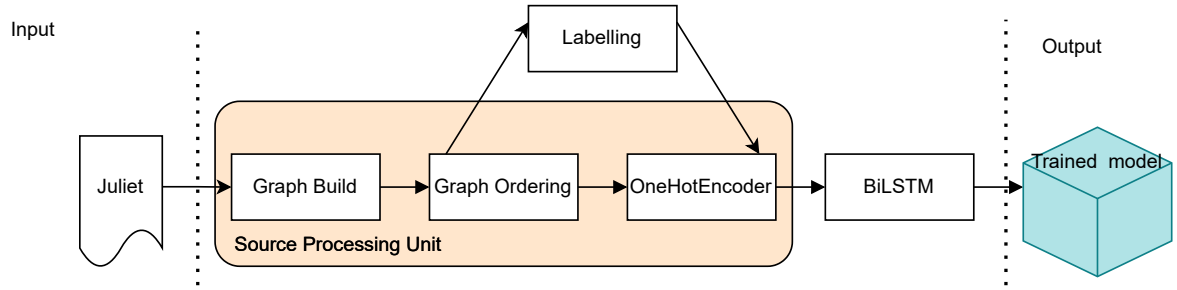


Figure 5.2: Implementation view of Shmulik training flow

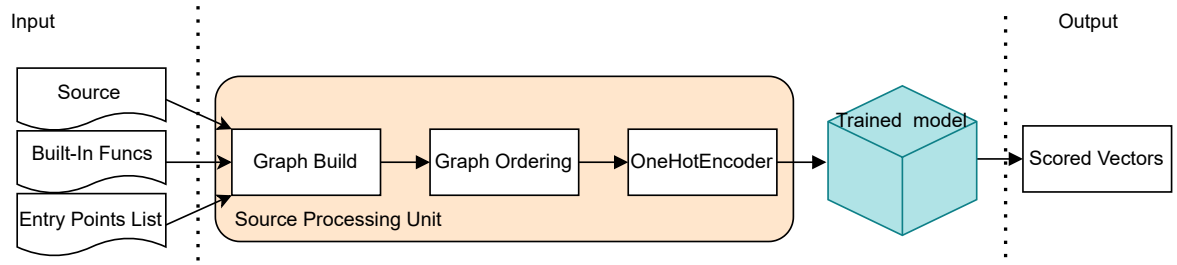


Figure 5.3: Implementation view of Shmulik inference flow

These conditions reflect common practices in software development, and their presence ensures that Shmulik can be effectively integrated into projects following standard conventions, enhancing its usability and applicability to a wide range of software projects. Support for other toolchains is a consideration for future work, aiming to broaden Shmulik's compatibility with diverse development environments and build systems.

5.2.3 Shmulik Execution Flow

As mentioned above, the Shmulik execution flow consists of one preliminary step and four main steps: creating a code flow graph, translating the graph into valid vectors, and using a machine learning model for training and prediction. Most of these steps are similar for both training of the model and inference. For instance, the code flow graph and vectors extraction are identical in both training and inference, with the exception of the labeling step (needed only for training), as noted in Section 5.2.2. Figures 5.2 and 5.3 show Shmulik execution flows for each path.

In this section, we provide a detailed description of each step in the execution flow.

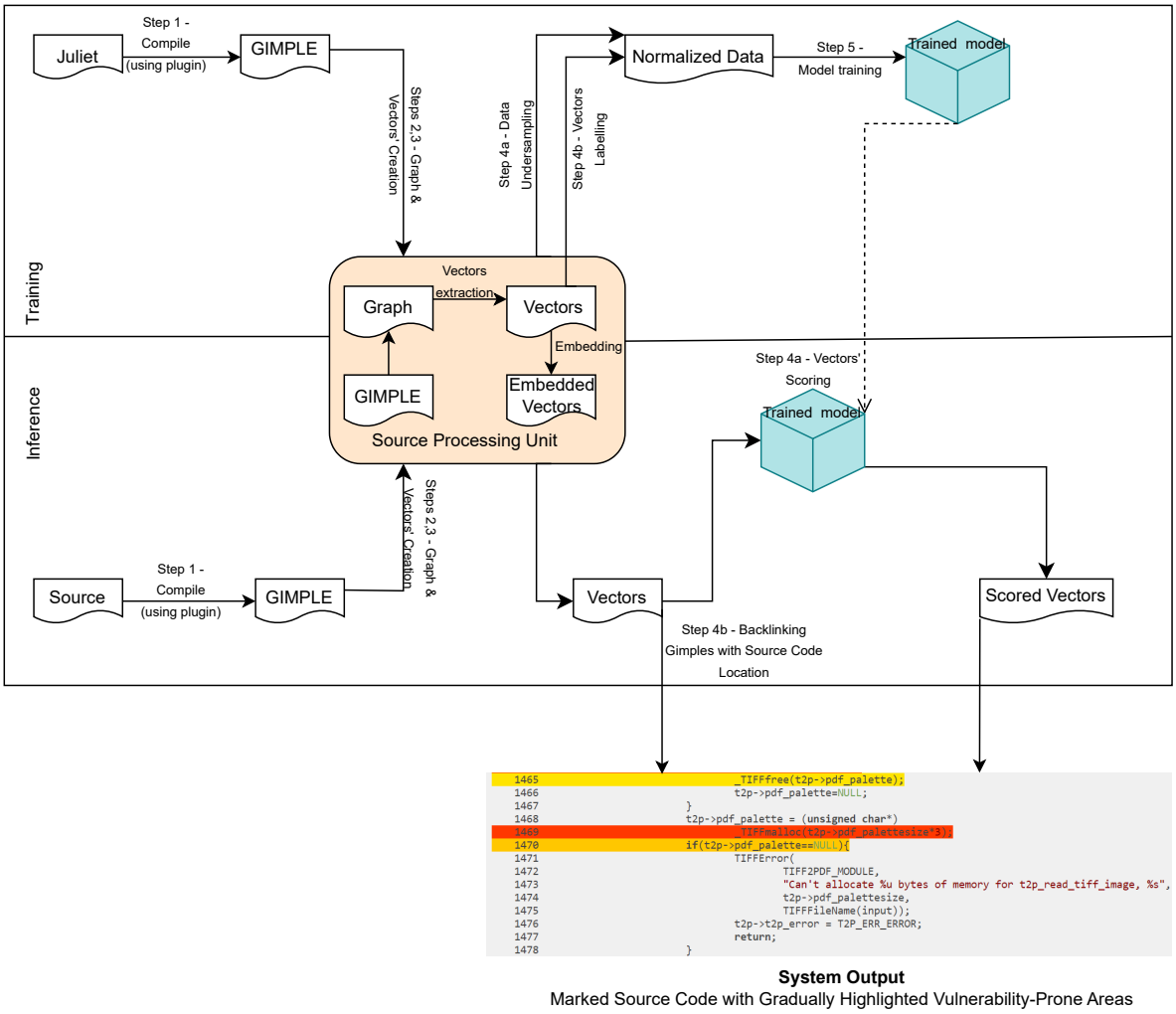


Figure 5.4: Shmulik Data Flow. Output example is code snippet from tiff2pdf.c showing silently patched vulnerability detected by Shmulik system in libtiff-3.9.2 (possible integer overflow of size for malloc, corresponding to CWE190)

Customized Compilation

Unlike in the standard compilation process, the main purpose of the Shmulik source compilation is not the executables, but the IL files. For instance, the GIMPLE representation [205]. It could be achieved by providing the *-fdump-tree-optimized-raw* flag and also by running the GCC plugin we implemented, which will preserve the interconnection between each GIMPLE command and the corresponding source code line.

GIMPLE is a three-address representation used by GCC compiler. Each GIMPLE is equivalent to one operation in the C source code. So, each line of code will be represented by one or more

equivalent GIMPLEs. An example of GIMPLE commands can be found in Fig. 5.5.

To achieve this purpose, our compiler plugin is invoked in the GIMPLE pass compilation phase and creates a log file (see Figure 5.5) that contains for each function all its GIMPLEs and the corresponding C file and line number.

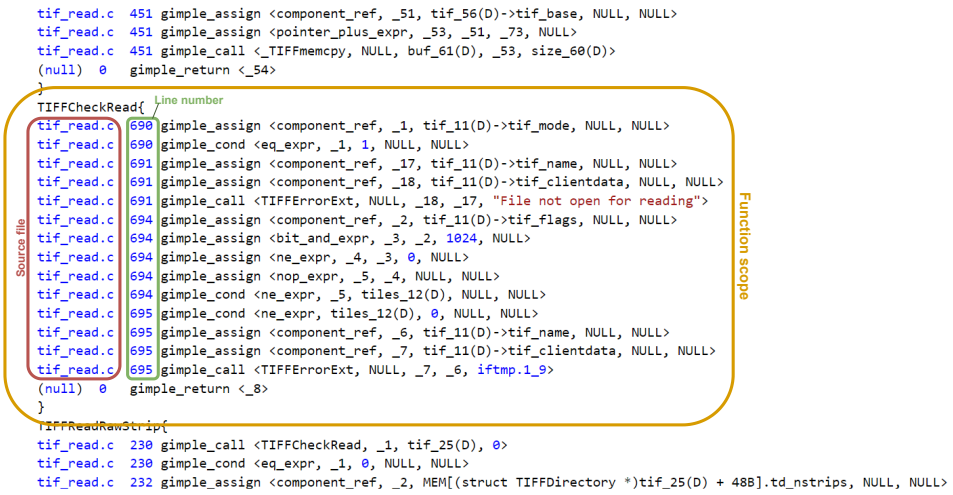


Figure 5.5: Plugin output sample on libtiff compilation

Code Flow Graph

Shmulik is designed to find bugs and vulnerabilities related to the control flow within the source code. To create an accurate representation of the code flow, the first step is to represent the code as a directed graph. To convert the C source code into a graph, Shmulik uses the GCC GIMPLE intermediate language obtained in the preceding step (see subsection 5.2.3. As explained earlier, the trace of the code flow begins from designated entry points. For API functions, this tracing originates from the function definition itself, while for non-API functions, it initiates from all calls to that particular function. The flow then progresses through the next GIMPLE operations, performing manipulations on the specified relevant arguments.

Algorithm 1 provides a high-level overview of the graph building process for a given entry point E. As mentioned above, the root GIMPLE and the relevant argument are determined based on the entry point definition (API or non-API). Afterward, Algorithm 1 computes the next GIMPLE according to the current one and to the specified relevant argument. For example, in case our current GIMPLE is a call action, the next GIMPLE will be the first GIMPLE

of the callee function only if any of the function arguments is relevant. Otherwise, it will be the GIMPLE following the call GIMPLE. This process continues until there are no more GIMPLE to process. During this process, any relevant GIMPLE is added to the graph G, and the relevant argument is updated based on the current GIMPLE. For example, if the current GIMPLE does an assignment, the variable being assigned is now considered relevant.

Algorithm 1 Code Flow Graph Building

Input: entry point E, list of functions F
Output: code flow graph G

1: $g \leftarrow Gimple(E)$
2: $relevantArg \leftarrow RelevantArgument(E)$
3: $g_{next} \leftarrow NextGimple(g, F, relevantArg)$
4: **while** $g_{next} \neq null$ **do**
5: **if** $isRelevant(g_{next}, relevantArg)$ **then**
6: $update(relevantArg, g)$
7: $G \leftarrow addEdge(g, g_{next})$
8: **end if**
9: $g \leftarrow g_{next}$
10: $g_{next} \leftarrow NextGimple(g, F, relevantArg)$
11: **end while**
12: **return** G

Vectors Extraction

After generating the code flow graph, Shmulik extracts valid sequences (vectors) from the graph using the Node2Vec algorithm [206]. This algorithm is chosen due to its ability to handle cycles in the graph (that could have led to infinite loops in the analysis process), ensuring each vector's extraction is a finite process.

Node2Vec algorithm itself consists of two steps: random graph walks (the first step) and vector embedding. Shmulik performs only the first step and continues each walk until it meets one of the following conditions: a) the walk reaches a graph's leaf, or b) the vector reaches a predetermined maximum length. The second Node2Vec step is skipped, and instead, we use One-Hot Encoding to convert GIMPLE corresponding strings from Node2Vec-extracted vectors into numerical representations.

Machine Learning Model

Despite the recent trend of using LLMs for vulnerability detection, as presented in Chapter 4, we chose not to employ LLMs in our method due to their limitations in handling complex real-world datasets. As noted in [196], while LLMs have demonstrated proficiency in simplified scenarios, their performance drops when dealing with more intricate real-world datasets or large codebases. This suggests that while LLMs have potential in certain scenarios, they may require enhancements to effectively process complex projects.

We selected a Bi-LSTM (Bidirectional Long Short-Term Memory) neural network model for the machine learning model trained at this stage. Originally proposed for phoneme classification [207], it has proved to be very efficient in other domains. Our choice is motivated by several reasons. Since Shmulik operates on source code, which can be seen as sequential data, it makes sense to use a model well-suited to handle sequential data to capture patterns over time. Furthermore, this model is also adequate to handle variable-length sequences and preserve context (past and future). It has also been studied and proven its good performance on different kinds of sequential data analysis [208, 209, 210].

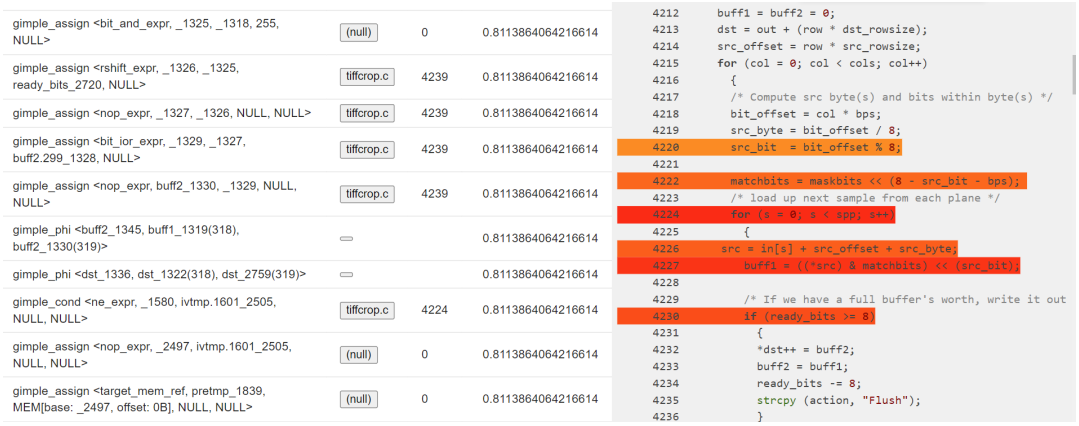


Figure 5.6: Vector view of CVE-2016-5321/5323

5.2.4 System Output

Shmulik's system output is designed to be easily interpretable by human agents, enabling them to quickly recognize code areas that are more likely to contain bugs. To achieve that, we

create an interactive web interface, which presents the scores vectors —the output of Shmulik's system—in various ways and links the score vectors back to the C source code. The output consists of two main components: the vector's score and the GIMPLE score.

- Vector's Score: The prediction of the model for specific vector. A higher score indicates a higher probability of the vector being buggy and potentially vulnerable.
- GIMPLE's Score: The average of the scores of all vectors that contain the same GIMPLE. This score provides an overall assessment of the potential vulnerability associated with a particular GIMPLE.

To ensure that Shmulik's results can be easily understood by human agents, the system also includes a presentation layer, which offers two types of views: vector-based and file-based.

Vector-Based View

This view presents a list of scored vectors, optionally accompanied by some code fragments (see Fig. 5.6) . Each line in the left table corresponds to a vector that can be expanded to see the list of GIMPLEs of the same vector. This list includes the source file name, source code line number, and the score of each GIMPLE.

Clicking on the file name opens the source file (on the right), and the lines referenced by the same vector are color-coded based on their GIMPLE's score, from yellow (benign) to dark-red (malicious). Lines that remain uncolorized are not part of the analyzed flow for this vector.

Aggregated View

In this *Aggregated View*, the source code is presented, with each line colored based on the score of its GIMPLEs (if there are numerous GIMPLEs derived from the same line, the color is determined by the highest score among them). Example of this view is shown in Fig. 5.7, where the line highlighted in red precisely indicates the code that was fixed to address CVE-2016-5321/CVE-2016-5323.

While both views demonstrate the same source code area, more source code lines are usually colored in the aggregated view, since all lines referenced in any vectors are colorized and not

```
4212      buff1 = buff2 = 0;
4213      dst = out + (row * dst_rowsize);
4214      src_offset = row * src_rowsize;
4215      for (col = 0; col < cols; col++)
4216        {
4217        /* Compute src byte(s) and bits within byte(s) */
4218        bit_offset = col * bps;
4219        src_byte = bit_offset / 8;
4220        src_bit  = bit_offset % 8;
4221
4222        matchbits = maskbits << (8 - src_bit - bps);
4223        /* load up next sample from each plane */
4224        for (s = 0; s < spp; s++)
4225          {
4226       src = in[s] + src_offset + src_byte;
4227          buff1 = ((*src) & matchbits) << (src_bit);
4228
4229          /* If we have a full buffer's worth, write it out */
4230          if (ready_bits >= 8)
4231            {
4232            *dst++ = buff2;
4233            buff2 = buff1;
4234            ready_bits -= 8;
4235            strcpy (action, "Flush");
4236            }
```

Figure 5.7: Aggregated view for a libtiff code snippet showcasing CVE-2016-5321/5323

only those referenced by a specific vector as in vector view. This view enables the researcher to visually and intuitively identify which parts of the code and which flows to focus their attention on. If necessary, he can switch to a specific vector-based view to help focus on a specific flow, as explained.

In summary, Shmulik's system output, combined with the presentation layer, provides developers and researchers with a powerful tool for identifying and prioritizing potential vulnerabilities in their code, ultimately improving code quality and security.

5.3 Experimental Analysis and Evaluation

In this section, we present the experiments and evaluations conducted to assess how effectively Shmulik fulfills the contributions and objectives outlined earlier, particularly in terms of software vulnerability detection and its practical utilization.

We implemented Shmulik data processing and Machine Learning functionality in Python 3 using Tensorflow [211], while transformer models were fine-tuned using PyTorch [212] and Hugging Face [213] pre-trained GPT2 model [214]; the GCC plugin was written in C++, version 9.4.0; the user interface was implemented in JavaScript using the MERN stack [215]. All the process was performed on a 32-core (Intel Xeon Silver 4208) machine with 2 NVIDIA A40 GPUs running Ubuntu 20.04 OS.

5.3.1 Dataset Selection

For our experiments, we selected the Juliet dataset, part of the SARD dataset created by the NSA [204]. This extensive dataset of C/C++ code comprises a considerable number of examples, providing a rich and (almost) balanced repository of both vulnerable and non-vulnerable source code snippets. These code samples are meticulously labeled, with each file containing one function labeled as "good" and another labeled as "bad". Our choice of this dataset was primarily guided by its comprehensive coverage of diverse source code vulnerabilities and its easy-to-process labeling system. Notably, Juliet's organization by CWE (Common Weakness Enumeration) sets it apart from other corpora used in related projects. This organizational approach plays a pivotal role, as elaborated in the following. In addition, the Juliet dataset is commonly used in research [186, 185, 216, 187], enabling Shmulik to benchmark with the state-of-the-art work. Furthermore, we opted for Juliet over other datasets such as [217], which were created by applying legacy static analyzers to large code bases. The decision was influenced by our preference for a code snippet-centric approach, as opposed to training our model solely on vulnerabilities previously identified by other software programs. This approach broadened our ability to detect nuanced vulnerabilities and zero-day exploits effectively.

5.3.2 Evaluation Methodology

As detailed in Section 5.2.3, Shmulik's model selection process involved training a BiLSTM neural network.

We experimented with two different approaches for CWE specificity of the model. The first approach was training an ensemble of multiple models, when each model was trained to identify single CWE. When each of the ensemble models outputs the probability of a certain vector containing CWE-specific bug, the total classification was an average of maximum 3 probabilities of the models to be vulnerable to this CWE. Formally, ensemble classification is performed as follows:

$$Score_{Ensemble}(V) = \frac{\sum_{i=1}^{3} SCORES_i}{3},$$

where $SCORES = \{s_1 \geq s_2 \geq ... \geq s_7\}$ for $s_i \in (Score_{CWE}(V) : CWE \in Ensemble)$.

The second approach we checked was building a single global model to recognize the existence of any CWEs from a selected CWE group. Both the ensemble and the global model's goal was to make a binary classification of benign or malicious labels. The experiment was conducted over buffer overflow CWEs (namely: CWE121, CWE122, CWE124, CWE126, CWE127, CWE680, CWE761 - as buffer-related CWEs that have enough linux-compatible representation in Juliet dataset). For each CWE, after extracting the vectors, undersampling was performed. Then, randomly chosen 10% malicious records of each CWE's dataset and 10% of benign vectors of each CWE's dataset were used as a evaluation set, while the remaining 90% of the vectors were used to train the evaluated models. We can formalize the train-test split as follows:

$$Trainset(CWE_i) = 0.9 \times CWE_{i_{benign}} + 0.9 \times CWE_{i_{malicious}}$$

$$Trainset(global) = \sum Trainset(CWE_i)$$

$$EvaluationSet = \sum (0.1 \times CWE_{i_{benign}} + 0.1 \times CWE_{i_{malicious}})$$

In this way, we have achieved (1) totally balanced trainsets and evaluation set from positive-negative point of view and (2) proportional representation of CWEs' samples in global model's

train- and evaluation set (assuming that there is a correlation between the frequency of the CWE in real world and its representation in Juliet dataset). All of the models were evaluated on the same evaluation set. To visualize the results of the experiment, we have plotted the ROC curves of multiple CWE model and of the ensemble single CWE models in figure 5.8. The X-axis corresponds to the false positive rate (FPR, FPR = $\frac{\text{FP}}{\text{FP+TN}}$), and the value by Y-axis corresponds to the true positive rate (TPR, TPR = $\frac{\text{TP}}{\text{TP+FN}}$). The blue line corresponds to the ROC of global CWE model on the evaluation set. The pink line, on the other hand, describes the ROC of 7 ensembles of CWE models, when the predicted value for each test set vector is calculated as described above (the mentioned threshold of 3 highest scores was chosen empirically when each of these models was trained on a very specific bug, there would be groups of bugs that are very similar and sometimes the same bug could be detected by several models). As we can see from this plot, multiple-CWE model outperforms ensemble-CWE models for our evaluation set containing samples marked by different CWEs. We assume that this evaluation method is reliable as real-world projects usually contain different kinds of bugs together, not limited by a certain CWE. To conclude, we assume that a single model trained to recognize a number of different yet related CWE bugs is more practical than a number of models each trained to recognize specific CWE.

5.3.3 Bridging AI and Human Expertise: Case study

To evaluate the effectiveness of our approach of creating a hybrid methodology and system for the detection of vulnerabilities, we conducted a case study. The case study involved a team of human code reviewers who were trained to use the Shmulik system. They utilized Shmulik to analyze the code and identify potential vulnerabilities. The code reviewers then reviewed and verified the results generated by the Shmulik system.

We applied Shmulik to the libtiff source code to evaluate its effectiveness in detecting known vulnerabilities and uncovering previously undetected vulnerabilities. This library has been chosen due to its high popularity in both open source and commercial projects. While being frequently used, this library has a history of security flaws being exploited in popular devices, such as the first iPhone jailbreak and Sony PSP exploit[218, 219]. In addition, each

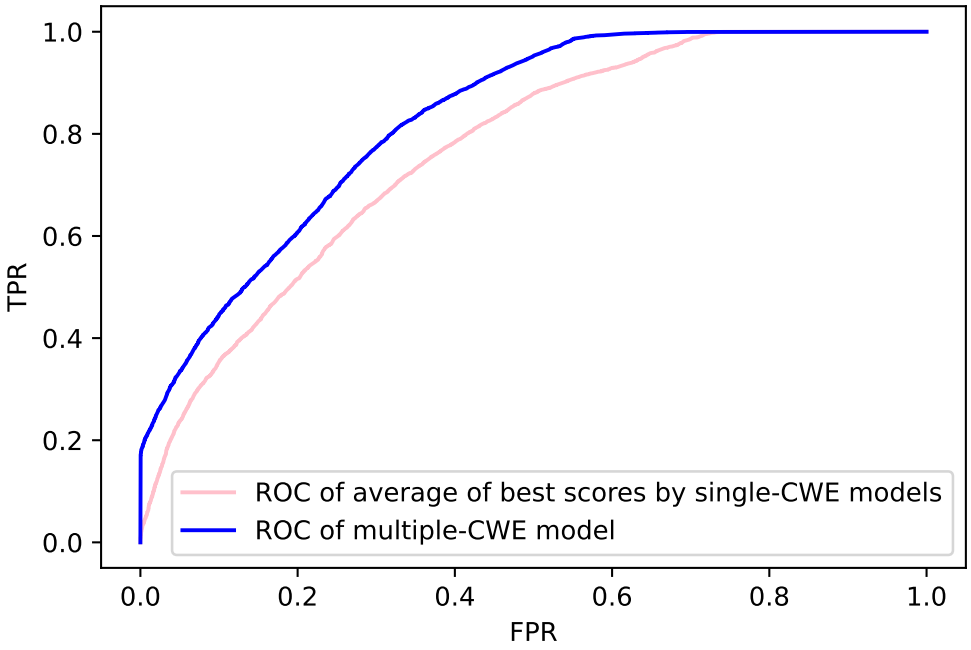


Figure 5.8: Models' ROC comparison for ensemble 7 single-CWE models and multiple CWEs model for CWEs belonging to the same family (improper buffer bounds restriction)

build of libtiff is automatically scanned by state-of-the-art static analyser, Coverity[220, 221]. Detecting known vulnerabilities is a known benchmarking practice in automatic vulnerability detection research, used to compare similar systems, and tested on libtiff version 3.9.2, Shmulik detected 5 known CVEs, 1 oCERT issue and 9 silent patches.

But as stated in Section 5.1.1, our goal is not to supplant other systems but complete them, and deter new vulnerabilities. As so, while most of the tools, such as [185, 187, 222], have only been evaluated by recognising previously known vulnerabilities (comparing reported vulnerabilities or silent-patches), Shmulik was used to discover previously unknown ("zero-day") vulnerabilities, on private systems and on open source projects. Namely, in libtiff version 4.5.1, Shmulik reported CVE-2023-41175 and CVE-2023-40745 [223, 224], showcasing its potential in identifying new security risks. Worth mentioning that these bugs have been present in the codebase for respectively 20 and 24 years. To demonstrate that these are not only theoretical bugs but exploitable vulnerabilities, we presented functional proof-of-concepts (working PoCs), and responsively contributed the relevant fixes[1].

[1]Available at `https://gitlab.com/libtiff/libtiff/-/issues/591` and `https://gitlab.com/libtiff/`

5.3.4 Prioritizing Resources for Effective Code Review

In section 5.2.4, we presented the concepts of Vector View and Aggregated View as the outputs of the Shmulik system.

The Aggregated View offers a broader overview of the codebase by coloring lines of code based on the score of their associated GIMPLEs. This view can highlight larger portions of the code, and provides a high-level view of potential areas of concern within the code. Reviewers can use the Aggregated View to identify general trends or patterns of vulnerability distribution across the codebase.

In such contexts, the Aggregated View helps teams identify broader trends and areas that appear more susceptible to vulnerabilities. This initial, macro-level assessment guides teams to focus their attention on these potentially problematic sections.

Following this initial assessment, teams can then delve deeper into specific areas using the Vector View. This detailed perspective highlights specific code segments associated with potential vulnerabilities and scores them based on their likelihood of containing issues. By employing the Vector View after the Aggregated View, reviewers can efficiently pinpoint and investigate specific vulnerabilities within the previously identified areas. This combined approach ensures that reviewers and research teams first prioritize areas that seem more prone to finding bugs and then meticulously pinpoint and address them. It not only enhances the efficiency of the code review process but also ensures that critical vulnerabilities are addressed promptly, making it a reliable and trustable method for vulnerability detection and mitigation.

5.3.5 Enriched Learning Platform: The vulnerability type-specific Lenses

Based on our observations presented in section 5.3.2, we explored how Shmulik serves as an enriched learning platform, offering a unique perspective on vulnerability identification and a pedagogical approach to cybersecurity education and training.

To illustrate this concept, we conducted an experiment involving vulnerable execution flows in the libtiff library. Within a code snippet, we strategically inserted a line vulnerable to an

`libtiff/-/issues/592`

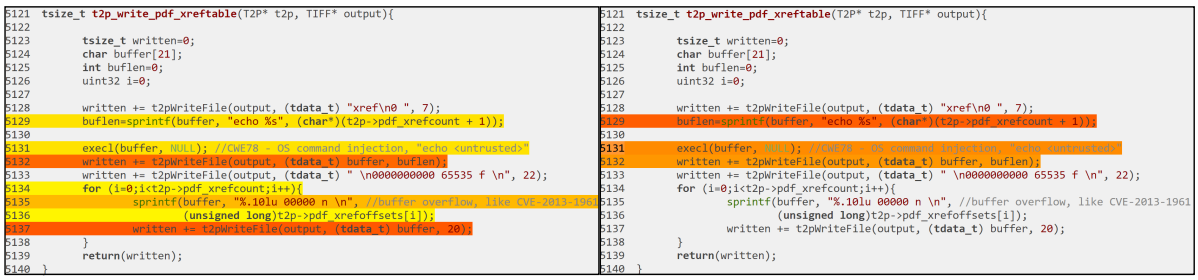


Figure 5.9: Comparison of file highlighting for two different CWEs for the same function: CWE-121 (Buffer Stack Overflow) on the left and CWE-78 (OS Command Injection) on the right.

OS command injection attack (CWE-78) amidst code that exhibits susceptibility to Buffer Stack Overflow (CWE-121) - based on CVE-2013-1961 (stack-based buffer overflow is the reported kind of vulnerability for this CVE). In Figure 5.9, we present the code, color-coded differently to distinguish between these two CWEs. On the left, the highlighting reflects the flows susceptible to CWE-121, while on the right, the part relevant to CWE-78 is colored. This unique perspective, akin to viewing the code through "CWE-specific lenses," not only assists researchers in vulnerability identification but also offers a pedagogical approach to training them in recognizing and mitigating diverse security issues. In addition, we can conclude that even CWE-specific model, which is less accurate than global model for a family of similar CWEs, as stated in section 5.3.2, performs well at recognizing vulnerabilities in real-world projects.

In the realm of cybersecurity education and training, it is often recommended to focus on teaching and learning about separate vulnerability categories individually. This approach aligns with well-established pedagogical practices, recognizing that humans, like machine learning models, benefit from dedicated immersion in specific subject areas. Training security researchers on one category at a time helps them become experts in each type of vulnerability. This way, they gain a deep understanding and valuable insights. This approach has been adopted by popular commercial training platforms [225, 226], but while they require manual preparation of the code base that trainees work on, Shmulik paves the way for some automatic preparation of categorized code, even based on real code bases.

Feature	**Shmulik**	**[186]**	**[185]**	**[187]**	**[184]**	**[227]**
Exploitably flow tracking	✓	✗	✗	✓	✗	✗
Full flow tracing	✓	✗	✗	✗	✗	✗
Proven performance on full codebase for real-world project	✓	✓	✗	✗	✓	✗
Syntax-independent	✓	✗	✗	✓	✗	✗
Heat-map source code colorization	✓	✓	✗	✓	✗	✗
Compilation-time backlinking intermediate representation to source code lines	✓	N/A No compilation	✗	N/A Binary targets	N/A No compilation	N/A No compilation
CWE lenses/ vulnerability kind detection	✓	✗	✓	✓	✓	✗
Integrated source code viewer	✓	✓	✗	✗	✗	✗
IDE integration	✓	✗	✗	✗	✗	✗
Confirmed 0-days detection	✓	✓?	✗	✗	✗	✗

Table 5.1: Features comparison of ML-based static analyzers

5.4 Features Comparison of Machine Learning-Based Static Analyzers

In the domain of static analysis for vulnerability detection, various machine learning (ML)-based tools exist, each offering distinctive capabilities. This section presents a comparative overview of the key features of several ML-based static analyzers, including Shmulik and the other prominent tools introduced in Section 4.4. This comparison aims to underscore the areas where Shmulik excels, demonstrating its unique advantages over other tools in the field.

Table 5.1 summarizes and contrasts the primary features of several ML-based static analyzers, emphasizing Shmulik's advantages:

The table offers a comprehensive comparison of the features of various ML-based static analyzers. Shmulik's strengths are particularly evident in its robust performance in flow tracking, syntax independence, heatmap visualization, and precise vulnerability detection. By combining advanced AI methodologies with human expertise, Shmulik emerges as a highly versatile tool, capable of addressing the needs of both small-scale and large-scale software development projects.

5.5 Challenges

Shmulik presents several challenges in its quest for enhanced vulnerability detection. One such challenge revolves around defining entry points, crucial for tracking the control flow. While it is relatively straightforward in cases involving command-line parameters or regular I/O APIs, complications arose when inputs were sourced from shared memory. The precise definition of the initiation of code flow in these scenarios remains an ongoing research area.

Furthermore, as presented in Section 5.2.4, Shmulik faces the task of accurately pinpointing the precise source code lines with vulnerabilities. Inference is performed at the numerical representation of vectors, and, while Shmulik effectively identifies areas of concern, mapping these back to exact source code lines can be intricate. Our GCC plug-in keeps the information connecting each GIMPLE with its corresponding C file, but in some cases, some information is still lacking to get back from the full vector to the precise line of code. Future improvements may involve embedding more contextual information within the code flow graph to enhance this mapping process.

5.6 Conclusion

In this chapter, we introduced Shmulik, a deep learning-based vulnerability detection system operating on software programs source code, leveraging intermediate representations (IR). Our aim was to contribute to the field of automatic vulnerability detection in software.

We recognized the challenge of false positives in existing systems, hindering their practical application. Our efforts focused on refining Shmulik to improve code reviews by researchers and developers. By highlighting code sections by risk levels, Shmulik aids in identifying potential vulnerabilities, facilitating prioritized review efforts.

Furthermore, Shmulik offers support beyond bug detection, assisting code review practices. It provides insights into project areas with higher vulnerability likelihood, aiding in resource allocation.

Our evaluation, conducted on a dataset designed for deep learning and open-source code,

revealed significant findings. We identified and reported several zero-day vulnerabilities, highlighting Shmulik's effectiveness in uncovering previously unknown security risks.

In conclusion, Shmulik demonstrates the potential of deep learning and IR for proactive vulnerability detection. Its focus on code review enhancement and prioritization guidance holds promise for developers and researchers. Our work represents a step forward in software security, acknowledging the ongoing evolution of the field.

Chapter 6

Conclusion

In conclusion, this thesis has embarked on an exploratory journey through the domains of Wireless Sensor Networks (WSN) and software vulnerability detection, converging on the common goal of enhancing IoT security. The research presented herein has systematically addressed the dual challenges of resource constraints in WSNs and the need for early detection of software vulnerabilities.

The first part of the thesis began with a comprehensive overview of message authentication schemes, providing valuable insights into the evolution of authentication techniques in resource-constrained environments. This was followed by a novel lightweight hybrid authentication scheme, specifically designed for power-constrained WSNs, demonstrating that security should not be sacrificed for efficiency. This scheme represents a significant advancement in the field and offers a viable solution to the ongoing problem of securing WSNs without compromising their operational longevity.

The second part of the thesis detailed the development of Shmulik, a deep learning-based system for the detection of software vulnerabilities. Shmulik exemplifies the potential of machine learning techniques to revolutionize the vulnerability detection process, providing a proactive approach to software security that can be integrated early in the development lifecycle. Notably, Shmulik successfully detected zero-day vulnerabilities, demonstrating its effectiveness in identifying previously unknown threats. Shmulik's ability to augment human code review makes it an invaluable tool for ensuring the security and integrity of software development, especially in resource-constrained devices, where its specialized focus on C code addresses a

critical need in the field of IoT security.

The interplay between the lightweight authentication scheme and Shmulik underscores a critical insight: robust security in the IoT era requires a multifaceted approach. The authentication scheme ensures the integrity of communication channels, while Shmulik enhances the security of the software itself. Together, they offer a comprehensive defense against the diverse threats faced by WSNs.

This thesis contributes to the body of knowledge by bridging the gap between energy-efficient security protocols and intelligent vulnerability detection systems. The findings suggest that the integration of these two approaches can lead to the development of more secure, reliable, and resilient WSNs, which are essential for the trustworthiness of IoT systems.

As we advance, it is imperative that the research community continues to build on the foundations laid by this work, seeking innovative solutions to the complex security challenges of our increasingly connected world. The methodologies and insights presented in this thesis are hoped to inspire further research and development in the field of IoT security.

Future works may explore other security dimensions such as lightweight mitigations against exploitation, physical security measures, intrusion detection systems, and the integration of artificial intelligence for predictive threat analysis. These areas represent the next steps in evolving the security infrastructure of WSNs and IoT devices, building upon the foundational work presented in this thesis.

References

[1] Xiaotao Feng, Xiaogang Zhu, Qing-Long Han, Wei Zhou, Sheng Wen, and Yang Xiang. Detecting Vulnerability on IoT Device Firmware: A Survey. *IEEE/CAA Journal of Automatica Sinica*, 10(1):25–41, January 2023. ISSN 2329-9274. doi: 10.1109/JAS.2022.105860. URL `https://ieeexplore.ieee.org/document/9878283`. Conference Name: IEEE/CAA Journal of Automatica Sinica.

[2] Darktrace. Global threat report 2017, 2017. URL `https://www.darktrace.com/resources/#white-papers`.

[3] J. Margolis, T. T. Oh, S. Jadhav, Y. H. Kim, and J. N. Kim. An in-depth analysis of the mirai botnet. In *2017 international conference on software security and assurance (ICSSA)*, pages 6–12, 2017.

[4] Amin Hassanzadeh, Amin Rasekh, Stefano Galelli, Mohsen Aghashahi, Riccardo Taormina, Avi Ostfeld, and M. Katherine Banks. A review of cybersecurity incidents in the water sector. *Journal of Environmental Engineering*, 146(5):03120003, 2020.

[5] Mohammad S Jalali, Bethany Russell, Sabina Razak, and William J Gordon. EARS to cyber incidents in health care. *Journal of the American Medical Informatics Association*, 26(1):81–90, November 2018. ISSN 1527-974X. doi: 10.1093/jamia/ocy148.

[6] Muhammed Zekeriya Gunduz and Resul Das. Cyber-security on smart grid: Threats and potential solutions. *Computer Networks*, 169, 2020. Publisher: Elsevier.

[7] C. Kolias, A. Stavrou, J. Voas, I. Bojanova, and R. Kuhn. Learning internet-of-things security "Hands-On". *IEEE Security & Privacy*, 14(1):37–46, 2016.

[8] M. Radovan and B. Golub. Trends in IoT security. In *2017 40th international convention on information and communication technology, electronics and microelectronics (MIPRO)*, pages 1302–1308, 2017.

[9] Mardiana binti Mohamad Noor and Wan Haslina Hassan. Current research on Internet of Things (IoT) security: A survey. *Computer Networks*, 148:283–294, January 2019. ISSN 1389-1286. doi: 10.1016/j.comnet.2018.11.025. URL `https://www.sciencedirect.com/science/article/pii/S1389128618307035`.

[10] B. D. Davis, J. C. Mason, and M. Anwar. Vulnerability studies and security postures of IoT devices: A smart home case study. *IEEE Internet of Things Journal*, pages 1–1, 2020.

[11] Inmarsat Research Programme. The future of IoT in enterprise, 2017. URL `https://www.inmarsat.com/content/dam/inmarsat/corporate/documents/enterprise/insights/Inmarsat_WP_Future_IoT_Enterprise_2017.pdf`.

[12] Aida Boghossian, Scott Linksy, Alicia Brown, Peter Mutschler, Brian Ulicny, Larry Barrett, Glenn Bethel, Michael Matson, Thomas Strang, Kelly Wagner Ramsdell, and others. Threats to precision Agriculture–Homeland security. *A study supported by the United States Office of the Director of National Intelligence (ODNI) and the Department of Homeland Security (DHS) October*, 3:2018, 2018.

[13] Bharat S. Chaudhari, Marco Zennaro, and Suresh Borkar. LPWAN technologies: Emerging application characteristics, requirements, and design considerations. *Future Internet*, 12(3), 2020. ISSN 1999-5903. tex.article-number: 46.

[14] US Congress. IoT Cybersecurity Improvement Act of 2020, 2020. URL `https://www.congress.gov/bill/116th-congress/house-bill/1668/text`. Publication Title: Congress.gov.

[15] DCMS. Code of practice for consumer IoT security. 2018. Publisher: HMG Department for Digital, Culture, Media and Sport London, UK.

[16] Bundesamt für Sicherheit in der Informationstechnik (BSI). BSI übergibt erstes IT-Sicherheitskennzeichen, February 2022. URL https://www.bsi.bund.de/DE/Service-Navi/Presse/Pressemitteilungen/Presse2022/220201_erstes-IT-Sicherheitskennzeichen.html;jsessionid=D4CDC654B28E072FB4EA0257178419F6.internet462. Publication Title: Bundesamt für Sicherheit in der Informationstechnik.

[17] Department of Home Affairs Australian Government. Code of Practice, Securing the Internet of Things for Consumers. page 10, 2020.

[18] Singapore Infocomm Media Development Authority. Internet of Things (IoT) Cyber Security Guide, 2020. URL https://www.imda.gov.sg/-/media/Imda/Files/Regulation-Licensing-and-Consultations/ICT-Standards/Telecommunication-Standards/Reference-Spec/IMDA-IoT-Cyber-Security-Guide.pdf.

[19] Department of Home Affairs Australian Government. Voluntary Code of Practice, 2021. URL https://www.homeaffairs.gov.au/reports-and-publications/submissions-and-discussion-papers/code-of-practice#content-index-1.

[20] Department for Digital, Culture, Media and Sport and Department for Business, Energy and Industrial Strategy, National Cyber Security Centre. Regulation of consumer connectable product cyber security: impact assessment, 2021. URL https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/1074182/Impact_Assessment.pdf.

[21] Parliament: House of Lords. Product Security and Telecommunications Infrastructure (PSTI) Bill, 2022. URL https://bills.parliament.uk/bills/3069.

[22] Nabil Sayfayn and Stuart Madnick. Cybersafety Analysis of the Maroochy Shire Sewage Spill. *Interdisciplinary Consortium for Improving Critical Infrastructure Cybersecurity, MIT Sloan*, May 2017. URL https://cams.mit.edu/wp-content/uploads/2017-09.pdf.

[23] Kaspersky ICS CERT. H1 2022 – a brief overview of the main incidents in industrial cybersecurity. 2022. URL https://ics-cert.kaspersky.com/publications/reports/2022/09/08/h1-2022-a-brief-overview-of-the-main-incidents-in-industrial-cybersecurity.

[24] Neeraj Chandnani and Chandrakant N Khairnar. An analysis of architecture, framework, security and challenging aspects for data aggregation and routing techniques in IoT WSNs. *Theoretical Computer Science*, 929:95–113, September 2022. ISSN 0304-3975. doi: 10.1016/j.tcs.2022.06.032. URL https://www.sciencedirect.com/science/article/pii/S0304397522004066.

[25] Akber Ali Khan, Vinod Kumar, Musheer Ahmad, and Saurabh Rana. LAKAF: Lightweight authentication and key agreement framework for smart grid network. *Journal of Systems Architecture*, 116:102053, June 2021. ISSN 1383-7621. doi: 10.1016/j.sysarc.2021.102053. URL https://www.sciencedirect.com/science/article/pii/S1383762121000461.

[26] Dipanwita Sadhukhan, Sangram Ray, Mohammad S. Obaidat, and Mou Dasgupta. A secure and privacy preserving lightweight authentication scheme for smart-grid communication using elliptic curve cryptography. *Journal of Systems Architecture*, 114: 101938, March 2021. ISSN 1383-7621. doi: 10.1016/j.sysarc.2020.101938. URL https://www.sciencedirect.com/science/article/pii/S1383762120301958.

[27] Eric Rescorla. Diffie-Hellman Key Agreement Method. Request for Comments RFC 2631, Internet Engineering Task Force, June 1999. URL https://datatracker.ietf.org/doc/rfc2631. Num Pages: 13.

[28] Elaine Barker, Lily Chen, Allen Roginsky, Apostol Vassilev, and Richard Davis. Recommendation for pair-wise key-establishment schemes using discrete logarithm cryptography. Technical Report NIST SP 800-56Ar3, National Institute of Standards and Technology, Gaithersburg, MD, April 2018. URL https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf.

[29] Elaine Barker, Lily Chen, Allen Roginsky, Apostol Vassilev, Richard Davis, and Scott Simon. Recommendation for pair-wise key establishment using integer factorization cryptography. Technical Report NIST SP 800-56Br2, National Institute of Standards and Technology, Gaithersburg, MD, March 2019. URL https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Br2.pdf.

[30] Peter Williston Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, November 1994. doi: 10.1109/SFCS.1994.365700.

[31] Joppe Bos, Leo Ducas, Eike Kiltz, T Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehle. CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 353–367, April 2018. doi: 10.1109/EuroSP.2018.00032.

[32] Information Technology Laboratory Computer Security Division. Announcing PQC Candidates to be Standardized, Plus Fourth Round Candidates | CSRC, March 2022. URL https://csrc.nist.gov/News/2022/pqc-candidates-to-be-standardized-and-round-4.

[33] National Security Agency (NSA). National Security Agency/Central Security Service > Cybersecurity > Quantum Key Distribution (QKD) and Quantum Cryptography QC, October 2020. URL https://www.nsa.gov/Cybersecurity/Quantum-Key-Distribution-QKD-and-Quantum-Cryptography-QC/.

[34] Agence nationale de la sécurité des systèmes d'information (ANSSI). Should Quantum Key Distribution be Used for Secure Communications?, May 2020. URL https://www.ssi.gouv.fr/publication/should-quantum-key-distribution-be-used-for-secure-communications/.

[35] UK National Cyber Security Centre(NCSC). Quantum security technologies, March 2020. URL https://www.ncsc.gov.uk/whitepaper/quantum-security-technologies.

[36] European Union Agency for Cybersecurity (ENISA). Post-Quantum Cryptography: Current state and quantum mitigation. Report/Study, May 2021. URL https://www.enisa.europa.eu/publications/post-quantum-cryptography-current-state-and-quantum-mitigation.

[37] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Message Authentication using Hash Functions; The HMAC Construction. page 5, 1996.

[38] Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. Request for Comments RFC 8446, Internet Engineering Task Force, August 2018. URL https://datatracker.ietf.org/doc/rfc8446. Num Pages: 160.

[39] Eric Rescorla, Hannes Tschofenig, and Nagena Modadugu. The Datagram Transport Layer Security (DTLS) Protocol Version 1.3. Request for Comments RFC 9147, Internet Engineering Task Force, April 2022. URL https://datatracker.ietf.org/doc/rfc9147. Num Pages: 61.

[40] Marc Petit-Huguenin and Gonzalo Salgueiro. Multiplexing Scheme Updates for Secure Real-time Transport Protocol (SRTP) Extension for Datagram Transport Layer Security (DTLS). Request for Comments RFC 7983, Internet Engineering Task Force, September 2016. URL https://datatracker.ietf.org/doc/rfc7983. Num Pages: 13.

[41] Osamu Honda, Hiroyuki Ohsaki, Makoto Imase, Mika Ishizuka, and Junichi Murayama. Understanding TCP over TCP: effects of TCP tunneling on end-to-end throughput and latency. In *Performance, Quality of Service, and Control of Next-Generation Communication and Sensor Networks III*, volume 6011, pages 138–146. SPIE, October 2005. doi: 10.1117/12.630496. URL https://www.spiedigitallibrary.org/conference-proceedings-of-spie/6011/60110H/Understanding-TCP-over-TCP--effects-of-TCP-tunneling-on/10.1117/12.630496.full.

[42] Ronald Rivest. The MD5 message-digest algorithm. Technical report, Internet Engineering Task Force, 1992.

[43] Alex Biryukov and Léo Perrin. State of the art in lightweight symmetric cryptography. *IACR Cryptology ePrint Archive*, 2017:511, 2017.

[44] Miles E. Smid. Development of the Advanced Encryption Standard. *Journal of Research of the National Institute of Standards and Technology*, 126:126024, August 2021. ISSN 2165-7254. doi: 10.6028/jres.126.024. URL https://nvlpubs.nist.gov/nistpubs/jres/126/jres.126.024.pdf.

[45] Matthew Robshaw. The eSTREAM Project. In Matthew Robshaw and Olivier Billet, editors, *New Stream Cipher Designs: The eSTREAM Finalists*, Lecture Notes in Computer Science, pages 1–6. Springer, Berlin, Heidelberg, 2008. ISBN 978-3-540-68351-3. doi: 10.1007/978-3-540-68351-3_1. URL https://doi.org/10.1007/978-3-540-68351-3_1.

[46] Dennis Agyemanh Nana Gookyi, Guard Kanda, and Kwangki Ryoo. NIST Lightweight Cryptography Standardization Process: Classification of Second Round Candidates, Open Challenges, and Recommendations. *Journal of Information Processing Systems*, 17(2):253–270, 2021. ISSN 1976-913X. doi: 10.3745/JIPS.03.0156. URL https://koreascience.kr/article/JAKO202116057023073.page. Publisher: Korea Information Processing Society.

[47] Meltem Sonmez Turan, Kerry McKay, Donghoon Chang, Cagdas Calik, Lawrence Bassham, Jinkeon Kang, and John Kelsey. Status Report on the Second Round of the NIST Lightweight Cryptography Standardization Process. Technical report, National Institute of Standards and Technology, July 2021. URL https://nvlpubs.nist.gov/nistpubs/ir/2021/NIST.IR.8369.pdf.

[48] Vishal A. Thakor, Mohammad Abdur Razzaque, and Muhammad R. A. Khandaker. Lightweight Cryptography Algorithms for Resource-Constrained IoT Devices: A Review, Comparison and Research Opportunities. *IEEE Access*, 9:28177–28193, 2021. ISSN 2169-3536. doi: 10.1109/ACCESS.2021.3052867. Conference Name: IEEE Access.

[49] Muhammad Rana, Quazi Mamun, and Rafiqul Islam. Lightweight cryptography in IoT networks: A survey. *Future Generation Computer Systems: the international journal of*

grid computing: theory, methods and applications, 129:77–89, April 2022. ISSN 0167-739X. doi: 10.1016/j.future.2021.11.011. Publisher: Elsevier.

[50] Mostafa Farhadi Moghadam, Mahdi Nikooghadam, Maytham Azhar Baqer Al Jabban, Mohammad Alishahi, Leili Mortazavi, and Amirhossein Mohajerzadeh. An Efficient Authentication and Key Agreement Scheme Based on ECDH for Wireless Sensor Network. *IEEE Access*, 8:73182–73192, 2020. ISSN 2169-3536. doi: 10.1109/ACCESS.2020.2987764. Conference Name: IEEE Access.

[51] Majid Alotaibi. An Enhanced Symmetric Cryptosystem and Biometric-Based Anonymous User Authentication and Session Key Establishment Scheme for WSN. *IEEE Access*, 6:70072–70087, 2018. ISSN 2169-3536. doi: 10.1109/ACCESS.2018.2880225. Conference Name: IEEE Access.

[52] Xue Li, Cheng Jiang, Dajun Du, Minrui Fei, and Lei Wu. A Novel Revocable Lightweight Authentication Scheme for Resource-Constrained Devices in Cyber-Physical Power Systems. *IEEE Internet of Things Journal*, pages 1–1, 2022. ISSN 2327-4662. doi: 10.1109/JIOT.2022.3221943. Conference Name: IEEE Internet of Things Journal.

[53] Jayaprakash Kar, Kshirasagar Naik, and Tamer Abdelkader. A Secure and Lightweight Protocol for Message Authentication in Wireless Sensor Networks. *IEEE Systems Journal*, 15(3):3808–3819, September 2021. ISSN 1937-9234. doi: 10.1109/JSYST.2020.3015424. Conference Name: IEEE Systems Journal.

[54] Shimon Even, Oded Goldreich, and Silvio Micali. On-Line/Off-Line Digital Signatures. In Gilles Brassard, editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, Lecture Notes in Computer Science, pages 263–275, New York, NY, 1990. Springer. ISBN 978-0-387-34805-6. doi: 10.1007/0-387-34805-0_24.

[55] Chandrashekhar Meshram, Agbotiname Lucky Imoize, Azeddine Elhassouny, Amer Aljaedi, Adel R. Alharbi, and Sajjad Shaukat Jamal. IBOOST: A Lightweight Provably Secure Identity-Based Online/Offline Signature Technique Based on FCM for Massive

Devices in 5G Wireless Sensor Networks. *IEEE Access*, 9:131336–131347, 2021. ISSN 2169-3536. doi: 10.1109/ACCESS.2021.3114287. Conference Name: IEEE Access.

[56] Lijang Yi, Guoqiang Bai, and Guozhen Xiao. Proxy multi-signature scheme: A new type of proxy signature scheme. *Electronics Letters*, 36(6):527–528, March 2000. ISSN 1350-911X. doi: 10.1049/el:20000422. URL https://digital-library.theiet.org/content/journals/10.1049/el_20000422. Publisher: IET Digital Library.

[57] Jayaprakash Kar. ELDA: an efficient and low-cost protocol for data authentication for IoT. *Wireless Networks*, 27(6):3969–3978, August 2021. ISSN 1572-8196. doi: 10.1007/s11276-021-02739-3. URL https://doi.org/10.1007/s11276-021-02739-3.

[58] Feng Cao and Zhenfu Cao. A secure identity-based proxy multi-signature scheme. *Information Sciences*, 179(3):292–302, January 2009. ISSN 0020-0255. doi: 10.1016/j.ins.2008.05.039. URL https://www.sciencedirect.com/science/article/pii/S0020025508003988.

[59] Nasr Abosata, Saba Al-Rubaye, and Gokhan Inalhan. Lightweight Payload Encryption-Based Authentication Scheme for Advanced Metering Infrastructure Sensor Networks. *Sensors*, 22(2):534, January 2022. ISSN 1424-8220. doi: 10.3390/s22020534. URL https://www.mdpi.com/1424-8220/22/2/534. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.

[60] Zach Shelby, Klaus Hartke, and Carsten Bormann. The Constrained Application Protocol (CoAP). Request for Comments RFC 7252, Internet Engineering Task Force, June 2014. URL https://datatracker.ietf.org/doc/rfc7252. Num Pages: 112.

[61] Utsav Banerjee, Chiraag Juvekar, Andrew Wright, Arvind, and Anantha P. Chandrakasan. An energy-efficient reconfigurable DTLS cryptographic engine for End-to-End security in iot applications. In *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*, pages 42–44, February 2018. doi: 10.1109/ISSCC.2018.8310174. ISSN: 2376-8606.

[62] LoRa Alliance. LoRaWAN® Specification v1.1, 2017. URL https://lora-alliance.org/resource_hub/lorawan-specification-v1-1/.

[63] LoRaWAN® Is Secure (but Implementation Matters). URL https://lora-alliance.org/resource_hub/lorawan-is-secure-but-implementation-matters/.

[64] Cesar Cerrudo, Esteban Martinez Fayo, and Matias Sequeira. LoRaWAN Networks Susceptible to Hacking: Common Cyber Security Problems, How to Detect and Prevent Them. Whitepaper, January 2020.

[65] Sébastien Dudek. Low Powered and High Risk: Possible Attacks on LoRaWAN Devices, January 2021. URL https://www.trendmicro.com/en_fi/research/21/a/Low-Powered-but-High-Risk-Evaluating-Possible-Attacks-on-LoRaWAN-Devices.html. Section: research.

[66] Frank Hessel, Lars Almon, and Matthias Hollick. LoRaWAN Security: An Evolvable Survey on Vulnerabilities, Attacks and their Systematic Mitigation. *ACM Transactions on Sensor Networks*, September 2022. ISSN 1550-4859. doi: 10.1145/3561973. URL https://doi.org/10.1145/3561973. Just Accepted.

[67] JungWoon Lee, DongYeop Hwang, JiHong Park, and Ki-Hyung Kim. Risk analysis and countermeasure for bit-flipping attack in LoRaWAN. In *2017 International Conference on Information Networking (ICOIN)*, pages 549–551, January 2017. doi: 10.1109/ICOIN.2017.7899554.

[68] Xueying Yang, Evgenios Karampatzakis, Christian Doerr, and Fernando Kuipers. Security Vulnerabilities in LoRaWAN. In *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 129–140, April 2018. doi: 10.1109/IoTDI.2018.00022.

[69] Poliana De Moraes and Arlindo Flavio Da Conceição. Protecting LoRaWan data against untrusted network servers. In *2021 IEEE International Conferences on Internet of Things (iThings) and IEEE Green Computing & Communications (GreenCom) and IEEE Cyber,*

Physical & Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics), pages 99–106, December 2021. doi: 10.1109/iThings-GreenCom-CPSCom-SmartData-Cybermatics53846.2021.00029.

[70] Enocean Alliance. EnOcean–The World of Energy Harvesting Wireless Technology, 2016. URL https://www.enocean.com/wp-content/uploads/redaktion/pdf/white_paper/WhitePaper_Getting_Started_With_EnOcean_v4.0.pdf.

[71] ISO Central Secretary. Information technology — Home electronic systems (HES) architecture — Part 3-10: Amplitude modulated wireless short-packet (AMWSP) protocol optimized for energy harvesting — Architecture and lower layer protocols, 2020. URL https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/08/09/80934.html.

[72] Katharina Hofer-Schmitz. A Formal Analysis of EnOcean's Teach-in and Authentication. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*, ARES 21, pages 1–8, New York, NY, USA, August 2021. Association for Computing Machinery. ISBN 978-1-4503-9051-4. doi: 10.1145/3465481.3470097. URL https://doi.org/10.1145/3465481.3470097.

[73] Yi Wu and Tao Feng. An Anonymous Authentication and Key Update Mechanism for IoT Devices Based on EnOcean Protocol. *Sensors*, 22(17):6713, January 2022. ISSN 1424-8220. doi: 10.3390/s22176713. URL https://www.mdpi.com/1424-8220/22/17/6713. Number: 17 Publisher: Multidisciplinary Digital Publishing Institute.

[74] Adrian Perrig, Ran Canetti, J D Tygar, and Dawn Song. The TESLA Broadcast Authentication Protocol. *CryptoBytes*, vol. 5 no. 2:2–13, 2002.

[75] Adrian Perrig, Robert Szewczyk, J.D. Tygar, Victor Wen, and David E. Culler. SPINS: Security Protocols for Sensor Networks. *Wireless Networks*, 8(5):521–534, September 2002. ISSN 1572-8196. doi: 10.1023/A:1016598314198. URL https://doi.org/10.1023/A:1016598314198.

[76] Ahren Studer, Fan Bai, Bhargav Bellur, and Adrian Perrig. Flexible, extensible, and efficient VANET authentication. *Journal of Communications and Networks*, 11(6):574–588, December 2009. ISSN 1976-5541. doi: 10.1109/JCN.2009.6388411. Conference Name: Journal of Communications and Networks.

[77] Donggang Liu and Peng Ning. Multilevel TESLA: Broadcast authentication for distributed sensor networks. *ACM Transactions on Embedded Computing Systems*, 3 (4):800–836, November 2004. ISSN 1539-9087. doi: 10.1145/1027794.1027800. URL https://doi.org/10.1145/1027794.1027800.

[78] Khouloud Eledlebi, Ahmed Adel Alzubaidi, Chan Yeob Yeun, Ernesto Damiani, Victor Mateu, and Yousof Al-Hammadi. Enhanced Inf-TESLA Protocol: A Continuous Connectivity and Low Overhead Authentication Protocol via IoT Devices. *IEEE Access*, 10: 54912–54921, 2022. ISSN 2169-3536. doi: 10.1109/ACCESS.2022.3177268. Conference Name: IEEE Access.

[79] Thibaut Vandervelden, Ruben De Smet, Kris Steenhaut, and An Braeken. Symmetric-Key-Based Authentication among the Nodes in a Wireless Sensor and Actuator Network. *Sensors*, 22(4):1403, January 2022. ISSN 1424-8220. doi: 10.3390/s22041403. URL https://www.mdpi.com/1424-8220/22/4/1403. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[80] Jie Cui, Lu Wei, Hong Zhong, Jing Zhang, Yan Xu, and Lu Liu. Edge Computing in VANETs-An Efficient and Privacy-Preserving Cooperative Downloading Scheme. *IEEE Journal on Selected Areas in Communications*, 38(6):1191–1204, June 2020. ISSN 1558-0008. doi: 10.1109/JSAC.2020.2986617. Conference Name: IEEE Journal on Selected Areas in Communications.

[81] Shihan Bao, Waleed Hathal, Haitham Cruickshank, Zhili Sun, Phillip Asuquo, and Ao Lei. A lightweight authentication and privacy-preserving scheme for VANETs using TESLA and Bloom Filters. *ICT Express*, 4(4):221–227, December 2018. ISSN

2405-9595. doi: 10.1016/j.icte.2017.12.001. URL https://www.sciencedirect.com/science/article/pii/S2405959517302333.

[82] Gang Han, Haibo Zeng, Yaping Li, and Wenhua Dou. SAFE: Security-Aware FlexRay Scheduling Engine. In *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–4, March 2014. doi: 10.7873/DATE.2014.021. ISSN: 1558-1101.

[83] Valeria Catalano, Ricardo Prata, Filipe Carvalho, Rui Nunes, Livio Marradi, Gianluca Franzoni, Marco Puccitelli, Roberto Campana, and Ciro Gioia. Galileo OSNMA Preliminary Implementation in the GIANO GNSS Receiver. In *Proceedings of the 33rd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2020)*, pages 3738–3750, 2020.

[84] Deok Kyu Kwon, Sung Jin Yu, Joon Young Lee, Seung Hwan Son, and Young Ho Park. WSN-SLAP: Secure and Lightweight Mutual Authentication Protocol for Wireless Sensor Networks. *Sensors*, 21(3):936, January 2021. ISSN 1424-8220. doi: 10.3390/s21030936. URL https://www.mdpi.com/1424-8220/21/3/936. Number: 3 Publisher: Multidisciplinary Digital Publishing Institute.

[85] Hassan N. Noura, Ola Salman, Raphaël Couturier, and Ali Chehab. A Single-Pass and One-Round Message Authentication Encryption for Limited IoT Devices. *IEEE Internet of Things Journal*, 9(18):17885–17900, September 2022. ISSN 2327-4662. doi: 10.1109/JIOT.2022.3161192. Conference Name: IEEE Internet of Things Journal.

[86] Pradeep Sudhakaran and Malathy C. Energy efficient distributed lightweight authentication and encryption technique for IoT security. *International Journal of Communication Systems*, 35(2):e4198, 2022. ISSN 1099-1131. doi: 10.1002/dac.4198. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/dac.4198. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/dac.4198.

[87] Doug Whiting, Russ Housley, and Niels Ferguson. Counter with CBC-MAC (CCM). Request for Comments RFC 3610, Internet Engineering Task Force, September 2003. URL https://datatracker.ietf.org/doc/rfc3610. Num Pages: 26.

[88] Gaurang Bansal and Biplab Sikdar. Beyond Traditional Message Authentication Codes: Future Solutions for Efficient Authentication of Message Streams in IoT Networks. *IEEE Internet of Things Magazine*, 5(2):102–106, June 2022. ISSN 2576-3199. doi: 10.1109/IOTM.001.2200024. Conference Name: IEEE Internet of Things Magazine.

[89] He Li, Vireshwar Kumar, Jung-Min Park, and Yaling Yang. Cumulative Message Authentication Codes for Resource-Constrained IoT Networks. *IEEE Internet of Things Journal*, 8(15):11847–11859, August 2021. ISSN 2327-4662. doi: 10.1109/JIOT.2021.3074054. Conference Name: IEEE Internet of Things Journal.

[90] Jackson Schmandt, Alan T. Sherman, and Nilanjan Banerjee. Mini-MAC: Raising the bar for vehicular security with a lightweight message authentication protocol. *Vehicular Communications*, 9:188–196, July 2017. ISSN 2214-2096. doi: 10.1016/j.vehcom.2017.07.002. URL `https://www.sciencedirect.com/science/article/pii/S2214209616301619`.

[91] Frederik Armknecht, Paul Walther, Gene Tsudik, Martin Beck, and Thorsten Strufe. ProMACs: Progressive and Resynchronizing MACs for Continuous Efficient Authentication of Message Streams. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, pages 211–223, New York, NY, USA, November 2020. Association for Computing Machinery. ISBN 978-1-4503-7089-9. doi: 10.1145/3372297.3423349. URL `https://doi.org/10.1145/3372297.3423349`.

[92] Priyanka Mall, Ruhul Amin, Ashok Kumar Das, Mark T. Leung, and Kim-Kwang Raymond Choo. PUF-Based Authentication and Key Agreement Protocols for IoT, WSNs, and Smart Grids: A Comprehensive Survey. *IEEE Internet of Things Journal*, 9(11):8205–8228, June 2022. ISSN 2327-4662. doi: 10.1109/JIOT.2022.3142084. Conference Name: IEEE Internet of Things Journal.

[93] Fadi Farha, Huansheng Ning, Karim Ali, Liming Chen, and Christopher Nugent. SRAM-PUF-Based Entities Authentication Scheme for Resource-Constrained IoT Devices. *IEEE Internet of Things Journal*, 8(7):5904–5913, April 2021. ISSN 2327-4662. doi: 10.1109/JIOT.2020.3032518. Conference Name: IEEE Internet of Things Journal.

[94] Susovan Chanda, Ashish Kumar Luhach, Waleed Alnumay, Indranil Sengupta, and Diptendu Sinha Roy. A lightweight device-level Public Key Infrastructure with DRAM based Physical Unclonable Function (PUF) for secure cyber physical systems. *Computer Communications*, 190:87–98, June 2022. ISSN 0140-3664. doi: 10.1016/j.comcom.2022.03.012. URL `https://www.sciencedirect.com/science/article/pii/S0140366422000871`.

[95] Sourav Roy, Dipnarayan Das, Anindan Mondal, Mahabub Hasan Mahalat, Bibhash Sen, and Biplab Sikdar. PLAKE: PUF based secure Lightweight Authentication and Key Exchange Protocol for IoT. *IEEE Internet of Things Journal*, pages 1–1, 2022. ISSN 2327-4662. doi: 10.1109/JIOT.2022.3202265. Conference Name: IEEE Internet of Things Journal.

[96] Mahmood Azhar Qureshi and Arslan Munir. PUF-RAKE: A PUF-Based Robust and Lightweight Authentication and Key Establishment Protocol. *IEEE Transactions on Dependable and Secure Computing*, 19(4):2457–2475, July 2022. ISSN 1941-0018. doi: 10.1109/TDSC.2021.3059454. Conference Name: IEEE Transactions on Dependable and Secure Computing.

[97] Hüsnü Yıldız, Murat Cenk, and Ertan Onur. PLGAKD: A PUF-Based Lightweight Group Authentication and Key Distribution Protocol. *IEEE Internet of Things Journal*, 8(7):5682–5696, April 2021. ISSN 2327-4662. doi: 10.1109/JIOT.2020.3032757. Conference Name: IEEE Internet of Things Journal.

[98] Sensen Li, Tikui Zhang, Bin Yu, and Kuan He. A Provably Secure and Practical PUF-Based End-to-End Mutual Authentication and Key Exchange Protocol for IoT. *IEEE Sensors Journal*, 21(4):5487–5501, February 2021. ISSN 1558-1748. doi: 10.1109/JSEN.2020.3028872. Conference Name: IEEE Sensors Journal.

[99] Mahabub Hasan Mahalat, Dipankar Karmakar, Anindan Mondal, and Bibhash Sen. PUF based Secure and Lightweight Authentication and Key-Sharing Scheme for Wireless Sensor Network. *ACM Journal on Emerging Technologies in Computing Systems*, 18(1):

1–23, January 2022. ISSN 1550-4832, 1550-4840. doi: 10.1145/3466682. URL https://dl.acm.org/doi/10.1145/3466682.

[100] Baibhab Chatterjee, Debayan Das, Shovan Maity, and Shreyas Sen. RF-PUF: Enhancing IoT Security Through Authentication of Wireless Nodes Using In-Situ Machine Learning. *IEEE Internet of Things Journal*, 6(1):388–398, February 2019. ISSN 2327-4662. doi: 10.1109/JIOT.2018.2849324. Conference Name: IEEE Internet of Things Journal.

[101] Guanxiong Shen, Junqing Zhang, Alan Marshall, and Joseph R. Cavallaro. Towards Scalable and Channel-Robust Radio Frequency Fingerprint Identification for LoRa. *IEEE Transactions on Information Forensics and Security*, 17:774–787, 2022. ISSN 1556-6021. doi: 10.1109/TIFS.2022.3152404. Conference Name: IEEE Transactions on Information Forensics and Security.

[102] Junqing Zhang, Chip-Hong Chang, Chongyan Gu, and Lajos Hanzo. Radio Frequency Fingerprints vs. Physical Unclonable Functions - Are They Twins, Competitors, or Allies? *IEEE Network*, 36(6):68–75, November 2022. ISSN 1558-156X. doi: 10.1109/MNET.107.2100372. Conference Name: IEEE Network.

[103] Arie Haenel, Yoram Haddad, Maryline Laurent, and Zonghua Zhang. Practical Cross-Layer Radio Frequency-Based Authentication Scheme for Internet of Things. *Sensors*, 21(12):4034, January 2021. ISSN 1424-8220. doi: 10.3390/s21124034. URL https://www.mdpi.com/1424-8220/21/12/4034. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[104] Rick Kuhn, Dylan Yaga, and Jeffrey Voas. Rethinking Distributed Ledger Technology. *Computer*, 52(2):68–72, February 2019. ISSN 1558-0814. doi: 10.1109/MC.2019.2898162. URL https://ieeexplore.ieee.org/document/8672407. Conference Name: Computer.

[105] Michel Rauchs, Anton Dek, and Apolline Blandin. Cambridge bitcoin electricity consumption index (cbeci). *URL: https://ccaf.io/cbeci/index (accessed 22th Jun 2022)*, 2022.

[106] Elie Kapengut and Bruce Mizrach. An Event Study of the Ethereum Transition to Proof-of-Stake, October 2022. URL http://arxiv.org/abs/2210.13655. arXiv:2210.13655 [q-fin].

[107] B S Anupama and N R Sunitha. Analysis of the Consensus Protocols used in Blockchain Networks – An overview. In *2022 IEEE International Conference on Data Science and Information System (ICDSIS)*, pages 1–6, July 2022. doi: 10.1109/ICDSIS55133.2022.9915929.

[108] Jiajing Wu, Jieli Liu, Yijing Zhao, and Zibin Zheng. Analysis of cryptocurrency transactions from a network perspective: An overview. *Journal of Network and Computer Applications*, 190:103139, September 2021. ISSN 1084-8045. doi: 10.1016/j.jnca.2021.103139. URL https://www.sciencedirect.com/science/article/pii/S1084804521001557.

[109] Shubhani Aggarwal and Neeraj Kumar. Hyperledger. In Shubhani Aggarwal, Neeraj Kumar, and Pethuru Raj, editors, *Advances in Computers*, volume 121 of *The Blockchain Technology for Secure and Smart Applications across Industry Verticals*, pages 323–343. Elsevier, January 2021. doi: 10.1016/bs.adcom.2020.08.016. URL https://www.sciencedirect.com/science/article/pii/S0065245820300711.

[110] Yujin Kwon, Jian Liu, Minjeong Kim, Dawn Song, and Yongdae Kim. Impossibility of Full Decentralization in Permissionless Blockchains. In *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, AFT '19, pages 110–123, New York, NY, USA, October 2019. Association for Computing Machinery. ISBN 978-1-4503-6732-5. doi: 10.1145/3318041.3355463. URL https://doi.org/10.1145/3318041.3355463.

[111] Bahareh Lashkari and Petr Musilek. A Comprehensive Review of Blockchain Consensus Mechanisms. *IEEE Access*, 9:43620–43652, 2021. ISSN 2169-3536. doi: 10.1109/ACCESS.2021.3065880. Conference Name: IEEE Access.

[112] Abigael Okikijesu Bada, Amalia Damianou, Constantinos Marios Angelopoulos, and Vasilios Katos. Towards a Green Blockchain: A Review of Consensus Mechanisms

and their Energy Consumption. In *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 503–511, July 2021. doi: 10.1109/DCOSS52077.2021.00083. ISSN: 2325-2944.

[113] Qiheng Zhou, Huawei Huang, Zibin Zheng, and Jing Bian. Solutions to Scalability of Blockchain: A Survey. *IEEE Access*, 8:16440–16455, 2020. ISSN 2169-3536. doi: 10.1109/ACCESS.2020.2967218. Conference Name: IEEE Access.

[114] Zachary Auhl, Naveen Chilamkurti, Rabei Alhadad, and Will Heyne. A Comparative Study of Consensus Mechanisms in Blockchain for IoT Networks. *Electronics*, 11(17): 2694, January 2022. ISSN 2079-9292. doi: 10.3390/electronics11172694. URL `https://www.mdpi.com/2079-9292/11/17/2694`. Number: 17 Publisher: Multidisciplinary Digital Publishing Institute.

[115] Mohamed Tahar Hammi, Badis Hammi, Patrick Bellot, and Ahmed Serhrouchni. Bubbles of Trust: A decentralized blockchain-based authentication system for IoT. *Computers & Security*, 78:126–142, September 2018. ISSN 0167-4048. doi: 10.1016/j.cose.2018.06.004. URL `https://www.sciencedirect.com/science/article/pii/S0167404818300890`.

[116] Bahar Farahani, Farshad Firouzi, and Markus Luecking. The convergence of IoT and distributed ledger technologies (DLT): Opportunities, challenges, and solutions. *Journal of Network and Computer Applications*, 177:102936, March 2021. ISSN 1084-8045. doi: 10.1016/j.jnca.2020.102936. URL `https://www.sciencedirect.com/science/article/pii/S1084804520303945`.

[117] Zhihua Cui, Fei XUE, Shiqiang Zhang, Xingjuan Cai, Yang Cao, Wensheng Zhang, and Jinjun Chen. A Hybrid BlockChain-Based Identity Authentication Scheme for Multi-WSN. *IEEE Transactions on Services Computing*, 13(2):241–251, March 2020. ISSN 1939-1374. doi: 10.1109/TSC.2020.2964537. Conference Name: IEEE Transactions on Services Computing.

[118] Nathan Sealey, Adnan Aijaz, and Ben Holden. IOTA Tangle 2.0: Toward a Scalable,

Decentralized, Smart, and Autonomous IoT Ecosystem, September 2022. URL http://arxiv.org/abs/2209.04959. arXiv:2209.04959 [cs].

[119] IOTA Foundation. Overview | Introduction | Streams | IOTA Documentation, 2020. URL https://legacy.docs.iota.works/docs/iota-streams/1.1/overview.

[120] IOTA Foundation. Introducing Masked Authenticated Messaging, November 2017. URL http://blog.iota.org/introducing-masked-authenticated-messaging-e55c1822d50e.

[121] Mohammed Elhajj, Hassan Jradi, Maroun Chamoun, and Ahmad Fadlallah. LASII: Lightweight Authentication Scheme using IOTA in IoT Platforms. In *2022 20th Mediterranean Communication and Computer Networking Conference (MedComNet)*, pages 74–83, June 2022. doi: 10.1109/MedComNet55087.2022.9810397.

[122] Iuon-Chang Lin, Chin-Chen Chang, and Yu-Sung Chang. Data Security and Preservation Mechanisms for Industrial Control Network Using IOTA. *Symmetry*, 14 (2):237, February 2022. ISSN 2073-8994. doi: 10.3390/sym14020237. URL https://www.mdpi.com/2073-8994/14/2/237. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.

[123] IOTA Foundation. IOTA Streams, 2020. URL https://legacy.docs.iota.works/docs/iota-streams/1.1/overview.

[124] Alberto Carelli, Andrea Palmieri, Antonio Vilei, Fabien Castanier, and Andrea Vesco. Enabling Secure Data Exchange through the IOTA Tangle for IoT Constrained Devices. *Sensors*, 22(4), 2022. ISSN 1424-8220. doi: 10.3390/s22041384. URL https://www.mdpi.com/1424-8220/22/4/1384.

[125] Giovanni Verhaeghe, Michael Marriott, and Andrew Till. A new formula for IoT security is risk equals probability multiplied by loss, May 2017. URL https://www.blmlaw.com/images/uploaded/File/IoT-Now-Magazine-April-May-2017.pdf. ISSN: 2397-2793 Issue: 2 Volume: 7 tex.howpublished: IoT Now Magazine.

[126] R. Sandhu. Good-enough security. *IEEE Internet Computing*, 7(1):66–68, January 2003. ISSN 1089-7801. doi: 10.1109/MIC.2003.1167341.

[127] Amine Erroutbi, Adnane El Hanjri, and Abderrahim Sekkaki. Secure and lightweight HMAC mutual authentication protocol for communication between IoT devices and fog nodes. In *2019 IEEE international smart cities conference (ISC2)*, pages 251–257, 2019. doi: 10.1109/ISC246665.2019.9071788.

[128] Hamza Khemissa and Djamel Tandjaoui. A lightweight authentication scheme for E-health applications in the context of internet of things. In *2015 9th international conference on next generation mobile applications, services and technologies*, pages 90–95, 2015. doi: 10.1109/NGMAST.2015.31.

[129] Jean-Philippe Aumasson and Daniel J. Bernstein. SipHash: A fast short-input PRF. In *Progress in cryptology - INDOCRYPT 2012, 13th international conference on cryptology in india, kolkata, india, december 9-12, 2012. Proceedings*, pages 489–508, 2012. doi: 10.1007/978-3-642-34931-7_28.

[130] Nicky Mouha, Bart Mennink, Anthony Van Herrewege, Dai Watanabe, Bart Preneel, and Ingrid Verbauwhede. Chaskey: An efficient MAC algorithm for 32-bit microcontrollers. In Antoine Joux and Amr Youssef, editors, *Selected areas in cryptography – SAC 2014*, pages 306–323, Cham, 2014. Springer International Publishing. ISBN 978-3-319-13051-4.

[131] Ying Li, Xiang Chen, Yun Lin, Gautam Srivastava, and Shuai Liu. Wireless transmitter identification based on device imperfections. *IEEE access : practical innovations, open solutions*, 8:59305–59314, 2020. doi: 10.1109/ACCESS.2020.2981428.

[132] Qiao Tian, Yun Lin, Xinghao Guo, Jin Wang, Osama AlFarraj, and Amr Tolba. An identity authentication method of a MIoT device based on radio frequency (RF) fingerprint technology. *Sensors*, 20(4), 2020. ISSN 1424-8220. doi: 10.3390/s20041213. URL `https://www.mdpi.com/1424-8220/20/4/1213`. tex.article-number: 1213.

[133] Guyue Li, Jiabao Yu, Yuexiu Xing, and Aiqun Hu. Location-invariant physical layer

identification approach for WiFi devices. *IEEE access : practical innovations, open solutions*, 7:106974–106986, 2019. doi: 10.1109/ACCESS.2019.2933242.

[134] Q. Xu, R. Zheng, W. Saad, and Z. Han. Device fingerprinting in wireless networks: Challenges and opportunities. *IEEE Communications Surveys Tutorials*, 18(1):94–104, 2016. ISSN 1553-877X. doi: 10.1109/COMST.2015.2476338.

[135] Mohamed Amine Ferrag, Leandros A Maglaras, Helge Janicke, Jianmin Jiang, and Lei Shu. Authentication protocols for internet of things: a comprehensive survey. *Security and Communication Networks*, 2017, 2017. URL `https://www.hindawi.com/journals/scn/2017/6562953/`. Publisher: Hindawi.

[136] C. Zhao, L. Huang, Y. Zhao, and X. Du. Secure machine-type communications toward LTE heterogeneous networks. *IEEE Wireless Communications*, 24(1):82–87, February 2017. ISSN 1536-1284. doi: 10.1109/MWC.2017.1600141WC.

[137] J. Toonstra and W. Kinsner. Transient analysis and genetic algorithms for classification. In *IEEE WESCANEX 95. Communications, power, and computing. Conference proceedings*, volume 2, pages 432–437 vol.2, May 1995. doi: 10.1109/WESCAN.1995.494069.

[138] Howard C. Choe, Clark E. Poole, Andrea M. Yu, and Harold H. Szu. Novel identification of intercepted signals from unknown radio transmitters. *Proc.SPIE*, 2491, 1995. doi: 10.1117/12.205415.

[139] J. Toonstra and W. Kinsner. A radio transmitter fingerprinting system ODO-1. In *Proceedings of 1996 canadian conference on electrical and computer engineering*, volume 1, pages 60–63 vol.1, May 1996. doi: 10.1109/CCECE.1996.548038. ISSN: 0840-7789.

[140] G. Baldini, R. Giuliani, G. Steri, and R. Neisse. Physical layer authentication of Internet of Things wireless devices through permutation and dispersion entropy. In *2017 global internet of things summit (GIoTS)*, pages 1–6, June 2017. doi: 10.1109/GIOTS.2017.8016272.

[141] J. Yu, A. Hu, G. Li, and L. Peng. A robust RF fingerprinting approach using multi-sampling convolutional neural network. *IEEE Internet of Things Journal*, 2019. ISSN 2327-4662.

[142] M. Köse, S. Taşcioğlu, and Z. Telatar. RF fingerprinting of IoT devices based on transient energy spectrum. *IEEE access : practical innovations, open solutions*, 7:18715–18726, 2019. ISSN 2169-3536.

[143] Christina Pöpper, Mario Strasser, and Srdjan Čapkun. Jamming-resistant broadcast communication without shared keys. In *Proceedings of the 18th conference on USENIX security symposium*, SSYM'09, pages 231–248, USA, 2009. USENIX Association. Number of pages: 18 Place: Montreal, Canada.

[144] William Allen Simpson. PPP challenge handshake authentication protocol (CHAP). *RFC 1994*, 1996. doi: 10.17487/RFC1994. tex.bibsource: dblp computer science bibliography, https://dblp.org.

[145] Bernard Aboba, Larry J. Blunk, John R. Vollbrecht, James Carlson, and Henrik Levkowetz. Extensible authentication protocol (EAP). *RFC 3748*, 2004. doi: 10.17487/RFC3748.

[146] David M'Raihi, Mihir Bellare, Frank Hoornaert, David Naccache, and Ohad Ranen. Hotp: An hmac-based one-time password algorithm. *The Internet Society, Network Working Group. RFC4226*, 2005.

[147] Hamed HaddadPajouh and Ali Dehghantanha and Reza M. Parizi and Mohammed Aledhari and Hadis Karimipour. A survey on internet of things security: Requirements, challenges, and solutions. *Internet of Things*, 2019. ISSN 2542-6605. doi: https://doi.org/10.1016/j.iot.2019.100129.

[148] Samy Kamkar. Drive it like you hacked it: New attacks and tools to wirelessly steal cars. *Presentation at DEFCON*, 23, 2015.

[149] Del-Valle-Soto, Carolina and Mex-Perera, Carlos and Aldaya, Ivan and Lezama, Fernando and Nolazco-Flores, Juan Arturo and Monroy, Raul. New detection paradigms to improve wireless sensor network performance under jamming attacks. *Sensors*, 19(11), 2019. ISSN 1424-8220. doi: 10.3390/s19112489. tex.article-number: 2489.

[150] B. Chatfield, R. J. Haddad, and L. Chen. Low-computational complexity intrusion detection system for jamming attacks in smart grids. In *2018 international conference on computing, networking and communications (ICNC)*, pages 367–371, 2018. doi: 10.1109/ICCNC.2018.8390345.

[151] O. Puñal, I. Aktaş, C. Schnelke, G. Abidin, K. Wehrle, and J. Gross. Machine learning-based jamming detection for IEEE 802.11: Design and experimental evaluation. In *Proceeding of IEEE international symposium on a world of wireless, mobile and multimedia networks 2014*, pages 1–10, 2014. doi: 10.1109/WoWMoM.2014.6918964.

[152] Guoquan Li, Enxu Geng, Zhouyang Ye, Yongjun Xu, Jinzhao Lin, and Yu Pang. Indoor positioning algorithm based on the improved RSSI distance model. *Sensors*, 18(9), 2018. ISSN 1424-8220. doi: 10.3390/s18092820. tex.article-number: 2820.

[153] Vincent K Nguyen. Authentication of smartphone user using RSSI geolocation. Technical report, Naval Postgraduate School Monterey CA, 2014.

[154] F. Ma and J. Li. A non-sense authentication scheme of WLAN based on RSSI location fingerprint. In *2017 4th international conference on information science and control engineering (ICISCE)*, pages 69–72, 2017. doi: 10.1109/ICISCE.2017.24.

[155] Muhammad Naveed Aman, Mohamed Haroon Basheer, and Biplab Sikdar. Two-factor authentication for IoT with location information. *IEEE Internet of Things Journal*, 6 (2):3335–3351, 2019. doi: 10.1109/JIOT.2018.2882610.

[156] Steven So, Jonathan Petit, and David Starobinski. Physical layer plausibility checks for misbehavior detection in V2X networks. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, WiSec '19, pages 84–93, New York, NY,

USA, May 2019. Association for Computing Machinery. ISBN 978-1-4503-6726-4. doi: 10.1145/3317549.3323406. URL https://doi.org/10.1145/3317549.3323406.

[157] International Organization for Standardization. Information technology — Lightweight cryptography — Part 6: Message authentication codes (MACs). Standard, International Organization for Standardization, 2019. Volume: 2019 tex.key: ISO/IEC 29192-6:2019.

[158] G. Saldamli, L. Ertaul, and A. Shankaralingappa. Analysis of lightweight message authentication codes for IoT environments. In *2019 fourth international conference on fog and mobile edge computing (FMEC)*, pages 235–240, 2019.

[159] Texas Instruments. MSP-EXP430FR5994 LaunchPad development kit. *MSP-EXP430FR5969 LaunchPad™ Development Kit User's Guide (SLAU535)*, 2016.

[160] Texas Instruments. CC110L value line transceiver, 2016. URL https://www.ti.com/lit/ds/symlink/cc110l.pdf.

[161] Su Hu, Pei Wang, Yaping Peng, Di Lin, Yuan Gao, Jiang Cao, and Bin Yu. Machine learning for RF fingerprinting extraction and identification of soft-defined radio devices. In Qilian Liang, Wei Wang, Jiasong Mu, Xin Liu, Zhenyu Na, and Bingcai Chen, editors, *Artificial intelligence in china*, pages 189–204, Singapore, 2020. Springer Singapore. ISBN 978-981-15-0187-6.

[162] Brittany Finch and William Goh. MSP430 advanced power optimizations: ULP advisor software and energy trace technology. *Applications Report Texas Instruments*, 2014. Publisher: Texas Instruments.

[163] Benoít Fournier, Valérie Tong, and Gilles Guette. Accurate Measurement of the Energy Consumption of Security Functions:. In *Proceedings of the 18th International Conference on Security and Cryptography*, pages 487–494, Online Streaming, — Select a Country —, 2021. SCITEPRESS - Science and Technology Publications. ISBN 978-989-758-524-1. doi: 10.5220/0010544604870494. URL https://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0010544604870494.

[164] Arvinder Kaur and Ruchikaa Nayyar. A Comparative Study of Static Code Analysis tools for Vulnerability Detection in C/C++ and JAVA Source Code. *Procedia Computer Science*, 171:2023–2029, January 2020. ISSN 1877-0509. doi: 10.1016/j.procs.2020.04.217. URL https://www.sciencedirect.com/science/article/pii/S1877050920312023.

[165] Katerina Goseva-Popstojanova and Andrei Perhinschi. On the capability of static code analysis to detect security vulnerabilities. *Information and Software Technology*, 68:18–33, December 2015. ISSN 0950-5849. doi: 10.1016/j.infsof.2015.08.002. URL https://www.sciencedirect.com/science/article/pii/S0950584915001366.

[166] M. Pistoia, S. Chandra, S. J. Fink, and E. Yahav. A survey of static analysis methods for identifying security vulnerabilities in software systems. *IBM Systems Journal*, 46(2): 265–288, 2007. ISSN 0018-8670. doi: 10.1147/sj.462.0265. URL https://ieeexplore.ieee.org/abstract/document/5386616. Conference Name: IBM Systems Journal.

[167] Xu Yaozong, Shao Xuebin, Zhou Shuhua, Zhao Qiujun, and Ju Weinan. Static Analysis Method of C Code Based on Model Checking and Defect Pattern Matching. In *2023 IEEE 5th International Conference on Power, Intelligent Computing and Systems (ICPICS)*, pages 567–573, July 2023. doi: 10.1109/ICPICS58376.2023.10235566. URL https://ieeexplore.ieee.org/abstract/document/10235566. ISSN: 2834-8567.

[168] Stephan Lipp, Sebastian Banescu, and Alexander Pretschner. An empirical study on the effectiveness of static C code analyzers for vulnerability detection. In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2022, pages 544–555, New York, NY, USA, July 2022. Association for Computing Machinery. ISBN 978-1-4503-9379-9. doi: 10.1145/3533767.3534380. URL https://doi.org/10.1145/3533767.3534380.

[169] Weina Niu, Xiaosong Zhang, Xiaojiang Du, Lingyuan Zhao, Rong Cao, and Mohsen Guizani. A deep learning based static taint analysis approach for IoT software vulnerability location. *Measurement*, 152:107139, February 2020. ISSN 0263-2241. doi: 10.1016/j.

measurement.2019.107139. URL https://www.sciencedirect.com/science/article/pii/S026322411931005X.

[170] Abdalla Wasef Marashdih, Zarul Fitri Zaaba, and Khaled Suwais. An Enhanced Static Taint Analysis Approach to Detect Input Validation Vulnerability. *Journal of King Saud University - Computer and Information Sciences*, 35(2):682–701, February 2023. ISSN 1319-1578. doi: 10.1016/j.jksuci.2023.01.009. URL https://www.sciencedirect.com/science/article/pii/S1319157823000095.

[171] Solmaz Salimi and Mehdi Kharrazi. VulSlicer: Vulnerability detection through code slicing. *Journal of Systems and Software*, 193:111450, November 2022. ISSN 0164-1212. doi: 10.1016/j.jss.2022.111450. URL https://www.sciencedirect.com/science/article/pii/S0164121222001443.

[172] Saikat Chakraborty, Rahul Krishna, Yangruibo Ding, and Baishakhi Ray. Deep Learning Based Vulnerability Detection: Are We There Yet? *IEEE Transactions on Software Engineering*, 48(9):3280–3296, September 2022. ISSN 1939-3520. doi: 10.1109/TSE.2021.3087402. Conference Name: IEEE Transactions on Software Engineering.

[173] Hongliang Liang, Xiaoxiao Pei, Xiaodong Jia, Wuwei Shen, and Jian Zhang. Fuzzing: State of the Art. *IEEE Transactions on Reliability*, 67(3):1199–1218, September 2018. ISSN 1558-1721. doi: 10.1109/TR.2018.2834476. URL https://ieeexplore.ieee.org/document/8371326. Conference Name: IEEE Transactions on Reliability.

[174] Xiaogang Zhu, Sheng Wen, Seyit Camtepe, and Yang Xiang. Fuzzing: A Survey for Roadmap. *ACM Computing Surveys*, 54(11s):230:1–230:36, September 2022. ISSN 0360-0300. doi: 10.1145/3512345. URL https://doi.org/10.1145/3512345.

[175] Sanoop Mallissery and Yu-Sung Wu. Demystify the Fuzzing Methods: A Comprehensive Survey. *ACM Computing Surveys*, 56(3):71:1–71:38, October 2023. ISSN 0360-0300. doi: 10.1145/3623375. URL https://doi.org/10.1145/3623375.

[176] Michał Zalewski. AFL (american fuzzy lop), 2014. URL http://lcamtuf.coredump.cx/afl/.

[177] Chen Chen, Baojiang Cui, Jinxin Ma, Runpu Wu, Jianchao Guo, and Wenqian Liu. A systematic review of fuzzing techniques. *Computers & Security*, 75:118–137, June 2018. ISSN 0167-4048. doi: 10.1016/j.cose.2018.02.002. URL https://www.sciencedirect.com/science/article/pii/S0167404818300658.

[178] Roberto Baldoni, Emilio Coppa, Daniele Cono D'elia, Camil Demetrescu, and Irene Finocchi. A Survey of Symbolic Execution Techniques. *ACM Computing Surveys*, 51(3): 50:1–50:39, May 2018. ISSN 0360-0300. doi: 10.1145/3182657. URL https://dl.acm.org/doi/10.1145/3182657.

[179] Zian Liu, Chao Chen, Ahmed Ejaz, Dongxi Liu, and Jun Zhang. Automated Binary Analysis: A Survey. pages 392–411. January 2023. ISBN 978-3-031-22676-2. doi: 10.1007/978-3-031-22677-9_21.

[180] Guanjun Lin, Sheng Wen, Qing-Long Han, Jun Zhang, and Yang Xiang. Software Vulnerability Detection Using Deep Neural Networks: A Survey. *Proceedings of the IEEE*, 108(10):1825–1848, October 2020. ISSN 1558-2256. doi: 10.1109/JPROC.2020.2993293.

[181] Boris Chernis and Rakesh Verma. Machine Learning Methods for Software Vulnerability Detection. In *Proceedings of the Fourth ACM International Workshop on Security and Privacy Analytics*, IWSPA '18, pages 31–39, New York, NY, USA, March 2018. Association for Computing Machinery. ISBN 978-1-4503-5634-3. doi: 10.1145/3180445.3180453. URL https://dl.acm.org/doi/10.1145/3180445.3180453.

[182] Yulei Pang, Xiaozhen Xue, and Akbar Siami Namin. Predicting Vulnerable Software Components through N-Gram Analysis and Statistical Feature Selection. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 543–548, December 2015. doi: 10.1109/ICMLA.2015.99. URL https://ieeexplore.ieee.org/abstract/document/7424372.

[183] Jacob A. Harer, Louis Y. Kim, Rebecca L. Russell, Onur Ozdemir, Leonard R. Kosta, Akshay Rangamani, Lei H. Hamilton, Gabriel I. Centeno, Jonathan R. Key, Paul M. Ellingwood, Erik Antelman, Alan Mackay, Marc W. McConley, Jeffrey M. Opper, Peter

Chin, and Tomo Lazovich. Automated software vulnerability detection with machine learning, August 2018. URL http://arxiv.org/abs/1803.04497. arXiv:1803.04497 [cs, stat].

[184] Zhen Li, Deqing Zou, Shouhuai Xu, Xinyu Ou, Hai Jin, Sujuan Wang, Zhijun Deng, and Yuyi Zhong. VulDeePecker: A Deep Learning-Based System for Vulnerability Detection. In *Proceedings 2018 Network and Distributed System Security Symposium*, 2018. doi: 10.14722/ndss.2018.23158. URL http://arxiv.org/abs/1801.01681. arXiv:1801.01681 [cs].

[185] Zhen Li, Deqing Zou, Shouhuai Xu, Zhaoxuan Chen, Yawei Zhu, and Hai Jin. VulDeeLocator: A Deep Learning-Based Fine-Grained Vulnerability Detector. *IEEE Transactions on Dependable and Secure Computing*, 19(4):2821–2837, July 2022. ISSN 1941-0018. doi: 10.1109/TDSC.2021.3076142. Conference Name: IEEE Transactions on Dependable and Secure Computing.

[186] Yueming Wu, Deqing Zou, Shihan Dou, Wei Yang, Duo Xu, and Hai Jin. VulCNN: an image-inspired scalable vulnerability detection system. In *Proceedings of the 44th International Conference on Software Engineering*, ICSE '22, pages 2365–2376, New York, NY, USA, July 2022. Association for Computing Machinery. ISBN 978-1-4503-9221-1. doi: 10.1145/3510003.3510229. URL https://doi.org/10.1145/3510003.3510229.

[187] Litao Li, Steven H. H. Ding, Yuan Tian, Benjamin C. M. Fung, Philippe Charland, Weihan Ou, Leo Song, and Congwei Chen. VulANalyzeR: Explainable Binary Vulnerability Detection with Multi-task Learning and Attentional Graph Convolution. *ACM Transactions on Privacy and Security*, 26(3):28:1–28:25, April 2023. ISSN 2471-2566. doi: 10.1145/3585386. URL https://doi.org/10.1145/3585386.

[188] OpenAI. GPT-4 Technical Report, March 2023. arXiv: 2303.08774 [cs] Issue: arXiv:2303.08774.

[189] Marwan Omar. Detecting software vulnerabilities using Language Models, February 2023. URL http://arxiv.org/abs/2302.11773. arXiv:2302.11773 [cs].

[190] Jin Wang, Zishan Huang, Hengli Liu, Nianyi Yang, and Yinhao Xiao. DefectHunter: A Novel LLM-Driven Boosted-Conformer-based Code Vulnerability Detection Mechanism, September 2023. arXiv: 2309.15324 [cs] Issue: arXiv:2309.15324.

[191] Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Merouane Debbah, Etienne Goffinet, Daniel Heslow, Julien Launay, Quentin Malartic, and others. Falcon-40B: an open large language model with state-of-the-art performance. Technical report, Technical report, Technology Innovation Institute, 2023.

[192] Norbert Tihanyi, Tamas Bisztray, Ridhi Jain, Mohamed Amine Ferrag, Lucas C. Cordeiro, and Vasileios Mavroeidis. The FormAI Dataset: Generative AI in Software Security Through the Lens of Formal Verification, September 2023. arXiv: 2307.02192 [cs] Issue: arXiv:2307.02192.

[193] Zimin Chen, Steve Kommrusch, and Martin Monperrus. Neural transfer learning for repairing security vulnerabilities in C code. *IEEE Transactions on Software Engineering*, 49(1):147–165, January 2023. ISSN 1939-3520. doi: 10.1109/TSE.2022.3147265.

[194] Francesco Lomio, Emanuele Iannone, Andrea De Lucia, Fabio Palomba, and Valentina Lenarduzzi. Just-in-time software vulnerability detection: Are we there yet? *Journal of Systems and Software*, 188:111283, June 2022. ISSN 0164-1212. doi: 10.1016/j.jss.2022.111283.

[195] Zhe Yu, Christopher Theisen, Laurie Williams, and Tim Menzies. Improving vulnerability inspection efficiency using active learning. *IEEE Transactions on Software Engineering*, 47(11):2401–2420, November 2021. ISSN 1939-3520. doi: 10.1109/TSE.2019.2949275.

[196] Yangruibo Ding, Yanjun Fu, Omniyyah Ibrahim, Chawin Sitawarin, Xinyun Chen, Basel Alomair, David Wagner, Baishakhi Ray, and Yizheng Chen. Vulnerability Detection with Code Language Models: How Far Are We?, March 2024. URL `http://arxiv.org/abs/2403.18624`. arXiv:2403.18624 [cs].

[197] Alberto Bacchelli and Christian Bird. Expectations, outcomes, and challenges of modern code review. In *2013 35th international conference on software engineering (ICSE)*, pages 712–721. IEEE, 2013.

[198] Wachiraphan Charoenwet. Complementing Secure Code Review with Automated Program Analysis. In *2023 IEEE/ACM 45th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, pages 189–191, May 2023. doi: 10.1109/ICSE-Companion58688.2023.00052. ISSN: 2574-1934.

[199] Chanathip Pornprasit and Chakkrit Kla Tantithamthavorn. DeepLineDP: Towards a Deep Learning Approach for Line-Level Defect Prediction. *IEEE Transactions on Software Engineering*, 49(1):84–98, January 2023. ISSN 1939-3520. doi: 10.1109/TSE.2022.3144348.

[200] David Hin, Andrey Kan, Huaming Chen, and M. Ali Babar. LineVD: Statement-level vulnerability detection using graph neural networks. In *Proceedings of the 19th International Conference on Mining Software Repositories*, MSR '22, pages 596–607, New York, NY, USA, October 2022. Association for Computing Machinery. ISBN 978-1-4503-9303-4. doi: 10.1145/3524842.3527949.

[201] Emanuele Iannone, Roberta Guadagni, Filomena Ferrucci, Andrea De Lucia, and Fabio Palomba. The Secret Life of Software Vulnerabilities: A Large-Scale Empirical Study. *IEEE Transactions on Software Engineering*, 49(1):44–63, January 2023. ISSN 1939-3520. doi: 10.1109/TSE.2022.3140868.

[202] Larissa Braz, Christian Aeberhard, Gül Çalikli, and Alberto Bacchelli. Less is more: Supporting developers in vulnerability detection during code review. In *Proceedings of the 44th International Conference on Software Engineering*, ICSE '22, pages 1317–1329, New York, NY, USA, July 2022. Association for Computing Machinery. ISBN 978-1-4503-9221-1. doi: 10.1145/3510003.3511560.

[203] Larissa Braz and Alberto Bacchelli. Software security during modern code review: The developer's perspective. In *Proceedings of the 30th ACM Joint European Software*

Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2022, pages 810–821, New York, NY, USA, November 2022. Association for Computing Machinery. ISBN 978-1-4503-9413-0. doi: 10.1145/3540250.3549135.

[204] Paul E Black and Paul E Black. *Juliet 1.3 test suite: Changes from 1.2*. US Department of Commerce, National Institute of Standards and Technology …, 2018.

[205] Richard M. Stallman. GNU Compiler Collection (GCC) Internals, 1988. URL `https://gcc.gnu.org/onlinedocs/gccint/`.

[206] Aditya Grover and Jure Leskovec. Node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864, San Francisco California USA, August 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939754.

[207] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5):602–610, July 2005. ISSN 0893-6080. doi: 10.1016/j.neunet.2005.06.042.

[208] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. The Performance of LSTM and BiLSTM in Forecasting Time Series. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 3285–3292, Los Angeles, CA, USA, December 2019. IEEE. ISBN 978-1-72810-858-2. doi: 10.1109/BigData47090.2019.9005997.

[209] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF Models for Sequence Tagging, August 2015. arXiv: 1508.01991 [cs] Number: arXiv:1508.01991.

[210] Haipeng Yao, Chong Liu, Peiying Zhang, Sheng Wu, Chunxiao Jiang, and Shui Yu. Identification of Encrypted Traffic Through Attention Mechanism Based Long Short Term Memory. *IEEE Transactions on Big Data*, 8(1):241–252, February 2022. ISSN 2332-7790. doi: 10.1109/TBDATA.2019.2940675.

[211] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghem-

awat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`.

[212] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, and others. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[213] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL `https://aclanthology.org/2020.emnlp-demos.6`.

[214] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[215] Vasan Subramanian. *Pro mern stack: Full stack web app development with mongo, express, react and node*. Springer, 2019.

[216] Zhen Li, Deqing Zou, Shouhuai Xu, Hai Jin, Yawei Zhu, and Zhaoxuan Chen. SySeVR: A Framework for Using Deep Learning to Detect Software Vulnerabilities. *IEEE Transactions on Dependable and Secure Computing*, 19(4):2244–2258, July 2022. ISSN

1941-0018. doi: 10.1109/TDSC.2021.3051525. Conference Name: IEEE Transactions on Dependable and Secure Computing.

[217] Rebecca L. Russell, Louis Kim, Lei H. Hamilton, Tomo Lazovich, Jacob A. Harer, Onur Ozdemir, Paul M. Ellingwood, and Marc W. McConley. Automated Vulnerability Detection in Source Code Using Deep Representation Learning, November 2018. arXiv: 1807.04320 [cs, stat] Number: arXiv:1807.04320.

[218] Mathieu Desmeules, Marc-André Labrie, and Kim Bouchard-Foster. iOS Security Overview. Defence Research Reports, DEFENCE RESEARCH AND DEVELOPMENT CANADA, VALCARTIER RESEARCH CENTRE, QUEBEC QC (CAN);LTI, QUEBEC QUE (CAN), 2013. Issue: DRDC-VALCARTIER-CR-2013-378 — Contract Report.

[219] Dimitrios Damopoulos, Georgios Kambourakis, and Stefanos Gritzalis. iSAM: An iPhone stealth airborne malware. In Jan Camenisch, Simone Fischer-Hübner, Yuko Murayama, Armand Portmann, and Carlos Rieder, editors, *26th international information security conference (SEC)*, volume AICT-354 of *Future challenges in security and privacy for academia and industry*, pages 17–28, Lucerne, Switzerland, June 2011. Springer. doi: 10.1007/978-3-642-21424-0_2. tex.hal_id: hal-01567607 tex.hal_version: v1.

[220] Al Bessey, Ken Block, Ben Chelf, Andy Chou, Bryan Fulton, Seth Hallem, Charles Henri-Gros, Asya Kamsky, Scott McPeak, and Dawson Engler. A few billion lines of code later: Using static analysis to find bugs in the real world. *Communications of the ACM*, 53(2): 66–75, February 2010. ISSN 0001-0782. doi: 10.1145/1646353.1646374.

[221] Coverity Scan: Libtiff- Static Analysis. URL `https://scan.coverity.com/projects/tiff`.

[222] Xiao Cheng, Haoyu Wang, Jiayi Hua, Guoai Xu, and Yulei Sui. DeepWukong: Statically Detecting Software Vulnerabilities Using Deep Graph Neural Network. *ACM Transactions on Software Engineering and Methodology*, 30(3):38:1–38:33, April 2021. ISSN 1049-331X. doi: 10.1145/3436877. URL `https://doi.org/10.1145/3436877`.

[223] CVE-2023-40745., 2023. URL https://access.redhat.com/security/cve/cve-2023-40745.

[224] CVE-2023-41175., 2023. URL https://access.redhat.com/security/cve/cve-2023-41175.

[225] Secure Code Warrior: Training Module Overview, May 2022. URL https://help.securecodewarrior.com/hc/en-us/articles/360035983992-Training-Module-Overview.

[226] Developer security training from Snyk, 2023. URL https://learn.snyk.io/.

[227] Yaqin Zhou, Shangqing Liu, Jingkai Siow, Xiaoning Du, and Yang Liu. Devign: effective vulnerability identification by learning comprehensive program semantics via graph neural networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, number 915, pages 10197–10207. Curran Associates Inc., Red Hook, NY, USA, December 2019.

Titre : Solutions de sécurité hybrides pour des appareils IoT

Mots clés : Authentification, RF Fingerprinting, Sécurité, Détection de vulnérabilités

Résumé : Le sujet de cette recherche porte sur l'authentification et la sécurité légères, adaptées aux appareils IoT à faibles ressources et applicables à la bande ISM sub-GHz. Notre premier objectif est l'authentification des appareils IoT. Ces appareils ont souvent des limites telles qu'une puissance et un budget limités, ils ont donc besoin de méthodes d'authentification à la fois efficaces et solides. Nous nous efforçons de trouver le juste équilibre en combinant des méthodes d'authentification bien établies avec des technologies plus récentes d'empreintes digitales par radiofréquence. Cette combinaison aboutit à une approche d'authentification hybride efficace et sécurisée, conçue spécifiquement pour les appareils IoT aux ressources limitées. En complément de cette approche, notre thèse présente Shmulik, un système pionnier basé sur l'apprentissage en profondeur, conçu pour découvrir les vulnérabilités logicielles. Shmulik s'efforce de compléter les analyseurs statiques traditionnels et d'identifier de manière proactive les risques de sécurité qui autrement auraient pu rester cachés dans des bases de code étendues.

Title : Hybrid security solutions for IoT devices

Keywords : RF Fingerprinting, Security, Authentication, Vulnerability detection

Abstract : The subject of this research is about lightweight authentication and security, adapted to low resource IoT devices, and applicable for sub-GHz ISM band. Our first focus is on the authentication of IoT devices. These devices often have limitations like limited power and budget, so they need authentication methods that are both efficient and strong. We're working on finding the right balance by combining well-established authentication methods with newer Radio Frequency Fingerprinting technologies. This combination results in a hybrid authentication approach that's efficient and secure, designed specifically for resource-constrained IoT devices. Complementary to this narrative, our thesis introduces Shmulik, a pioneering deep learning-based system crafted to unearth software vulnerabilities. Shmulik endeavors to complement traditional static analyzers and proactively identify security risks that might have otherwise remained lurking within extensive codebases.

Institut Polytechnique de Paris
91120 Palaiseau, France