IMT School for Advanced Studies, Lucca
Lucca, Italy

Optimized Monitoring and Detection of Internet of Things resources-constraints Cyber Attacks

PhD Program in Institutions, Markets and Technologies

Curriculum in Computer Science and Systems Engineering

XXXIV Cycle

By

Zainab Ali Obaid Al-Waisi

2023

The dissertation of Zainab Ali Obaid Al-Waisi is approved.

PhD Program Coordinator: Prof. Alberto Bemporad, IMT School for Advanced Studies Lucca

Advisor: Prof. Rocco De Nicola, IMT School for Advanced Studies Lucca

Co-Advisor: Dr. Simone Soderi, IMT School for Advanced Studies Lucca

The dissertation of Zainab Ali Obaid Al-Waisi has been reviewed by:

Prof. Romano Fantacci, University of Florence, Italy

Prof. Alberto Lluch Lafuente, Technical University of Denmark, Denmark

IMT School for Advanced Studies Lucca
2023

To my father **Ali Alwaisi**, my mother **Elham Alkhafaji**, my husband **Mohammed Mohammed**, my brothers **Hussein Alwaisi and Abass Alwaisi**, my sister **Israa Alwaisi**, and to my life, my world, my son **Ali**.

“The way of success is the way of continuous pursuit of knowledge.”

- Napoleon Hill -

Contents

List of Figures

List of Tables

List of Abbreviations

2FA	two-Factor-Authentication
3GPP	3rd Generation Partnership Project
6LoWPAN	Low-power Wireless Personal Area Networks
AI	Artificial Intelligence
AMQP	Advanced Message Queuing Protocol
ANN	Artificial Neural Network
AP	Access Point
AR	Attack Rate
BLE	Bluetooth Low Energy
BSSID	Basic Service Set Identifier
CoAP	Constrained Application Protocol
CPS	Cyber-Physical System
CPU	Central Processing Unit
CSP	Content Security Policy
CSS	Chirp Spread Spectrum
DB	Database
DDoS	Distributed Denial of Services
DDS	Data Distribution Service
DL	Deep Learning
DLLs	Dynamic-Link Libraries
DNS	Domain Name System
DODAG	Destination Oriented Directed Acyclic Graph
DoS	Denial of Services

EC-DDoS	Distributed Denial of Services
ECC	Elliptic Curve Cryptography
EMC	ElectroMagnetic Compatibility
ETSI	European Telecommunication Standards Institute
F-APs	Fake Access Points
GPS	Global Positioning System
HTTP	HyperText Transfer Protocol
ICT	Information and Communication Technologies
ICU	Intensive Care Unit
IDC	International Data Corporation
IDS	Intrusion Detection System
IDT	Interrupt Descriptor Table
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IIoT	Industrial Internet of Things
IoE	Internet of Everything
IoT	Internet of Things
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ITU	International Telecommunication Union
LLNs	Low-power, and Lossy Networks
LoWPAN	Low-power Wireless Personal Area Networks
LPWAN	Low-Power-Wide-Area-Network
LR-WPAN	Low-Rate Wireless Personal Area Network
LTE	Long-Term Evolution
LTE-A	Long-Term Evolution Advanced
M2M	Machine-to-Machine
MAC	Medium Access Control
MAHN	Mobile Ad Hoc Networks

MCN	Mobile Cellular Networks
MCU	Micro-Controller Unit
MEC	Mobile Edge Computing
MIT	Massachusetts Institute of Technology
MITM	Man-In-The-Middle attack
ML	Machine Learning
MPC	Model-based Predictive Control
MQTT	Message Queue Telemetry Transport
MTU	Maximum Transmission Unit
OASIS	Advancement of Structured Information Standards
OSINT	Open-Source Intelligence
PLA	Physical Layer Authentication
PLS	Physical Layer Security
RES	Renewable Energy Sources
REST	REpresentational State Transfer
RF	Radio Frequency
RFI	Radio Frequency Interference
RFID	Radio Frequency Identification
SBC	Single Board Computer
SCADA	Supervisory Control And Data Acquisition
SCM	Supply Chain Management
SD	Survival Duration
SIG	Bluetooth Special Interest Group
SoA	Service-oriented Applications
SSDT	System Service Dispatch Table
SSID	Service Set Identifier
TCP	Transmission Control Protocol
TinyML	Tiny Machine Learning
UDP	User Datagram Protocol
UI	User Interface
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle

VEM	Virtual ElectroMagnetic
VLAN	Virtual Local Area Network
W3C	World Wide Web Consortium
WLAN	Wireless Local Area Networks
WPAN	Wireless Personal Area Networks
WSN	Wireless Sensor Networks
XML	eXtensible Markup Language
XMPP	eXtensible Messaging and Presence Protocol
XSS	Cross-Site Scripting

Acknowledgements

Henry Ford said, "You say I started out with practically nothing, but that is not correct. We all start with all there is; it's how we use it that makes things possible". Most PhD students say the same things about their PhD studies, but surely and truly, we all started with something that makes us reach the final destination of honouring the PhD.

Undertaking this PhD has been a truly life-chaining experience for me, and it would not have been possible to do without the support and guidance that I received from many people. First and foremost, I am extremely grateful to my supervisor, Prof.Rocco De Nicola, for his invaluable advice, continuous support, patience, motivation, and immense knowledge during my PhD study. I would like to express my sincere gratitude to my co-advisor Dr.Simone Soderi for his treasured support, which was really influential in shaping my experiment methods and critiquing my results. I also extend my gratitude to Prof. Gabriele Costa for his invaluable advice during the initial stages of my PhD studies. My gratitude extends to the IMT School for Advanced Studies for the funding opportunity to undertake my studies at the Department of Computer Science and System Engineering. Additionally, I would like to thank all the members of the PhD office, IMT Facilities, and IMT library. Their kind help and support have made my study and life in the IMT a wonderful time. A special thanks to Barbara Iacobino for being there to help my family with visas and other things. I am grateful to Dr. Ornella Bucci for her kind assistance and support, which significantly impacted my PhD studies in a positive way. Thank you very much, Daniela Giorgetti, for your kind assistance in various aspects.

Part of the present thesis is based on the articles published and listed in the publications section. The work presented in the thesis is based on the article co-authored by Prof. Rocco De Nicola and Dr.Simone Soderi. I sincerely thank all my co-authors who inspired me and for the fruitful discussions.

Moreover, I would like to acknowledge the funding from the Erasmus+ program of the European Union for the traineeship, which was utilized for three months blended visiting period, including fifteen days physically at the University of Oulu (Finland) starting in April 2023.

I would also like to express my gratitude to Prof.Romano Fantacci and Prof.Alberto Lluch Lafuente for their valuable time spent reviewing my thesis and for providing insightful comments that significantly enhanced the quality and content of the thesis.

To all my friends, thank you for your understanding and encouragement in my many, many moments of crisis. Your friendship makes my life a wonderful experience. You all were like a second family to me through this PhD. Thanks to Dr.Heather Formaini for supporting me in many different aspects. Thanks a lot, Dr.Surya, for being my second sister, being beside me, and for your love and care for my child. Thank you so much, Dr.Maria; you are the one I can not forget in my life, as I can still remember your help and support during my pregnancy in Italy. Thanks a lot, Dr.Pavan, for helping me in different aspects and for your support through the first six months in Italy and the rest of my PhD studies. Thank you, Francis John, for your support through my first months in Italy. Mohadesah and her family (Dr. Hamid and their lovable child Ameer), Anil, Francesca Fine, Citrah, Dr.Sampath, Mengjia, Shaima, Kristina, Sam, Ivana, and Ajinkya for being beautiful friends to me. Thanks to all my other friends and those who supported me with my father's case. You always made a difference.

Last but not least, I want to thank my family for the encouragement which helped me complete this thesis. A big thanks to my parents for their love and support throughout my life. Thank you both for giving me the strength to reach for the stars and chase my dreams. I would like to honour this PhD to the one who encouraged me to be like what I am today, my father "Ali Alwaisi." To my mother, "Elham Alkhfaji", who never left me alone. Thanks a lot to my sister, Lawyer. Israa, and my brothers' Lawyer. Hussein, his beautiful fiancée Zahraa and Dr.Abass, you also deserve my wholehearted thanks. Big love and a special thanks to my beloved and supportive husband, Mohammed, who is always by my side when I needed him most and helped me a lot in this journey, and my lovable child, Ali, who inspired me to pursue this undertaking

and inspired me to 'work-smart.' I also want to thank my grandmother for being in my dreams. I know you are now proud of me, and if you are here, you will be the happiest person in this world. Thank you all for the strength you gave me.

Not to forget, expressing gratitude for the negative people I've met in my life is essential, as these experiences have significantly shaped my path towards success. Without the challenges posed by these individuals, I might not have been pushed to maximize my potential and achieve my best in life. I've used their words as stepping stones on my journey towards earning my PhD degree.

Finally, I extend my gratitude and blessings to all those who supported me in any respect during the completion of my PhD study.

Vita

April 29, 1993	Born, Babel (Ḥïlla), Iraq
09/2011-01/2013	B.Sc., Information Technology (Software Engineering) University of Babylon Babel, Iraq
09/2013-07/2014	Pre-sessional English course, English language course University of Northampton Northampton, United Kingdom
09/2014-07/2016	B.Sc., Software Engineering University of Northampton Northampton, United Kingdom
09/2016-09/2017	M.Sc., Software Engineering University of Northampton Northampton, United Kingdom
11/2018-09/2023	PhD Student "Includes One-year Maternity leave", IMT School for Advanced Studies Lucca, Italy
04/2023-07/2023	Visting Period at the University of Oulu, Oulu, Finland Lucca, Italy

Publications

This thesis is based on the following papers:

1. **Zainab Alwaisi**, Simone Soderi, Rocco De Nicola, "Energy Cyber Attacks to Smart Healthcare Devices: A Testbed", EAI BICT 2023: 14^{th} EAI International Conference on Bio-inspired Information and Communications Technologies, (2023).
2. **Zainab Alwaisi**, Simone Soderi, Rocco De Nicola, "Detection of Energy Consumption Cyber Attacks on Smart Devices" at EAI FABULOUS 2023: 7^{th} EAI International Conference on Future Access Enablers of Ubiquitous and Intelligent Infrastructures, (2023).
3. **Zainab Alwaisi**, Simone Soderi, Rocco De Nicola, "Mitigating and Analysis Memory Usage Attack in IoE system", INISCOM 2023: 9^{th} EAI International Conference on Industrial Networks and Intelligent Systems, (2023).

Conference/Presentations

1. **Zainab Alwaisi**, Simone Soderi, Rocco De Nicola, "Energy Cyber Attacks to Smart Healthcare Devices: A Testbed", BICT 2023: 14^{th} EAI International Conference on Bio-inspired Information and Communications Technologies, at Okinawa, Japan.
2. **Zainab Alwaisi**, Simone Soderi, Rocco De Nicola, "Detection of Energy Consumption Cyber Attacks on Smart Devices" at EAI FABULOUS 2023: 7^{th} EAI International Conference on Future Access Enablers of Ubiquitous and Intelligent Infrastructures, at Zagreb, Croatia.
3. **Zainab Alwaisi**, Simone Soderi, Rocco De Nicola, "Detection of Energy Consumption Cyber Attacks on Smart Devices" at EAI INISCOM 2023: 9^{th} EAI International Conference on Industrial Networks and Intelligent Systems, at Ho Chi Minh City, Vietnam.
4. **Zainab Alwaisi**, Simone Soderi, Rocco De Nicola, "Energy Cyber Attack to smart healthcare devices" at IMT School for advanced studies 2023: Research Seminar, at IMT, Lucca.
5. **Zainab Alwaisi**, Simone Soderi, Rocco De Nicola, "Energy Cyber Attack to smart healthcare devices" at University of Oulu 2023: Research Seminar at CWC centre, at Oulu, Finland.
6. **Zainab Alwaisi**, Simone Soderi, Rocco De Nicola, "Detection of Energy Consumption Cyber Attacks on Smart Devices" at University of Oulu 2023: WEALTH Research Seminar, at Oulu, Finland.

Abstract

This research takes place in the context of the optimized monitoring and detection of Internet of Things (IoT) resource-constraints attacks. Meanwhile, the Internet of Everything (IoE) concept is presented as a wider extension of IoT. However, the IoE realization meets critical challenges, including the limited network coverage and the limited resources of existing network technologies and smart devices. The IoT represents a network of embedded devices that are uniquely identifiable and have embedded software required to communicate between the transient states. The IoT enables a connection between billions of sensors, actuators, and even human beings to the Internet, creating a wide range of services, some of which are mission-critical. However, IoT networks are faulty; things are resource-constrained in terms of energy and computational capabilities. For IoT systems performing a critical mission, it is crucial to ensure connectivity, availability, and device reliability, which requires proactive device state monitoring.

This dissertation presents an approach to optimize the monitoring and detection of resource-constraints attacks in IoT and IoE smart devices. First, it has been shown that smart devices suffer from resource-constraints problems; therefore, using lightweight algorithms to detect and mitigate the resource-constraints attack is essential. Practical analysis and monitoring of smart device resources' are included and discussed to understand the behaviour of the devices before and after attacking real smart devices. These analyses are straightforwardly extended for building lightweight detection and mitigation techniques against energy and memory attacks. Detection of energy consumption attacks based on monitoring the package reception rate of smart devices is proposed to detect energy attacks in smart devices effectively. The proposed lightweight algorithm efficiently detects energy attacks for different protocols, e.g., TCP, UDP, and MQTT. Moreover, analyzing memory usage attacks is also considered in this thesis. Therefore, another lightweight algorithm is also built to detect the memory-usage attack once it appears and stops. This algorithm considers monitoring the memory usage of the smart devices when the smart devices are *Idle*, *Active*, and *Under attack*. Based on the presented methods and monitoring analysis, the problem of resource-constraint attacks in IoT systems is systematically eliminated by parameterizing the lightweight algorithms to adapt to the resource-constraint problems of the smart devices.

Chapter 1

Introduction

The term "Internet of Things (IoT)" was first coined by Kevin Ashton, the Executive Director of Auto-ID laboratories at the Massachusetts Institute of Technology (MIT), in 1999 [1]. Kevin Ashton predicted that computers or other objects would be able to gather information without human intervention during his presentation for the Procter and Gamble supply chain system. Since then, the IoT has emerged as one of the most promising research disciplines in the field of smartness in this past decade [2]. Given the smartness and real-time monitoring, which do not require human intervention, IoT services have greatly relieved human life. IoT has many applications, such as smart homes, smart transportation, smart agriculture, supply chain systems, smart metering, smart grids, smart healthcare systems, industrial automation, smart retail, and more. The number of smart devices is expected to increase to 3.6 billion in 2025 [3, 4]. According to the International Data Corporation (IDC), the collective sum of the global data could grow from 33 ZB in 2018 to 175 ZB in 2025, where 90 ZB of data will be created on IoT devices by 2025 [5]. IoT applications are frequently employed because they offer high efficiency, automation, and comfort. Many IoT devices and users are connected to the Internet, generating huge volumes of data. However, it exposes systems to fundamental security problems in relation to confidentiality, integrity, and availability. To successfully use the ever-growing IoT applications, security, privacy, and trust must be addressed to secure IoT devices and user privacy from attackers [2].

An IoT is a smart network connected to various other smart devices and data centres. An IoT device continuously gathers and analyzes data with the help of sensors from the surrounding environment. The majority of devices are autonomous and function with minimal human intervention. Some smart devices have unique identifications for authentication [6]. Low power and low memory consumption are the main characteristics of IoT devices. Intelligent devices can be placed in remote areas for data collection and transmission. IoT application is expanding into Artificial Intelligence (AI), cloud, big data, and smart systems such as smart homes and offices. An IoT system typically consists of three major stages [7, 8, 6]: collecting, transferring, and analyzing data. The first stage involves data collection and transmission, where sensor antennas and microcontrollers are involved. This stage is also known as the physical layer. The second stage concerns data transfer to an IoT hub or a gateway using a network. The last stage is data analysis, which comprises user data and other

data sources which may be hosted in the cloud. Any breach in those stages will lead to critical information leaks from sensor-based devices. Recently various vulnerabilities started to emerge in smart devices due to the lack of advanced encryption and authentication systems; Chapter 2 discusses IoT, IoT security, and application vulnerabilities in detail.

In IoT systems, *energy* and *memory consumption attacks* are becoming frequent and detecting such attacks is necessary to protect the IoT system from vulnerability threats that could lead to accessing the home network and attacking the smart devices. The resource-constrained problems make IoT systems vulnerable to different attacks affecting resources, e.g., energy, memory, Central Processing Unit (CPU), etc. Although many IoT applications are not time-sensitive, there is a whole class of mission-critical applications, particularly those that target human safety where timely intervention is essential. Examples are applications for critical control, health monitoring, and fault detection [9, 10, 11]. Taking health monitoring as an example, as illustrated in Figure 1, the focus lies on aggregating, gathering, and extracting information related to a patient's health, such as heart rate. The collected data from sensors is then stored in a cloud or a Database (DB). Afterwards, the fetched data can be used for further calculations. For example, ambulances could automatically be notified in emergencies and locate the patient through Global Positioning System (GPS) signals. High reliability is critical in this situation, where data must be processed and shared immediately and within strict constraints. Unfortunately, resource constraints impose hard-duty cycles

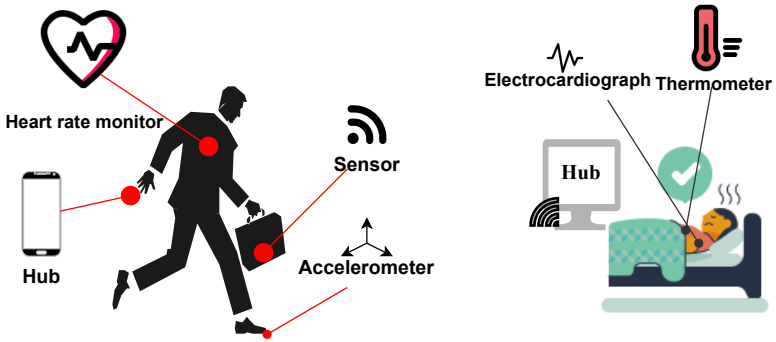

Figure 1. Health Monitoring with IoT.

to maximize longevity [12], which can cause unreliable connectivity. According to the IEEE, reliability definition is "the ability of a system or component to perform its required functions under stated conditions and for a specified time [13]." Therefore, it is mandatory to monitor the resources of the IoT system, analyze attack effects in terms of energy and memory constraints and build lightweight algorithms to detect resource constraint attacks. The absence of any monitoring and mitigation mechanisms for detecting device faults would dramatically reduce the performance of an IoT network, which renders *monitoring* the IoT devices a vital research area that will significantly develop the security of the IoT systems. An effective and efficient resources monitoring and detection mechanism could significantly improve the robustness of devices, IoT devices connectivity, and reliability, which will significantly increase stakeholders' uptake of the technology.

1.1 Research Motivation

IoT is a global network and service architecture with connectivity and self-configuring capabilities based on open-standard and interoperable protocols. The IoT consists of heterogeneous objects with identities and physical and virtual properties that are securely integrated into the Internet [14, 11]. The main goal of the IoT is to enable things to be connected anywhere, anytime, with anything. IoT created many applications that touch every aspect of human life by connecting billions of things to the Internet. For example, health monitoring, wearables [15], military applications such as intrusion detection in remote or hostile environments, smart homes, smart cities [16], smart grids [17] and others. Smart devices suffer from resource constraints, such as energy and memory limitations. Consequently, attackers employ various techniques to gain unauthorized access to the data stored on these devices or cause damage by exploiting their resources. Therefore, making IoT devices available for the end-users is critical, and preventing resource constraint attacks is essential. This thesis aims to analyze the effect of resource-constraint attacks in IoT systems and to develop solutions that consider the resources of these devices and protect them from resource-constraint attacks. Any similar smart devices with the same architecture as industrial IoT devices or on the consumers' side can utilize and apply the concept of the presented solutions in this thesis. A monitoring system has been developed to control and detect resource-constrained attacks in smart devices. In the initial stage, the monitoring system is utilized to analyze the impact of resource constraint attacks, such as energy and memory attacks. Then, depending on the collected results by the first stage, lightweight detection algorithms are introduced to monitor energy and memory usage and detect an attack from start to end.

The cybersecurity threats in IoT systems are broadening and becoming increasingly sophisticated. Hacking and destroying smart devices could compromise systems availability or, worse, lead to fatal accidents that could affect people. The scenario considered in this thesis concerns the security of smart devices against resource constraint attacks. The final results show high efficiency in detecting resource constraint attacks. This work studies and analyzes the effect of resource-constraint attacks on IoT devices regarding communication protocols, attack rates, payload sizes, and victim devices' ports state as the vital factors in determining victim devices' energy and memory consumption. Early detection of resource-constraint attacks on IoT devices is performed, as described in Chapter 3. A lightweight algorithm has been developed to detect energy consumption attacks in IoT systems, considering three different protocols: Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Message Queue Telemetry Transport (MQTT) protocol. Further details regarding this algorithm are reported in Chapter 4. Additionally, an analysis and detection algorithm for memory usage attacks in IoT systems is presented in Chapter 5, which involves calculating memory usage tasks to determine normal and abnormal behaviours.

1.2 Research Objectives and Methodology

1.2.1 Resource Monitoring Requirements

In general, the monitoring mechanisms deployed in IoT networks and smart devices have two primary objectives: detecting and localizing network faults and identifying attacks

on smart home devices [11]. To accomplish these goals, it is essential to provide suitable tools and capabilities for overseeing the state of devices and the network, ensuring the availability of smart devices, maintaining connectivity between different devices and nodes, and detecting potential attacks that exploit the resource constraints of smart devices. Implementing a robust monitoring infrastructure makes it possible to identify the root causes of problems and map their symptoms, enabling appropriate corrective actions to be taken. Therefore, the monitoring infrastructure should encompass the entire smart home network domain to effectively detect resource-limiting attacks on smart home devices [11]. IoT devices and networks operate at an immense scale, potentially encompassing millions or even billions of devices and network nodes. To ensure comprehensive detection of resource constraint attacks, it is crucial to embed resource monitors in appropriate locations throughout the network strategically. Given the limited resources of smart devices, minimising the costs associated with monitoring and detection is imperative, including energy consumption and memory usage. Monitoring the energy and memory constraints is essential for maintaining the availability of smart devices in both home and industrial wireless environments and enabling the early detection of resource consumption attacks. Furthermore, it is essential to analyze the impact of resource constraint attacks and study the behaviour of smart devices when subjected to such attacks. This analysis can facilitate the development of suitable detection algorithms designed explicitly for resource constraint attacks.

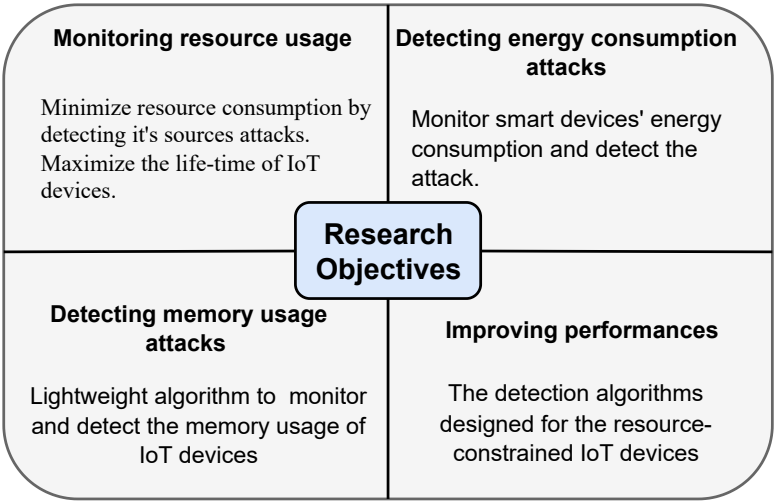

Figure 2. Research Objectives.

To review the requirements for the proposed solutions as shown in Figure 2:

- monitors should be able to analyze the effect of resource constraint attacks to study their behaviour in the IoT environment;
- monitoring the resources usage should allow one to detect different behaviours of the smart devices, e.g., *Idle*, *Active*, *under attack*;

- the monitoring of energy and memory constraints should be minimal to satisfy the low-cost, low-power IoT devices;
- improves the performance by designing the detection algorithms for resource-constrained IoT devices.

1.2.2 Research Methodology

To achieve the stated goals, the research methodology employed is as follows (Figure 3):

- extensive studies of state-of-the-art monitoring and detection of the resources of IoT devices and especially the IoT energy consumption and memory usage for different protocols;
- identifying monitoring requirements and research goals;
- modelling and formulating the corresponding IoT resources optimization problem;
- creating precise analytical solutions for relevant issues and evaluating proposed models for complexity and reliability;
- extensive tests on real smart devices for performance evaluations to confirm the efficiency and effectiveness of the proposed models;
- definition of lightweight algorithms for detecting resource constraint attacks in IoT devices.

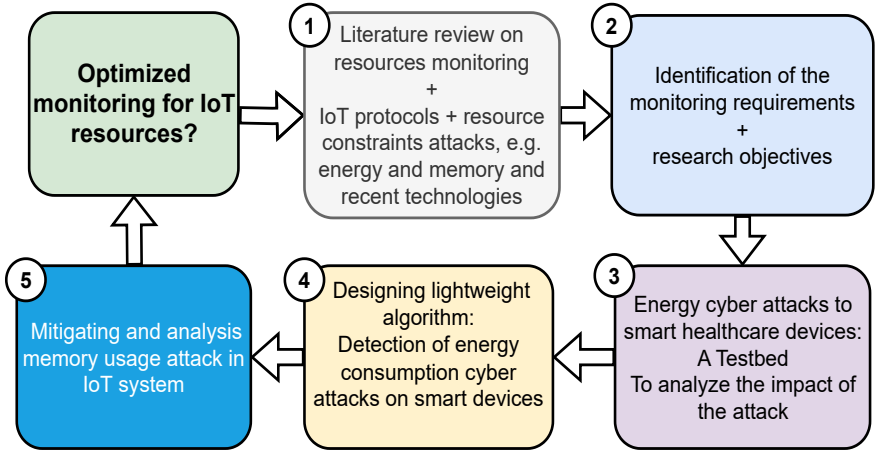

Figure 3. Research Methodology.

1.3 Research Contributions

The aim is to target resource-efficient monitoring of smart devices in IoT systems, which contributes to energy and memory savings and enables the detection of resource constraint

attacks. The first contribution is experimenting with energy consumption attacks on smart home devices. It aims to observe the behaviour of smart devices, protocols, and networks and infer their states.

It can also collect different information about the devices like *online* or *offline*, Internet Protocol (IP) address, and Medium Access Control (MAC) address. Also, identifying port status (open/closed, filtered/not filtered, and others). Moreover, monitoring the smart devices' energy consumption and checking their connectivity once disconnected from the main Access Point (AP) and once it connects to the Fake Access Points (F-APs) attacks. This work better explains the effect of Distributed Denial of Services (DDoS), energy-consumption Distributed Denial of Services (EC-DDoS), and F-AP attacks on smart healthcare devices within a wireless network. In this work, a practical combination of DDoS and F-AP attacks was designed to impact the energy of real smart devices, e.g., Raspberry Pi[1] and Arduino[2]. It also offers a better understanding of monitoring the resources of smart devices and analyzing the impact of the attack for building lightweight algorithms to detect energy attacks in IoT systems. This contribution is a baseline for the next contribution, as it helps to fully understand the resources' monitoring of IoT devices and build a detection mechanism based on the packet reception rate analysis.

Therefore, in the second contribution, an algorithm was designed to detect energy consumption attacks in smart devices. The technique employed in this contribution aims to detect energy consumption attacks on smart home devices by monitoring the packet reception rate and energy consumption to determine the final results' status as normal or abnormal behaviours. The algorithm considers different protocols and device statuses to detect the attack, including TCP, UDP, and MQTT. The algorithm shows high efficiency in detecting energy consumption attacks in smart home devices compared to other algorithms that use the current energy consumption measurement for detecting this attack. As this algorithm is easy to use and not expensive to implement, it also considers the resource constraints of smart devices. The key observations made from this work present a thorough understanding of the packet reception rate of IoT devices within a home wireless environment. Moreover, it shows the detection of energy consumption attacks depending on measuring the packet rate received by the smart device. This contribution offers a better understanding of studying the packet reception for different protocols of smart devices and detecting energy consumption attacks.

The main goal of this thesis is to protect smart IoT devices from resource-constrained attacks. Therefore, the work presented in Chapter 5 studied the effect of memory usage attacks on smart devices and analyzed the impact of the presented attack to build a lightweight algorithm to protect the smart device from such attacks. This contribution considers different behaviours on the memory of smart devices. Memory usage is measured under different scenarios, including read and write operations, with or without the attack, to evaluate the best detection of memory usage attacks. A mitigation technique is also simulated, and the results are assessed by implementing the proposed technique on smart devices such as Raspberry Pi and Arduino. The current memory usage of the smart device is also measured to monitor the memory usage to discriminate between normal and abnormal behaviours. Therefore, this algorithm design is a protection strategy for smart devices to maintain their integrity,

[1]https://www.raspberrypi.com/documentation/
[2]https://store.arduino.cc/

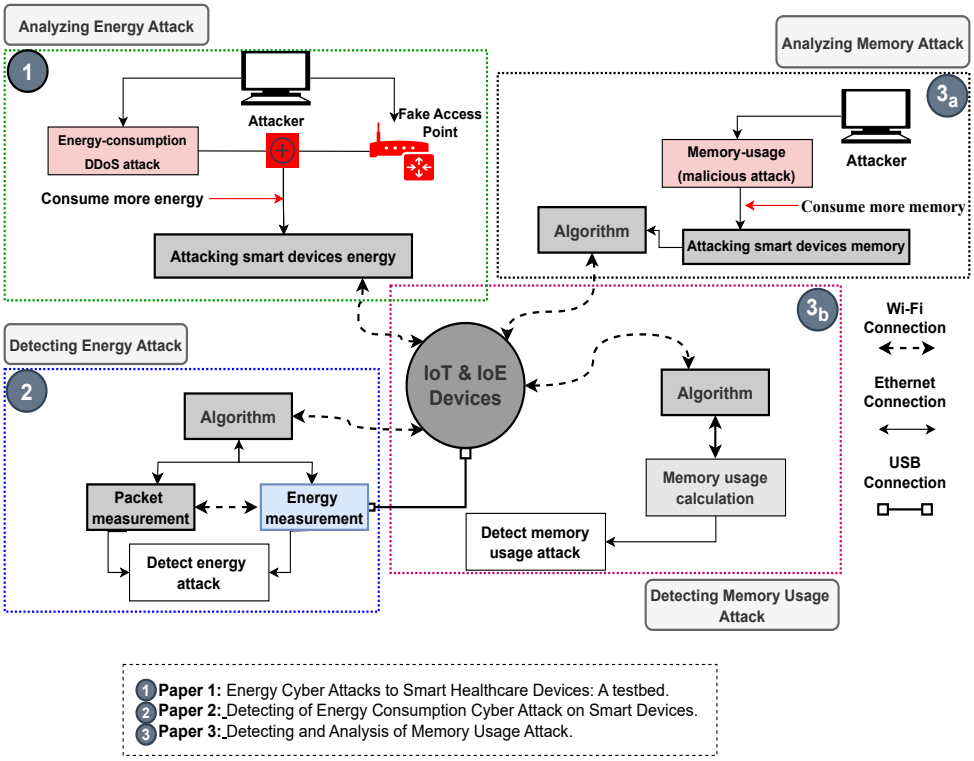

Figure 4. Main Topics covered in this thesis.

seamlessly make them available to legitimate users, and protect them from memory attacks by considering their resource constraints. The results demonstrate high efficiency in analyzing, detecting, and mitigating memory usage attacks in smart devices within the IoT environment, specifically in the pillar of "Things" in the Internet of Everything (IoE) framework. This experiment can be extended to similar smart devices with the same architecture as this study uses. The IoE, which builds upon the "four pillars" of people, data, process, and things [18], extends business and industrial processes to enhance people's lives [19]. Thus, the monitoring analysis and detection methods can be further extended to include other smart devices within the IoE environment. Chapter 2 gives more information about IoT and IoE environments.

To summarize the contributions made towards addressing the challenges in monitoring and detecting IoT resources efficiently (as illustrated in Figure 4 and 5), the proposed solutions are as follows:

- Energy Cyber Attacks on Smart Healthcare Devices [20] aim to enhance our understanding of the impact of DDoS, EC-DDoS, and F-APs attacks on smart healthcare devices' energy consumption and connectivity within a home wireless network. Ad-

ditionally, this study analyzes the effects of energy consumption attacks within a smart home system.

- The Detection of Energy Consumption Cyber Attacks on Smart Devices [21] aims to monitor the network, ports, device statuses, energy consumption, and packet reception rate of smart devices. The key observations from this study provide a comprehensive understanding of the packet reception rate of IoT devices within a home wireless environment and how energy consumption attacks can be detected by measuring the received packet rate from smart devices.
- Detection and Analysis of Memory Usage Attacks in IoE Systems [22] involve monitoring the memory usage of smart devices to understand the impact of memory usage attacks on these devices. Additionally, a lightweight mitigation mechanism is developed to detect and mitigate memory usage attacks promptly when the attack begins and ceases.

The achieved results are presented in Chapter 3,4, and5, contributing to the overall understanding of the topic.

1.4 Thesis Outline

This thesis investigates the security of the IoT system, particularly the security of IoT devices. It consists of seven chapters, including the introduction, conclusions, and appendix. The main aim of the thesis is to contribute to securing IoT devices.

Chapter 2 presents background information regarding IoT and IoE, including IoT and IoE security and attack mitigation, and a description of some IoT and IoE applications. It also presents related studies about resource constraint attacks in IoT systems and discusses current detection mechanisms.

Chapter 3 describes the energy monitoring system, presents and analyzes the energy attack, and displays the results of the effect of the energy cyberattacks on real IoT devices.

Chapter 4 presents the techniques used to detect energy consumption attacks by studying the packet reception rate and showing the final results of mitigating and detecting such attacks.

Chapter 5 explores memory usage attacks and their impact on smart sensors. Detection methods are also presented to mitigate memory usage attack effects on smart devices.

Chapter 6 wraps up the thesis with some concluding remarks and introduces possible topics for future research.

Finally, Chapter 7 contains essential details about the databases utilized in the experiments of this thesis. Additionally, it provides comprehensive information about the energy consumption tools and technologies employed to measure smart devices' energy and memory usage.

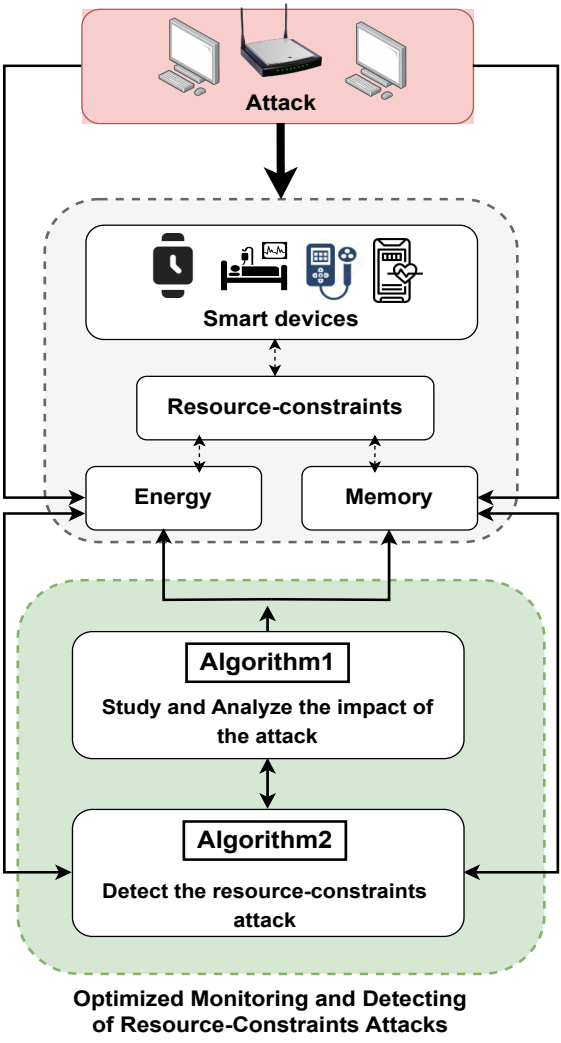

Figure 5. Main Contributions.

Chapter 2

Background and Related Studies

Wireless communications are widely used in transportation, military, industrial, and healthcare applications and play a significant role in industry and the whole of society. The IoE is a superset of the IoT, which means a connection between people, processes, data, and things. It connects all these concepts into one cohesive world. IoE builds on the pillars of IoT, which include an intelligent network system. IoE is, therefore, a global network through which people, things, and intelligent devices are connected and can share information and services. The IoT paradigm, which heavily relies on wireless communication, has already received considerable attention. IoT is considered, in fact, to be one aspect of the future, and it could have a very significant impact on our lives. Some aspects of our world may be able to improve thanks to the benefits provided by IoT. Looking at what might be called a downside, however, it is possible to see that there could be certain security issues amongst all of these connections. The vulnerability of IoT sensors to hacking is a significant concern. Due to their limited computing, storage, and network capabilities, they can be more susceptible to attacks compared to endpoint devices like tablets and smartphones. Therefore, it becomes crucial to establish robust security and privacy protocols in IoT networks to safeguard the integrity, authenticity, and confidentiality of IoT-based services. Ensuring confidence in the security measures deployed is essential for maintaining the trustworthiness of these services.

This chapter considers IoT and IoE potential security attacks and discusses existing solutions that deal with some issues. Moreover, related work and background reading about Chapter 3, 4, and 5 reviewed in this chapter as well.

This chapter is organized as follows. Section 2.1 provides a general introduction to IoT and IoE, the different security issues in the literature, and sheds some light on the current security solutions. Section 2.2 presents the definition of IoE, expectations, applications, and challenges. Section 2.3 and 2.4 provide different information about IoT, such as its definitions, relevant technologies, architecture, IoT standards and protocols, applications, security challenges, and current mitigation of the IoT. Finally, the related work and background reading of Chapter 3, 4, and 5 are presented in Section 2.5.

2.1 Introduction

The IoE paradigm is based on the convergence of the digital and physical world to make this world smarter with intelligence, cognition, and connectivity. IoE is a system that interconnects billions of heterogeneous physical devices, computing elements, objects, animals, and humans that can set up, share, and self-organize their limited resources to achieve a system-wide goal [23]. The main objective of IoE networks is to enhance the performance of the underlying IoT physical infrastructures by providing services to humans [24]. The IoT is the foundation for a wide range of intelligent application domains, such as smart cities, healthcare, and transportation. It plays a crucial role in enabling the three key aspects of a smart city, known as the 3Is: Instrumentation, Interconnections, and Intelligence. The IoT is fundamental to achieving these components. On the other hand, the IoE is a broader concept that includes the IoT as one of its integral components [18]. IoT is a network of physical devices, vehicles, appliances, and other objects embedded with sensors, software, and connectivity capabilities that can collect and exchange data over the Internet. The smart devices in IoT environments typically focus on specific functions and interactions between devices, such as smart thermostats, wearable fitness trackers, or home security systems [25]. On the other hand, IoE represents a broader concept that extends beyond just physical devices. It encompasses the convergence of people, processes, data, and things, as shown in Figure 6, creating an interconnected digital ecosystem. IoE encompasses not only IoT devices but also includes social, mobile, cloud, and other technological elements. It aims to create intelligent and dynamic connections across various domains, such as healthcare, transportation, manufacturing, and more. In terms of cybersecurity, both IoT and IoE pose significant challenges [26]. However, IoE introduces additional complexities due to the interconnected nature of various elements, e.g., scale and complexity, ecosystem collaboration, system resilience, data security and privacy, etc. The main reason behind the security challenges of IoE compared to IoT is IoE involves a more extensive network of interconnected devices, services, and platforms, resulting in a larger attack surface and increased complexity [27].

The IoT is a collection of computing devices connected through the Internet. Such a networking system provides communication capabilities to objects used in everyday life. The IoT sensors have unique identifiers and can exchange data without human intervention. IoT plays an essential role in our daily life by exploiting Wireless Sensor Networks (WSN) capabilities [28, 29]. IoT will allow more data to be collected from the physical world. The commercialisation of the next wave of AI technologies will surely be accompanied by Machine Learning (ML) from IoT data. With its rapid evolution, the IoT can be considered an enabling technology in many applications, such as transportation, military, industrial, and healthcare. The primary purpose of these applications is to improve the quality of daily life. Indeed, IoT supports objects to acquire computing capabilities and make decisions based on the data exchanged over the Internet. Some of these actions need to be done by a person, but other activities can be used to facilitate the comfort of human lives [30]. The hardware techniques evolution, such as expanding the module's bandwidth by incorporating cognitive radio-based networks to address the under-utilization of the frequency spectrum, are supporting the tremendous growth of IoT [31, 32, 33].

In this scenario, IoT systems generate massive amounts of data transmitted via a networking infrastructure in which many computing devices communicate among them. It poses another serious problem because, in such a hyper-connected society, the risk of be-

ing a target of a cyber-attack can be very high, and generating a large amount of data in IoT systems can become very attractive to hackers. Indeed, these data might contain personal information or information used by automated processes employed for automatic control, e.g., in smart cities. Furthermore, evaluating the cyber risk in such a distributed network is challenging because an attacker could take away customers' data. Security attacks may occur in the central unit, i.e., server-side, the sensors, and the communication. The security of WSN, Machine-to-Machine (M2M), and Cyber-Physical System (CPS) is rising in the context of IoT with IP which is the primary standard for connectivity. The entire architecture must also be secured from attackers who might threaten the data's privacy, integrity, authenticity, or confidentiality [33]. Today, IoT developers and manufacturers must cope with these problems by introducing security by design and mitigating security issues. Several techniques can be used to protect the data, the devices, the protocols, or the server by resorting to 1) *access control* (e.g., passwords, two-Factor-Authentication (2FA), etc.); 2) *built-in restrictions* embedded by the manufacturer; 3) *cryptography* (e.g., utilization of security protocols that encrypt the data exchanged between the IoT device and the server); 4) *network security* (e.g., firewalls implementation on the server-side); 5) *confidentiality* (e.g., data stored in the cloud, etc.). This chapter reviews IoT and IoE in different aspects, such as IoT and IoE applications, security and challenges of IoT and IoE, and current studies on security mitigation of IoT and IoE. This chapter represents an introduction to the problem statement and a solid foundation for the main solutions of smart devices in the next chapters.

2.2 The Internet of Everything (IoE): An Overview

2.2.1 The birth of IoE

Undoubtedly, the development of the IoT stimulated the creation of the IoE concept. IoT refers to the process of connecting various types of electrical or electronic equipment to the Internet. Recently, IoT technology has rapidly advanced by focusing on connecting M2M communications across various communication protocols, networks, and applications. This includes technologies such as 802.11ah, Industrial IoT, and NB-IoT. [34]. Such prosperous IoT ecosystems pave a solid foundation for the IoE's communications with broad coverage and ubiquitous connection. However, the real birth of the IoE concept comes from enabling automated machines through ubiquitous Internet, big data processing, and AI. Back in 2012, Cisco offered the idea that the IoE is based on the "four pillars" of people, data, process, and objects [18] [35]. This perspective highlights that the IoE encompasses a holistic interconnection of not just "things" but also automated processes and human-based interactions. It extends beyond the IoT's scope of merely connecting devices and machines to include the integration of intelligent machines and people-driven processes [18]. Moreover, the proliferation of big data and AI technologies brings new bricks for IoE's construction. In recent years, more relevant literature has replenished IoE's essence, e.g., gathering big data that is hidden from the Internet, utilizing different AI algorithms, and enabling all devices and machines with automated abilities [36] [37] [38] [39]. IoE has the ability to enable *automated intelligence* by extracting and analyzing real-time data from millions of linked devices and then making smart, proactive decisions. While the IoE concept has been presented and discussed for a long time, its implementation is still in its early stages. Despite challenges in the

full realization of IoE, the attractive vision of IoE will never prevent us from implementing IoE. The following sections discuss the definition of IoE, enabling technologies, and challenges.

2.2.2 Definition of IoE

IoE represents an expansion of the IoT, encompassing the interconnection of people, processes, data, and things, as depicted in Figure 6. It brings together these elements to form a unified ecosystem. Building upon the foundations of the IoT, the pillars of IoE are characterized as follows:

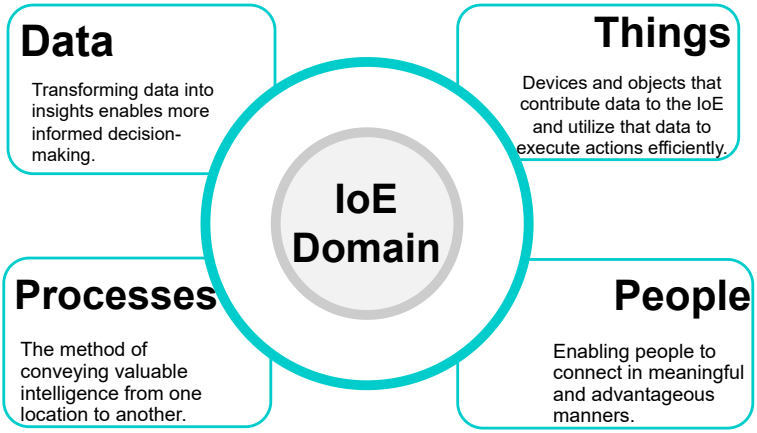

Figure 6. Internet of Everything domains.

- People: people play a vital role in the IoE environment, contributing their personal insights through various means of communication such as smart sensors, social networks, and smartwatches. These data are collected and analyzed on servers to provide relevant information for personal, industry, or business needs. This enables timely decision-making and issue resolution [40].
- Data: data transmission in the IoE environment follows a similar pattern to traditional IoT networks. The collected data from devices can be transmitted directly or undergo initial processing at the edge layer. While raw data from smart devices may not hold significant value, once transformed, classified, and analyzed either by the device itself or the cloud server at the edge layer, the resulting data becomes invaluable content. This data enables control and monitoring and facilitates accurate and swift decision-making, empowering smart solutions [40].
- Process: the process is based on various systems such as AI, computer vision, deep learning, social networks, or other technologies that help to deliver the proper information to the designated people, devices, or places at the expected time. This process

will extract information from data, and the network will control the data communication. The main purpose of processes is to get the optimum outcome for further processing or decision-making.

- Things: things encounter the definition of IoT. Different sensing components are embedded with physical items that serve the purpose of data collection. Smart devices must have communication capabilities, e.g., wireless or wired, for transmitting data, e.g., generated and processed data, to the right destination across the IoT system.

The massive growth of smartphones, smart hospitals, smart home devices, smart grids, and population are initially key concepts of the IoE. This concept is followed by the appearance of wearable devices, computers [41], and intelligent transportation systems. Initially, the concept of IoE was coined by Cisco Systems in 2013, and it is the most popular technology in today's world. The IoT is just a tiny subset of the IoE. It describes a world where trillions of intelligent devices have sensors to verify, measure, and estimate their positions, all connected over public or private networks that use specific protocols. According to Cisco Systems, a market of devices could reach 50 billion by 2024 [42]. The communication infrastructure in an IoE environment is enhanced by using sensors or intelligent systems attached to each device. Every sensor node or intelligent system is connected using a WSN. The sensor node is used to detect various parameters, such as motion, humidity, temperature, pressure, lights, and others [43]. IoE provides advanced capabilities within the area of information sharing, but this requires appropriate measures to be taken in the initial phases of its design and implementation to be widely accepted in all domains of the IoE aspects.

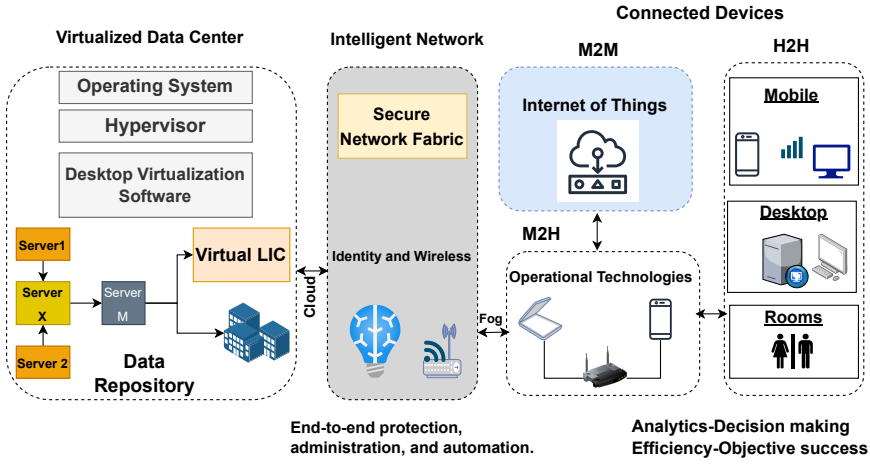

Figure 7. Comprehensive IoE Architecture.

Figure 7 depicts the overall structure of the IoE system [44] [45]. The IoE system combines blocks of a visualized data centre, connected devices, and intelligent networks. A virtual data centre consists of desktop virtualization software, an operating system, and others. The IoE system interfaces with an intelligent network to deliver services to various

interconnected smart devices, including smart sensors, actuators, mobile terminals, wearable devices, and even human users. These devices fall into three categories: machine-to-machine, human-to-human, and machine-to-human interactions. To support high-speed, low-latency, and high-quality IoE services, the system utilizes an optical fibre network as its backbone [44] [45]. Alternatively, a wireless network may be used as a substitute for the fibre-optic network [40].

2.2.3 Three Expectations of IoE

The concept of IoE is to connect electronic devices (i.e., terminal nodes of IoE) to the Internet, then analyze massive data generated from connected terminal nodes, thereby offering intelligent applications for advancing certain aspects of human society. IoE is expected to fulfil three key expectations for achieving this concept: To fulfil the IoE concept, three primary expectations have been identified. These expectations involve establishing a scalable network architecture with ubiquitous coverage, creating a global computing facility to facilitate intelligent decision-making, and supporting diverse applications, as indicated by the concept of diversity. Figure 8 showcases these three expectations and their typical enabling technologies [18].

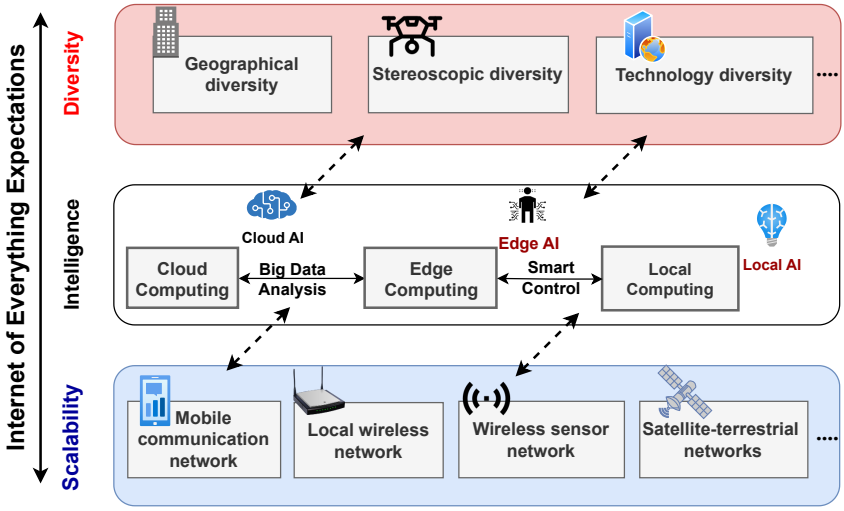

Figure 8. Three fundamental expectations of IoE (i.e., scalability, intelligence, and diversity) [18]

1. **Scalability:** Establishing a scalable network for IoE aims to create an elastic and widespread infrastructure that covers diverse geographical scenarios, such as rural, urban, underwater, terrestrial, aerial, and space environments. The primary objective of the scalable IoE network is to fulfil various communication requirements by ensuring massive access, wide coverage, and ubiquitous connectivity. This entails integrating multiple

communication technologies with varying transmission distances (ranging from a few meters to a thousand meters) and different network topologies (including star, hybrid, and point-to-point topologies). Key communication networks forming the foundation of IoE include Wireless Local Area Networks (WLAN), Mobile Cellular Networks (MCN), satellite networks, WSN, and Mobile Ad Hoc Networks (MAHN). The scalability of IoE facilitates efficient data collection and serves as a valuable resource for intelligent analytics [18].

2. **Intelligence:** enables predictions, decisions, actions, and intelligent analysis for all smart devices in IoE system. In particular, IoE needs to gather massive data from its broad and scalable network, extract useful information, e.g., decisions or smart commands from the collected data, and then use this information to enable intelligent actions or controls for everything. The computing infrastructure consists of distributed database and storage systems, on top of which different big data processing techniques are deployed. Storage systems and databases play a crucial role in storing and preserving the accumulated IoT data. To cater to various intelligent applications, diverse big data processing algorithms, such as predictive, descriptive, and prescriptive analytical schemes, are employed [41, 18]. IoE's intelligence, facilitated by distributed computing facilities, can be categorized into edge, cloud, and local intelligence. It suggests that computing resources and intelligent algorithms are strategically distributed across the edge, local (terminal nodes), and remote cloud environments. Orchestrating local, edge, and cloud intelligence becomes essential to achieve IoE's overarching global intelligence [18].
3. **Diversity:** IoE encompasses a wide range of applications that facilitate automated and people-based processes. The successful implementation of various IoE applications relies heavily on the intelligence and scalability of the IoE framework, as these factors determine computing security, energy efficiency, network performance, and overall capability. The automated and people-based processes of IoE give rise to diverse categories of applications, and depending on specific application requirements, IoE's diversity can be classified into three categories, including technology diversity, which encompasses various Information and Communication Technologies (ICT) technologies; stereoscopic diversity, which involves different spatial positions; and geographical diversity, which pertains to different geographical regions. IoE brings about additional diversities such as intelligence, equipment, and mobility, expanding its range of applications. Ultimately, these diverse applications converge to fulfil a crucial role in IoE [18].

2.2.4 IoE Applications

This section presents and discusses the relative IoE applications as follows:

A. **Smart Home**: By employing a unified communication system, it combines various home services to ensure an efficient, secure, and comfortable home operation. This integration incorporates intelligent features and offers high flexibility to cater to diverse needs. Typically, smart homes operate through web interfaces or dedicated applications. They utilize the Arduino board with an Ethernet shield or WiFi connection as the equipment control module, which is integrated into the smart home. The software component of the control

module employs communication protocols. Smart homes comprise smart devices and sensors integrated into an intelligent system, providing management, monitoring, support, and responsive services. This integration offers many benefits, including economic, social, health-related, emotional, sustainability, and security advantages.

B. **Agriculture**: IoE has gained prominence across various sectors, including agriculture, where it offers benefits such as remote monitoring and control of agricultural systems through mobile devices. This integration allows for monitoring parameters like soil moisture, water levels, humidity, and temperature using WSN. Implementing IoT-based smart systems in agriculture involves using drones, sensors, and robots for tasks like spraying and weeding and monitoring humidity and temperature [18, 46]. This interconnected system allows all smart devices to be monitored and controlled remotely via the Internet. Studies, such as the one mentioned in [47], have demonstrated the effectiveness of smart irrigation frameworks powered by renewable energy sources in significantly improving crop yield and agricultural productivity. Data collected from sensors is used to predict climate conditions using techniques like radial basis function networks. This anticipated climate information is then utilized to regulate smart irrigation systems and monitor them through a web application.

C. **Industry**: There have been limited studies on the impact of IoT or IoE technologies in the industrial environment [48]. The adoption of these technologies in such environments requires careful consideration. Implementing IoT or IoE systems in industrial settings involves balancing the need for improvement and testing while adhering to existing system principles and requirements, managing risks, and organizing tasks using the IoT or IoE platform. Solutions that aim to reduce operational expenses must prioritize stability and flexibility [18, 46]. According to [49], Cyber-Physical Systems (CPS) is one such system that meets the requirements of IoT through the utilization of cloud computing services. SCADA systems are commonly employed in industrial CPSs to monitor and control critical infrastructure. However, like any emerging internet-based system, IoT-based SCADA systems present security challenges. While they offer cost efficiency, flexibility, and scalability through the use of cloud computing, they also introduce critical risks and privacy concerns due to the storage of data in third-party-operated servers [18, 46].

Other IoE applications include smart cities, smart health, and smart environment monitoring.

2.2.5 IoE Challenges

IoE challenges are reflected in the following four constraints: security, battery, computing, and coverage constraint. The four constraints are discussed in detail as follows:

A. **Security constraint**: Many potential security threats are encountered in IoE, attributed to the vulnerabilities of communication protocols and resource limitations of IoE nodes. Specifically, the current IoE mainly adopts low-cost and simplified access protocols (i.e., NB-IoT, Low-power Wireless Personal Area Networks (LoWPAN)) to reduce network costs. At the same time, it makes the communications vulnerable to malicious attacks such as eavesdropping and forging. On the one hand, the data emitted from end nodes

can be wiretapped (or eavesdropped) by malicious nodes; on the other hand, pseudo-base-stations can easily forge the normal IoE communication links to obtain IoE data [50]. Therefore, an effective but easily deployed security mechanism is required to protect IoE communications from malicious attacks.

B. **Battery constraint**: IoE nodes or devices suffer from the battery constraint. IoE devices are generally power-limited due either to hardware cost or portability concerns. IoE uses low-power or battery-free communication technologies to access network infrastructure nodes, e.g., BS, AP, and IoT gateways. One inevitable fact is that battery-limited nodes are easily exhausted and eventually lose connections with IoE [18]. Furthermore, the adversary uses the battery constraint issues to attack these nodes by affecting their resource-constrained. Thus, developing a sustainable detecting or mitigation algorithm for battery-constrained nodes is necessary to prevent future attacks.

C. **Computing constraints**: In IoE, most terminal nodes lack sufficient computing capabilities to process local intelligent algorithms. Traditionally, the prevalent approach has been to transmit all data to remote cloud servers, which provide centralized intelligence for big data processing. However, this cloud computing paradigm introduces significant latency, especially for latency-sensitive IoE applications in the future. Additionally, the growing number of IoE computing tasks not only burdens the cloud servers but also leads to congestion in the backbone communications of IoE, raising concerns about privacy risks. To address these challenges, it is imperative to fully leverage both edge and local computing resources, complementing centralized cloud servers and enabling pervasive intelligence across all aspects of IoE [18].

D. **Coverage constraint**: The deployment of IoE communication infrastructures in rural areas poses challenges due to the difficulty and high cost involved. Consequently, achieving comprehensive coverage for IoE nodes in these areas becomes a significant challenge. Deploying existing communication networks in coverage-constrained regions is not economically viable due to the imbalance between construction costs and expected benefits. Notably, IoE projects in these areas do not necessitate continuous and ubiquitous communication. Therefore, a cost-effective solution lies in providing flexible and recoverable coverage, catering to on-demand IoE communications for specific periods in coverage-constrained areas [18].

2.3 The Internet of Things (IoT): An Overview

2.3.1 What is IoT?

IoT refers to a variety of devices that are connected through the network. Significantly, the IoT paradigm starts by providing a kind of smartness to an object and continues by providing that object with the capability to perform actions based on the exchanged data. This concept has been applied in different domains in the last few years. As shown in Figure 9, today, we have IoT applications covering transportation, smart cities, e-health, smart sensors, automotive, banking, industrial IoT, and wearable devices. The primary purpose of these smart devices is to collect data and make intelligent decisions. However, IoT applications consist of various devices that use software, hardware, communication technologies,

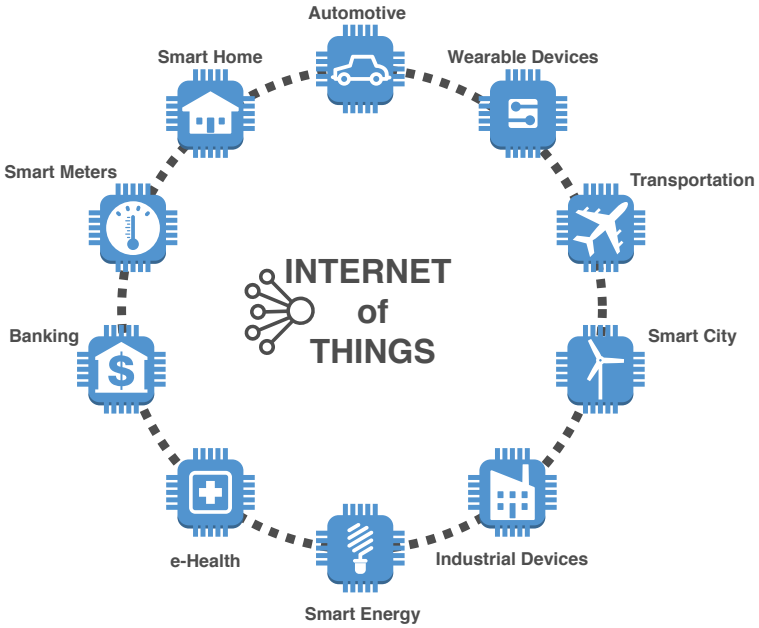

Figure 9. Internet of Things domains.

protocols, and standards. The definition of IoT comes from the convergence of multiple technologies [51], such as real-time analytics, embedded systems, etc. Devices send their data through the network without the need of human intervention. IoT applications facilitate our daily lives and improve industrial systems' performance.

Figure 10 depicts an example of smart metering. The appliances are connected through the Internet and send their data to the smart-meter control unit, which sends data to the server. With IoT devices, the user can implement the remote reading of energy consumption (water, gas, and electricity). This solution saves time and money by automating remote data collection. Also, analyzing these data makes it possible to identify problems or anomalies whose preventive treatment may significantly improve the operational processes. One way to facilitate data management is by using the cloud and using it to enable devices to exchange sensors' data. When the cloud receives the data, it might trigger specific actions, such as sending an alert or automatically adjusting some parameters, without user intervention. Typically, a user can control IoT devices by acting on a dedicated User Interface (UI).

2.3.2 Relevant IoT technology trends

In general, IoT systems are composed of *hardware*, *middleware*, and *platforms* [52]. Some examples of these components are reported in this section and depicted in Figure 11.

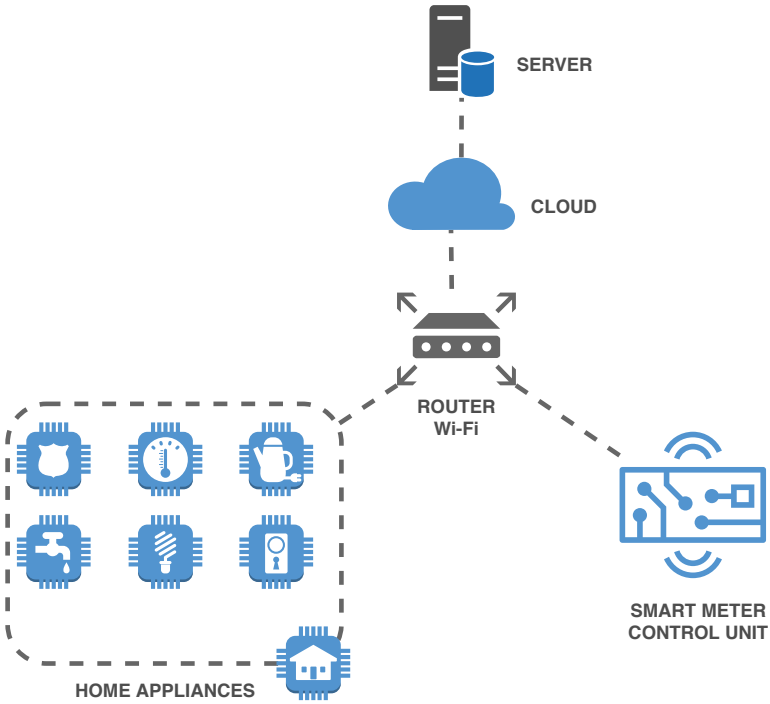

Figure 10. Example of smart metering.

Hardware

IoT projects employ various types of configurable and programmable hardware products. Among the most popular, we would like to mention the following ones, ranging from individual devices to full-blown platforms.

- *Particle*[1]*:* A fully-integrated IoT platform that offers everything to build an IoT device. This solution consists of the hardware, software, and connectivity needed to build a reliable, secure, and scalable IoT device. Besides, the Particle device cloud gives control over all the customer's devices.
- *Arduino*[2]*:* An open-source computer hardware and software platform that can be used for implementing embedded system projects. Its digital devices can sense physical movements and can be easily controlled to provide reliable and accessible communications between physical objects [53]. Also, the platform provides Arduino IoT Cloud, a powerful service allowing anyone to create IoT applications in a few simple steps [54].
- *Raspberry Pi*[3]*:* A standalone computer system. It is a very cheap and popular plat-

[1]https://www.particle.io/
[2]https://www.arduino.cc/
[3]https://www.raspberrypi.org/

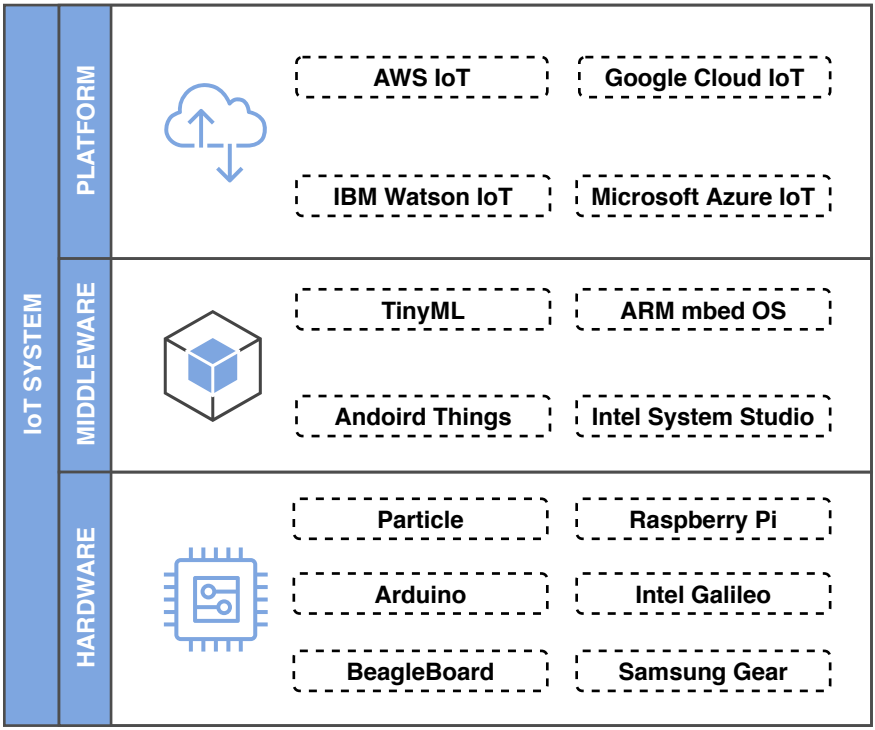

Figure 11. IoT system overview.

form that includes network interfaces such as Wi-Fi, Ethernet, and Bluetooth. Moreover, it supports different programming languages and has many input-output (I/O) interfaces. It is undoubtedly a flexible and more powerful solution for many IoT projects [55].

- *Intel Galileo*[4]*:* An Arduino-certified development board based on Intel $x86$ architecture. The primary goal of this platform is to broaden Arduino projects [56]. In addition, Intel Galileo enables complex IoT projects with dedicated board extensions, e.g., DevKit.
- *Samsung Gear*[5] A smartwatch with Tizen operating system. It is a wearable IoT device that collects users' personal digital information by assembling and processing the data related to the environment, daily routines, and communities and enabling smart digital services based on this pool of information [57]. This technology supports and implements IoT forensics, and thus, the possibility of using the data collected by these devices for forensics analysis [58].

[4]https://www.intel.com
[5]https://www.samsung.com/it/wearables/gear/:

- *BeagleBoard*[6]*:* A device based on a high-performance Texas Instruments Sitara processor, typically equipped with an open-source GNU/Linux distribution, an open hardware design, and with available add-on daughter boards. It is the right choice for IoT developers because it is supported by the Microsoft Azure platform.

All the devices listed above are used as bricks during the development of any IoT project.

Middleware

The middleware layer is used to play different roles that depend on the context in which it is used. Middleware generally provides services to lower and upper layers and is accountable for establishing connections between devices. A middleware connects applications, operating systems, and networks [52] and abstracts from the complexities of specific hardware, simplifying application development [59].

- *ARM mbed OS*[7]*:* A free, open-source embedded real-time operating system (RTOS) designed IoT devices. It is designed for an Arm microcontroller and includes many useful features for security and connectivity and drivers for sensors and I/O devices.
- *Android Things*[8]*:* An Android-based embedded operating system platform provided by Google that is also known as Brillo. It is aimed to be used with low-power and memory-constrained IoT devices. It supports Bluetooth Low Energy (BLE) and Wi-Fi. Along with Brillo, Google also introduced the Weave protocol, which is mainly used for communication with other compatible devices.
- *Intel System Studio*[9]*:* A cross-platform tool suite that simplifies application development for systems and IoT devices. It offers tools to build, analyze and debug the code to boost application performance and power efficiency and strengthen systems reliability.
- *TinyML*[10]*:* A *tiny* ML platform useful for a smart device that needs to include ML architectures, techniques, and tools for performing on-device analytics. This framework includes various sensing modalities (vision, audio, motion, environmental, human health monitoring, etc.) and can be used in IoT projects to add machine intelligence to devices.

These operating systems represent the bridge between devices and the Internet, where there are platforms to develop and control IoT devices.

Platforms

An IoT platform comprises several integrated services supporting data storage, processing, analytics, and visualization.

Several leading IoT platforms have been selected, which are commonly utilized by many IoT projects to leverage their features and implement innovative applications.

[6]https://beagleboard.org/bone
[7]https://os.mbed.com/mbed-os/
[8]https://developer.android.com/things
[9]https://beagleboard.org/bone
[10]https://www.tinyml.org/home/index.html

- *Amazon Web Service (AWS) IoT*[11]*:* A cloud service that connects IoT devices to other devices and AWS cloud services. AWS IoT provides software that supports the integration of devices into AWS IoT-based solutions. This platform provides an operating system, communication protocols, security services, and analytics tools [60].
- *Microsoft Azure IoT*[12]*:* A collection of managed services from perimeter devices to the cloud that permits connecting, monitoring, and controlling billions of IoT assets. It also includes security and operating systems for devices and equipment and data and analytics that help businesses create, deploy, and manage IoT applications. It provides a two-way connection between devices and platforms connected to the IoT, providing strong security mechanisms that ensure scalability and easy integration with the system [61].
- *Google Cloud IoT*[13]*:* A complete set of tools to connect, process, store, and analyze data both at the edge and in the cloud. The platform consists of scalable and fully managed cloud services. It includes an integrated software stack for edge computing with machine learning capabilities to support IoT projects [62].
- *IBM Waston IoT*[14]*:* A platform that supports the quick implementation of IoT projects. It is a completely managed, cloud-hosted service designed to get value from IoT devices. It provides features such as device recording, connectivity, control, quick view, and data storage.

2.3.3 IoT Architecture

As already mentioned, IoT enables the communications of billions of smart objects. Therefore, an IoT architecture has to support this tremendous undertaking. The literature reports many contributions to these aspects. Figure 12 depicts the three layers model. It is a variant of the open system interconnection (OSI) model. The variation goes toward simplifying the model to represent a smart device better. Thus the model adopted in this chapter considers *perception layer*, *network layer* and *application layer* [63], as follows:

- The perception layer is a physical device and communication layer that consists of sensors and actuators that sense, aggregate, and process data, then transfer the data to the network layer. This layer is classified into the perception node, e.g., sensors, controllers, physical objects, and actuators, and the perception network that interconnects this layer with the network layer. Data is acquired and controlled at the perception node, while control instructions for sending and managing data are carried out at the perception network layer. Perception layer technologies include all types of sensors, such as ZigBee, Radio Frequency Identification (RFID), sensor nodes, and sensor gateways.
- The network layer is a communication layer that transmits the aggregated data and storage awareness from the perception layer to the application layer using different

[11]https://aws.amazon.com/it/iot/
[12]https://azure.microsoft.com/en-us/overview/iot/
[13]https://cloud.google.com/solutions/iot/
[14]https://www.ibm.com/cloud/internet-of-things

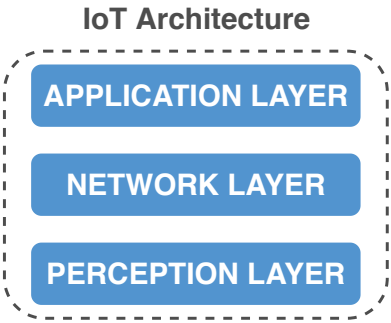

Figure 12. IoT three layers architecture model.

devices such as routers, gateways, and switches. The network layer includes mobile networks, cloud computing, and the Internet.

- The application layer is a visible messaging layer to interact with end-users. This layer comprises applications such as smart cities, smart grids, healthcare systems, and intelligent transportation protocols. An application layer protocol is distributed over multiple end systems, where the application in one end system uses a protocol to exchange information packets with an application in another end-system[64].

Alongside the growth of IoT, in recent years, we have seen the development of the standardization of IoT. Many organizations are involved in this important task, including the International Telecommunication Union (ITU), the European Telecommunication Standards Institute (ETSI), the 3rd Generation Partnership Project (3GPP), the World Wide Web Consortium (W3C), the Institute of Electrical and Electronics Engineers (IEEE), the Internet Engineering Task Force (IETF) and the Organization for the Advancement of Structured Information Standards (OASIS) [63].

As described in Figure 13 each IoT layer uses different protocols and standards. The perception layer and communication technology use WiFi, 4G/5G, LoRaWAN, IEEE 802.15.4, and others. The network uses Internet Protocol version 6 (IPv6), 6LoWPAN, Low-power, and Lossy Networks (LLNs), mDNS, TLS, and DTLS. The application layer typically includes MQTT, Constrained Application Protocol (CoAP), HyperText Transfer Protocol (HTTP), and eXtensible Markup Language (XML). The rest of the section describes standards and protocols used by IoT devices grouped for each layer.

2.3.4 IoT Standard and Protocols

Various standards have been introduced to assess the services and relevance that are utilized for IoT solutions to link several things to the Internet in IoT common standards [65] [66]. Although multiple protocols have been developed, they are not all required for a single IoT application simultaneously. The IoT protocols for a given application are chosen considering the nature of the application [45]. The most common IoT protocols (depicted in Figure 13), which are utilized in a variety of applications, are listed below:

A. **Message Queue Telemetry Transport (MQTT)**

MQTT is a messaging transport protocol that performs data aggregation of the environmental data and sends it to a web server [67]. This protocol is based on the TCP subscribe and publish messaging model and is intended for lightweight M2M, server-to-server, and machine-to-server interactions [68, 45]. Here, the clients act as publishers-subscribers, and the server should act as a broker where the clients are connected to the server through TCP. Generally, the subscriber registers for a particular task in a device, and the data are generated and transferred to subscribers by the publishers through brokers [69]. MQTT is appropriate for utilization in things with limited resources, like those with low power and computing capabilities connected to low bandwidth or unstable networks. However, the MQTT protocol is not suited for usage with all IoT applications because it operates over TCP, and the overhead is raised because it uses topic names as texts [70, 45].

B. **Constrained Application Protocol**

The IETF, Constrained RESTful Environments (CoRE) research team developed the CoAP,

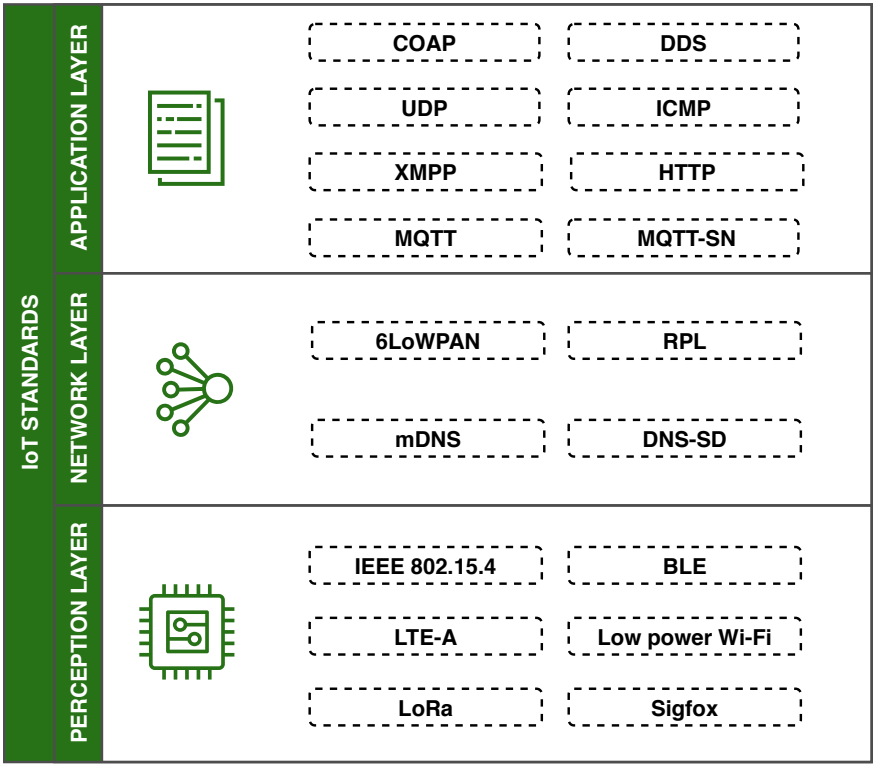

Figure 13. IoT Standard and Protocols.

which is a HTTP is functional and lightweight application layer protocol [71]. As most IoT devices have limited power and storage, the CoAP protocol extends the functionalities of HTTP (which has a relatively high complexity) by fulfilling the needs of IoT devices [72]. This protocol shows how to build a web transfer protocol called REpresentational State Transfer (REST) on the upper level of HTTP. The CoAP uses the UDP because it is simple in nature and has a small message size and layout, which helps to decrease the need for bandwidth, utilize resources, and decrease the overhead of TCP handshaking before data transfer [73]. This protocol has two sub-layers: the messaging sub-layer and the request/response sub-layer. The first sub-layer (messaging) determines the replications and ensures efficient data transmission over the UDP through exponential backoff as UDP is constrained by error recovery technique. On the contrary, the REST communications are handled by the request/response sub-layer. There are four kinds of messages in CoAP: confirmable, non-confirmable, reset, and acknowledgement. The CoAP enables efficient delivery, congestion control, and flow control for IoT applications in resource-constrained and unsynchronized objects [74]. The CoAP has several drawbacks, including increased communication delay, packet delivery instability, and the inability to transfer complicated data [75].

C. **Extensible Messaging and Presence Protocol (XMPP)**

eXtensible Messaging and Presence Protocol (XMPP) is a protocol that ensures low bandwidth communication and short message transfer, making it ideal for video conferencing, publish-subscribe systems, telepresence, multi-party chatting, and talking in IoT [76]. For instant messaging applications, XMPP is appropriate for authentication, security measures, access control, hop-by-hop and end-to-end encryption, and interoperability with various protocols. This protocol serves three functions: client, server, and gateway, and it facilitates two-way communications between any two of these roles [77]. In this scenario, the client connects to the server through a TCP protocol and transfers data using the XML streaming standard; the server is in charge of the connection management and routing of the message, and the gateway ensures reliable connectivity among distributed systems. This protocol allows communication among a variety of applications as it is flexible and simple in nature. However, XMPP requires high computing capabilities devices, consumes bandwidth of the network, transfers simple types of data, and cannot provide QoS [78].

D. **Advanced Message Queuing Protocol (AMQP)**

Advanced Message Queuing Protocol (AMQP) is an open platform messaging standard that is utilized at the application level to provide message services such as privacy, queuing, durability, and routing [79]. AMQP ensures reliable and consistent information exchange using message-passing primitives such as one-to-one, one-to-many, and exactly-once delivery. This protocol needs a stable transport protocol architecture, and middleware serves as a gateway between applications and available resources, connecting institutions and mechanisms throughout time and space. The message and exchange queues are the two main steps in the AMQP data transmission process. In a message queue paradigm, messages are kept until they are delivered to the recipient. The messages are transmitted in an appropriate sequence in another scenario (exchange queue model) [80]. AMQP also enables the publish/subscribe communication architecture and point-to-point data transfer. There are two kinds of messages found in AMQP: the bare

messages provided by the sender and annotated messages available to the recipient. However, AMQP requires comparatively higher bandwidth and does not guarantee resource discovery [81].

E. **Bluetooth Low Energy (BLE)**

BLE, an extended version of Bluetooth offers a small radio with reduced power consumption to operate for a longer period for controlling and monitoring applications [82]. The protocol stack utilized in BLE is almost identical to the standard of conventional Bluetooth technology, but it has a larger coverage, approximately 100 meters, with low latency [79]. The devices that employ the BLE standard are categorized into master and slave. The master devices are the ones that play the most important roles and link to slaves. Additionally, the slaves can access and subscribe to several master devices. BLE enables devices to investigate as masters or slave channels in star topology [83]. This technology turns off the radio while in idle time and only turns it on to broadcast or receive minimal data packets, resulting in minimal energy consumption. A gateway (another BLE device with network connectivity) is required for BLE devices while transferring data over the Internet.

F. **Zigbee**

Zigbee is a communication standard that ensures reliable, low power, and cost-effective data transfer, but it covers a small range of communication [84]. Zigbee supports star, cluster-tree, and P2P network topologies. A controller, in general, is in charge of the structure and can be found in the middle of a star network, at the root of a tree or cluster architecture, or anywhere else in a P2P topology [85]. There are two stacks in Zigbee standard, ZigBee and ZigBee Pro; these stacks allow mesh network architectures to operate with various applications, allowing for low storage and processing power implementations.

G. **Z-Wave**

Z-Wave was initially developed by ZenSys (presently Sigma Designs) but was then improved by Z-Wave Alliance [6]. Z-wave was applied as a wireless network protocol; different from ZigBee, Z-wave defines all protocol layers and supports communication, networking, and application layer protocols [86]. Z-wave can support mesh networking, broadcasting, and multi-casting [86]. This technology operates in ISM band frequency (900 MHz) and allows a transmission rate of 0.04 Mbps. The recent version of Z-Wave technology can support up to 0.200 Mbps and covers about 30 m point-to-point communication [87]. Thus, it is applied for specific applications which require tiny and miniature data transmission, such as home automation services [88] [89], Home Automation Networks, health care control, and smart energy [86]. It has a specific architecture based on a controller and slave nodes. Controllers are able to manage the slaves using sending commands. Indeed, in wireless technology, routing protocols should ensure reliable packet transfer and maintain connectivity between different nodes [90]. These protocols are always performed through a source routing algorithm. The routing method requires that the controller keeps a table of the whole network topology. This controller is asked to submit the path inside the packet [90].

H. **Low-Power Wireless Personal Area Networks (LoWPAN**

LoWPAN are made up of a variety of cost-effective devices that are linked via wireless communication. This protocol has many applications in IoT architectures due to its

small packet sizes, low computing power, low data throughput, and low latency [91, 45]. Additionally, the 6LoWPAN protocol was introduced by incorporating the most recent release of the IPv6 and LoWPAN. 6LoWPAN makes it easier to maintain the administrative process by allowing each constrained object to be accessed independently within the network. Additionally, it is in charge of segmenting and reorganizing IPv6 traffic, guaranteeing unitary routing, reducing protocol stack headers, and providing compliance with the higher levels [92]. This protocol eliminates overall packet overhead as it does not include the extra header information during routing. Besides, 6LoWPAN contains a mesh address header to enable packet routing in a mesh architecture, but it is unable to provide detailed information on routing to the link layer. 6LoWPAN has various advantages, including ad-hoc self-organization, robust connectivity, standard compatibility, and low power consumption [45].

Finally, the Data Distribution Service (DDS) provides low-latency data connectivity, extreme reliability, and scalable architecture for mission-critical IoT applications [93]. IoT devices sense and act upon physical environments, and the *network layer* needs a standard Domain Name System (DNS) to catalogue and discover resources efficiently. Multicast DNS (mDNS) [94] and DNS Service Discovery (DNS-SD) [95] are widely used today to discover resources and services offered by IoT devices. IoT devices are uniquely identified by a network address called IPv6. The IETF developed the IPv6 Low-power Wireless Personal Area Networks (6LoWPAN), which is used as an adaptation layer that allows sensor nodes to implement the IP [96]. This protocol is an adaptation layer allowing the transfer of IPv6 packets over IEEE 802.15.4 networks and overcomes the small size of the Maximum Transmission Unit (MTU), which is 127 bytes. Furthermore, IETF designed the routing protocol for LLNs as a link-independent distance-vector routing protocol, which is based on IPv6 for resource-constrained nodes [62]. The RPL is a tree-based routing protocol in which nodes build a Destination Oriented Directed Acyclic Graph (DODAG) by exchanging distance vectors and roots with a controller. It can be considered as the standard routing protocol for IoT [97, 98]. There are several other kinds of networks involved in IoT communications, namely *mobile communications:* cellular networks (3G, 4G, 5G, 6G), WLAN, Wireless Personal Area Networks (WPAN), and *world-wide communications* [63].

In the *perception layer*, one of the main standards that support low power and lossy networks (LLNs) is the IEEE 802.15.4 standard, which forms the backbone of WSNs as part of the IoT. This standard defines the physical and data-link layers of the network and provides a framework of operation at low costs [99]. However, physical devices and communication evolved to the Low-Power-Wide-Area-Network (LPWAN) and the Low-Rate Wireless Personal Area Network (LR-WPAN) [100]. LPWAN and LR-WPAN have been developed to help small sensor communications. These last two protocols have been designed to meet typical requirements of WSN, namely, burst and low-power communications for long-range links.

LR-WPAN described two low-level layers which have to be controlled, e.g., the physical layer and the MAC layer. On the other hand, LPWAN is used for long-range communication devices and sensors. It supports low-power and low-bit-rate communications. Long-range (LoRa[15]) defines the physical layer of an LPWAN protocol, and it uses a proprietary Chirp Spread Spectrum (CSS) modulation. In this case, the other higher layer protocols are defined

[15]https://lora-alliance.org/

by LoRaWAN, which is a network protocol that manages the routing, the data rate, and the frequency selection as well. Sigfox[16] is an LPWAN network operator which introduces a solution of end-to-end IoT connectivity based on its patented technologies. BLE is a de-facto key wireless technology for WSNs and wearable smart devices. This protocol, released by Bluetooth Special Interest Group (SIG)[17], offers a wider range, lower latency, and minimal power than Bluetooth. Long-Term Evolution Advanced (LTE-A)[18] provides sufficient scalability and flexibility to adapt to the M2M communications and IoT applications in cellular networks [101]. Wi-Fi is the most ubiquitous wireless Internet connectivity technology today. Unfortunately, it was developed many years before the IoT era, and its energy consumption represents the major barrier to its use in small sensors. Low-power Wi-Fi solves this problem by integrating Wi-Fi into emerging IoT applications and battery-operated devices [102].

2.3.5 IoT Applications

Due to the self-sufficiency of each IoT component, its role can be defined in various semi-automated or fully automated environments. The early adopters of IoT, such as the transportation, healthcare, and automotive industries, have witnessed significant positive impacts from its implementation. However, to provide a structured classification, this work adopts the three-layered IoT application taxonomy presented by [103]. This taxonomy categorizes IoT applications into different classes, including Service-oriented Applications (SoA), RFID-oriented applications, WSN, Supply Chain Management (SCM), healthcare (e.g., m-health and e-health), cloud-based services, smart society, and social computing. Each of these classes primarily corresponds to specific application domains, which may involve one or more layers of the IoT architecture [33].

<table>
<tr><td>Application Protocols</td><td colspan="2">CoAP</td><td colspan="2">MQTT</td><td>MQTT SN</td><td>XMPP</td><td>AMQP</td><td>DDS</td><td>HTTP</td></tr>
<tr><td>Network and Service</td><td colspan="6">mDNS</td><td colspan="3">DNS-SD</td></tr>
<tr><td rowspan="4">IoT sensing Network</td><td colspan="2">Routing Protocol</td><td colspan="3">RPL</td><td>AODV</td><td colspan="3">OAODV</td></tr>
<tr><td colspan="2">Network Layer</td><td colspan="7">6LoWPAN</td></tr>
<tr><td colspan="2">Application Layer</td><td colspan="7">IEEE 802.15.4</td></tr>
<tr><td colspan="2">Perception Layer</td><td>LTE-A</td><td>WiFi</td><td>NFC</td><td>RFID</td><td>IEEE 802. 15.4</td><td>Z-wave</td></tr>
</table>

Figure 14. General Standardization of IoT Application.

This section focuses on describing the domain applications relevant to this chapter. A service-oriented application refers to a versatile architectural approach employed for automating business processes by integrating various components within each environment

[16]https://www.sigfox.com/en
[17]https://www.bluetooth.com/
[18]https://www.3gpp.org/technologies/keywords-acronyms/97-lte-advanced

across the enterprise network. However, as the IoT service applications expand, the environment may involve diverse third-party components [33]. The IoT service architecture must possess robust features to ensure effective integration, capability, and interoperability in such heterogeneous environments. This enables the components to efficiently address business requirements through rapid development or by leveraging existing resources through reuse and integration [104, 105]. Simultaneously, RFID technology is widely recognized for its ability to connect objects through RFID tags. This technology relies on the perception layer for environmental sensing and the network layer for communication handling. It is essential to validate the security mechanisms in both these layers to ensure safe and secure functionalities [106]. WSN plays a pivotal role as a fundamental component in the IoT ecosystem and can be considered as the precursor to IoT [107]. WSN consists of sensor nodes (motes) managed by a coordinator (sink) and utilizes a diverse range of protocols organized in a multi-layered structure to facilitate communication and data transmission [108] [109].

IoT plays a crucial role in supply chain management processes across various application domains. This enables automated data transfer from suppliers to consumers, streamlining the entire supply chain [110]. IoT applications in the healthcare system have witnessed significant advancements in recent years. Wearable devices, such as sensors, are extensively utilized in IoT-based healthcare systems, collecting sensitive data. However, these devices are susceptible to adversarial environments, making it crucial to implement robust security mechanisms to prevent malicious interactions with the coordinator system [111]. Cloud-based services serve as the backend infrastructure for IoT, categorized into stationary and mobile applications. Given the resource constraints and limited communication coverage and mobility of IoT devices, there is a high demand for efficient packet forwarding in cloud-based services within IoT environments. Social computing in IoT resembles a social network structure, where each node represents an interconnected device that establishes social relationships with other connected devices. This process operates fully automated under predefined rules, monitored by a coordinator [33].

The concept of a smart society revolves around leveraging accurate city information at the right time and in the appropriate context to make optimal decisions for various events. To establish an IoT-based smart city, a multitude of wired and wireless sensors are being deployed. The main challenge in building smart societies lies in integrating and linking the vast amounts of city data generated by diverse smart systems and sensors into a centralized platform. Furthermore, ensuring the privacy and security of the collected data poses a significant challenge due to the potential risks associated with handling large volumes of sensitive information [112]. It is equally important to secure smart devices, as adversaries often exploit resource constraint vulnerabilities to target these devices [33].

2.4 Requirements and Challenges in IoT

2.4.1 IoT Security Challenges

We live in a hyper-connected society, and massive numbers of smart devices surround us. In this scenario, IoT security plays a crucial role. In this section, an analysis of security requirements, attacks, and vulnerabilities associated with this technology is conducted.

A contribution by Jurcut *et al.* [113] reviews different market-available solutions that deal

with IoT security. In this work, the author addresses the complexity of securing and monitoring the environment of IoT smart devices by presenting different detection mechanisms that consider the resource constraints of the IoT devices. Another research paper by Ali *et al.* [114] reviews different lightweight algorithms used to protect IoE devices from attackers and presents some authentication methods for enhancing the authentication of IoT. Alaba *et al.* [115] categorizes the issues of IoT security in terms of communication, data, and architecture; also, they discuss IoT hardware and network threats. Another survey by [116] discussed the privacy, confidentiality, access control, and data security of IoT. Their investigation includes home appliances connected by WSN, Mobile Edge Computing (MEC), fog computing, cloud computing, access control, and trust management mechanisms. Zhou *et al.* [117] state the security and privacy requirements for the next generation of cloud-based IoT systems. Zhang *et al.* [118] investigate the main security issues of IoT by focusing on authentication, authorization, privacy, and the need for cryptographic tools to mitigate software vulnerabilities and malware attacks. Other researchers discuss encrypting the communications between clients and servers to guarantee privacy, confidentiality, and authenticity [119]. Moreover, the problems are addressed at various layers within the IoT stack to ensure proper data usage, secure communications, and applications.

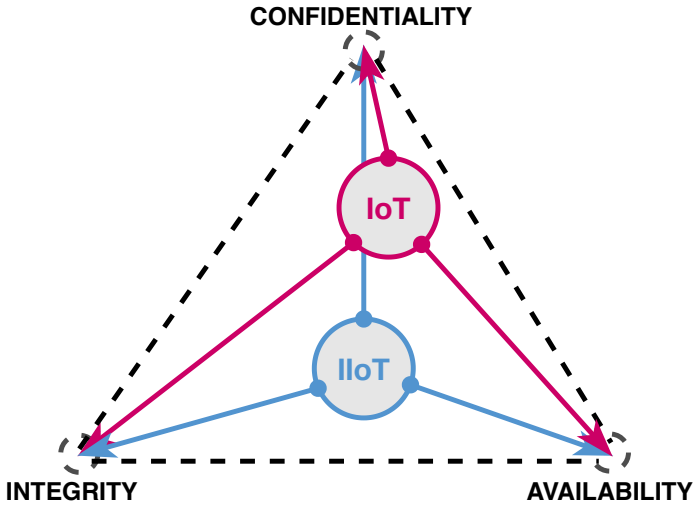

Figure 15. IoT CIA security model.

I. Security requirements

IoT applications can be classified as *consumer IoT*, named IoT, and *Industrial Internet of Things (IIoT)* [120]. Depending on this division, the author will consider IoT security challenges. The Confidentiality, Integrity, and Availability (CIA) triad (see Figure 15) is a well-known model that defines the security requirements and supports organizations to define the core security objectives of their systems [121]. Satisfaction of CIA properties ensures data

security of IoT and IIoT devices. Moreover, the IoT inherits all security requirements as a network. However, it also has numerous constraints and limitations regarding resources and devices, including computational and power resources, which pose additional challenges.

A. **Classical Security Requirements**

Confidentiality maintains privacy and protects proprietary information between the two devices involved in communication by implementing mathematical algorithms to transform data into a form that is not readily intelligible. This property is usually guaranteed through specific mechanisms for data encryption or access control [121].

Integrity refers to the protection of useful information from the attacker or external interference during data transit or rest through some common methods like data integrity algorithms that prevent data alteration. Two categories of integrity properties can be identified: i) connection-oriented integrity that guarantees that messages are received as sent without duplication, insertion, modification, reordering, or replays, and ii) connectionless integrity that protects messages' modifications and improper information destruction [121].

Availability ensures timely and reliable access to resources by authorized parties. This property is guaranteed through hardware redundancy, firewall, and software maintenance [121].

IoT and IIoT emphasize these requirements differently (see Figure 15). IoT security is more concerned with confidentiality to avoid stealing private information. IIoT security is more concerned with data integrity to avoid unplanned system outages. Besides the above deliberations, other security requirements deal with Authentication, Access control, and Accountability (AAA) [121]. The priority of these requirements in IoT and IIoT depends on the task performed by the device.

Authentication is concerned with assuring the authenticity of the communication. At the time of connection initialization, the service assures that the two communicating entities are those they claim to be. During the communication, it assures that the exchange of messages is not interfered with by a third-party device that can masquerade as one of the two entities [121].

Access control specifies access rights or privileges to resources in order to enable different users to access the required resources [121]

Accountability ensures that devices or individuals are responsible for their actions in case of theft or an abnormal event. This security goal calls for uniquely tracing actions of a specific device [122].

B. **Specific Security Requirement**

Information security encompasses all strategies to preserve, restore, and ensure information security in computer systems against attack. IoT inherits all security requirements as a network, but it also has numerous constraints and limitations regarding resources and devices, including computational and power resources, which define additional challenges. Therefore, *resources efficiency* could be considered one of the security requirements used to ensure that the adversary will not carry out attacks on the IoT architecture, leading to increased resource usage due to duplicated or faked service requests [123].

II. IoT security attacks

The three levels of IoT architecture depicted in Section 2.3.3 are all prone to security attacks. In this section, security attacks are identified and classified according to these three levels.

A. **Perception layer security**

Very often, the devices used in IoT are low-power and memory-constrained IoT devices. It is thus challenging to implement cryptography protocols and other authentication mechanisms needed to protect them. It is also essential to distinguish between sensors and actuators. The first sense information and make it available remotely, whereas the latter are devices that can be controlled remotely [124] and perform specific actions. Some devices falling into this category and commonly used at the perception layer are listed below:

- *perception nodes:* RFID nodes and tags;
- *sensor nodes:* wireless sensors composed of Radio Frequency (RF) transceiver, Micro-Controller Unit (MCU), memory, and a power source;
- *actuator gateways:* smart-sensors that check and record temperature, electricity, humidity, pressure, speed, and other magnitudes.

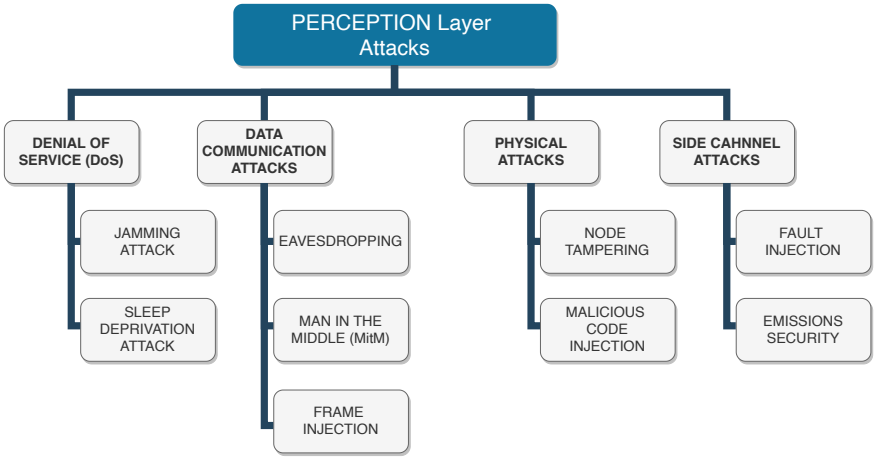

Figure 16. Perception layer security attacks.

These devices could be subjected to many security attacks, as shown in Figure 16, described below.

(a) *Denial of Services (DoS)*: the adversary degrades the node from correct service delivery by exploiting its features. Examples are *jamming attacks* (intentional interferences that disrupt wireless communications), and *sleep deprivation attacks* (adversaries drain the energy of the IoT devices by forcing sensor nodes to stay awake and drain the system battery with useless tasks [125].

(b) *Data communication attacks*: the adversary attack data during the transmission.

(i) *Confidentiality attack*: unauthorized interception of private information. This attack invades privacy via, e.g., *eavesdropping* (capturing and decoding data sent over the wireless link to obtain sensitive information) or a *Man-In-The-Middle attack (MITM)* (the attacker intercepts the communication between the victims and injects new messages while each endpoint is not able to detect the intruder) [121].

(b) *Integrity attack*: modification of data in transit over a wireless network in order to mislead the receiver or facilitate another attack, via, e.g., the *frame injection* (the attacker injects frames into the wireless communication in order to display malicious content on a legitimate channel) [121].

(c) *Physical attacks*: the devices are attacked by unauthorized access to the hardware components. This approach is effective if the attacker has the target or a copy of the target. Examples are *node tampering* (an unauthorized modification of the device in which the adversary hacks the device by physically replacing some hardware components to get access to private information), and *malicious code injection* (the attacker injects a malicious code into the IoT device to get control of it) [126].

(d) *Side channel attacks*: a technique in which the information is leaked through a channel other than deliberately engineered for communication. These attacks require the attacker to be close enough to the target and, e.g., use *fault injection* (the attacker leaks information about the target by monitoring the effects at the lower layer of a fault such as a clock glitching, voltage glitching, overclocking, and electromagnetic injection) or *Emission Security* (EmSec) (the opponent exploits electromagnetic leakage intercepting information carried by electromagnetic emanations) [121].

Perception layer attacks can cause serious effects, leading to different issues. In this context, once for accidentally revealing user data to Facebook and Google via third-party trackers embedded into their Android applications, and once due to an IoT security breach, cybercriminals successfully hacked into several families' connected doorbells and home monitoring systems. The attacker could access live feeds from the cameras around consumers' homes and communicate remotely using the devices' integrated microphones and speakers. Over 35 people in 15 families reported that attackers were verbally harassing them [127].

B. **Network layer security**

The network layer must ensure that the data sent over communication channels (the air) cannot be altered. The main purpose of the network layer is to transmit the gathered information received from the perception layer to the application layer through existing communication networks. Network-level security analysis focuses on the threats that may occur to IoT communications, as shown in Figure 17.

(a) *Routing attacks*: these attacks modify the routing path during communication. In a *sinkhole attack*, attackers force their nodes to respond to the routing requests. Thus, they can use the packet node for malicious activity on the network [128]. The *wormhole attack* causes a breakdown to the operation of the 6LoWPAN by creating a tunnel between two nodes, and a packet is routed to a malicious node instead of the original destination [129].

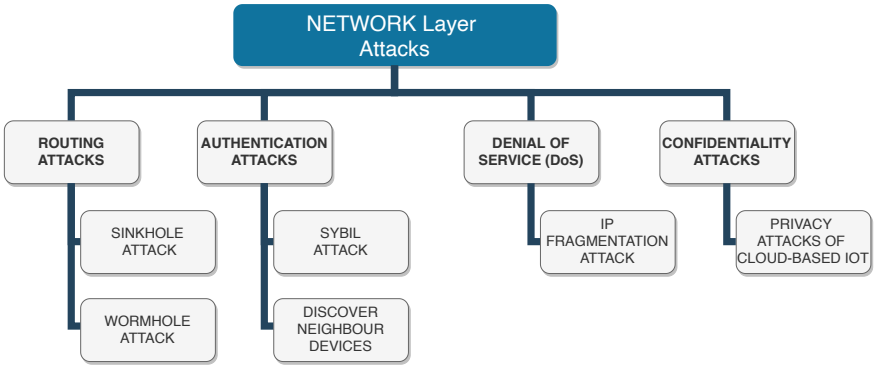

Figure 17. Network level security attacks.

(b) *Authentication attack*: IoT devices must be authenticated using critical management systems. Any mistake on the system network may expose the system to different types of vulnerabilities. The *Sybil attack* uses a malicious node to forge false identities that act as multiple distinct nodes. Moreover, the adversary may thus be able to control devices over the network, influence the network operations, and disrupt middleware services [130]. The *discovery of neighbouring devices* protocol common in IoT can be used for MITM attacks.

(c) *DoS*: the adversary degrades the network performance. An *IP fragmentation attack* may cause resource depletion and a buffer overflow. Duplicating the fragments sent by malicious resources will affect the packet re-assembly and block the processing of other legitimate packets [131].

(d) *Confidentiality attack*: unauthorized interception of sensitive information. This attack invades privacy but leaves confidential data intact. *Privacy attacks of cloud-based IoT* are different types of attacks that violate the identity or the location privacy and might affect the cloud-based IoT platforms; consequently, the data on the IoT cloud can be accessed by malicious attackers [132].

Network attacks can cause serious effects, leading to a complete shutdown. The US electricity grid was attacked in late 2017. The attacker gained remote access to energy sector networks. Then, they conducted network reconnaissance, moved laterally, and collected information about industrial control systems. The attackers successfully switched off the power for a couple of hours [133]. A DDoS attack hit a DNS service provider in October 2016 [134]. The attack lasted for a few hours, affecting the services of Twitter, GitHub, and other services.

C. **Application layer security**

As previously discussed, there are countless applications that implement industrial automation or smart home functionalities. Possible applications for IoT and IIoT are:

- *IIoT*: industry 4.0, smart-grid;
- *wearables*: smart watch, e-health;

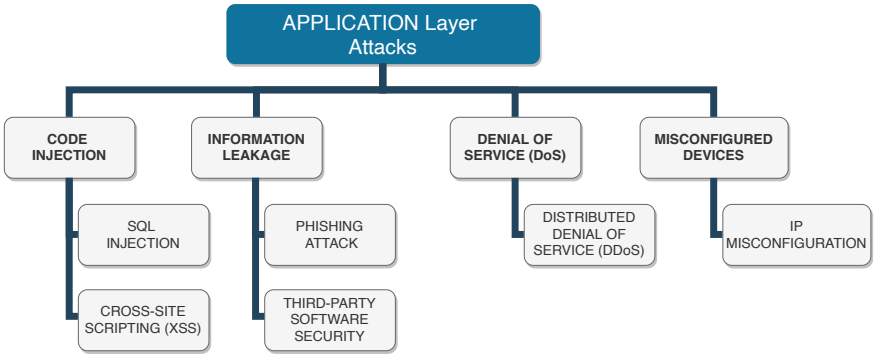

Figure 18. Application level security attacks.

- *transportation*: intelligent transportation systems, Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) systems;
- *smart-city*: smart-building and smart-environment.

Security issues for this level mainly focus on attacking IoT applications through the interface or the languages used to facilitate IoT projects. This section reviews the most important application-level security attacks that are depicted in Figure 18 and are also possible for IoT systems.

(a) *Code injection*: the *software security* in which various vulnerabilities in IoT can come from codes and languages used to write a specific code (e.g., SQL, XML, JSON, etc.). The programmer needs to be more careful about testing the code before publishing it. *Web security* deals with IoT applications that can be represented through the web. This interface is vulnerable to different types of attacks such as *SQL injection* and *Cross-Site Scripting (XSS) [135]*.

(b) *Information leakage*: The attacker can steal data by knowing the vulnerabilities of the service or application and the threats of the IoT security that can cause information leakage. Since IoT devices are deeply connected to our lives and industrial applications, data leakage might lead to social damage or economic losses [136]. The authenticity of *third-party applications* cannot be checked, and when the end-user installs this software, it might corrupt the IoT system [137].

(c) *DoS attack*: A hacker makes several attempts as an authenticated user and logs into the system, interrupting the normal working of the network [126]. A *DDoS* attack disrupts the normal traffic of a targeted service by flooding the victim or its surrounding infrastructure with a flood of Internet traffic.

(d) *Misconfiguration*: IoT devices and their applications should be configured appropriately. Indeed, an attacker might exploit a misconfiguration in the application, operating system, DB, and protocols to implement an attack that might damage the whole IoT network. For instance, an *IP misconfiguration* may put at risk the performance of the whole system [138].

In addition to exploiting weaknesses of IoT applications, attackers can also use Open-Source Intelligence (OSINT) tools to gather information on IoT targets. The IP device search engines such as *Shodan*[19] and *Censys*[20] build searchable databases storing the information of the devices connected on the Internet and provide access to query for any user [139].

Shodan is a tool that lets anyone search for IoT devices online. Like Google, it pings every webpage to create an extensive list of them. Shodan records any metadata, which is then publicly broadcast. These metadata are named *banner*. For instance, the banner is the header for HTTP, while for FTP, it is the welcome string. The banner is the fundamental unit of data for these IP search engines. Shodan supports different types of filters to make its search engine more accurate and easy to use. These filters include country, city, device type, port, and product. It works by relying on servers around the world that crawl the Internet to access devices. It also generates a random Internet Protocol version 4 (IPv4) address and a random port before invoking such a service to obtain the banner. Moreover, *Shodan* collects information about the devices, such as the location, hostname, and operating system. This collected information can be seen using the Shodan API or Shodan website [139].

Censys is newer than Shodan, and it is a search engine that allows users to query the devices and networks on the Internet. Unlike Shodan, which captures the data in banners, Censys is built upon the Zmap[21] that is a faster alternative to Nmap and can scan the entire Internet address space in a short time. That makes it possible to have an almost real-time update on every IP address and consequently of the IoT's banners [139].

The application layer attacks could also cause severe effects on the IoT system. In March 2018, Cambridge Analytica successfully gained access to confidential information on more than 50 million Facebook users [140]. Also, in July 2015, a team of researchers was able to hijack the vehicle over the Sprint cellular network by exploiting a firmware update vulnerability. They discovered that they could control the vehicle's speed and turn it off from the road [127] by attacking the application layer of that vehicle.

2.4.2 IoT Security Mitigations

IoT projects are exposed to various vulnerabilities of different IoT components such as applications, interfaces, networks, communication, and firmware protocols. However, a good strategy to secure IoT systems is to define security mitigations for the different layers and understand how these solutions can improve the IoT communication chain. Undoubtedly, cooperation between layers can make overall IoT security more robust.

I. Perception layer

The IoT devices and sensors belonging to the perception layer are typically used in low-power and lossy networks, where memory, energy, and processing power are constricted

[19] https://www.shodan.io/
[20] https://censys.io/
[21] https://zmap.io/

compared to the localization of network nodes in conventional Internet platforms. Consequently, employing authentication schemes based on public-key encryption may prove impractical due to their high computational requirements and storage demands. Consequently, incorporating a lightweight cryptographic protocol becomes challenging when considering factors such as context awareness, ease of deployment, and scalability. It is important to consider various types of attacks that can occur in the perception layer. In the following lines, we will delve into the problems associated with these attacks and discuss mitigation strategies for each type [109]:

Jamming attack is a form of DoS attack where the attacker floods the network with a high range of malicious signals to disturb the communication and to deplete their resources such as bandwidth, battery life, and storage. Generally, the jamming attack can occur in a wireless medium through different situations such as collisions and interference, noises. However, attackers use jamming attacks in WSNs to interfere with the physical transmission of signals during the communication process. In WSNs, the perception and MAC layers are prone to jamming attacks. Recent research proposed methods for detecting jamming attacks in WSNs. These methods use prior information about communication behaviours during jammed and normal conditions, which can be tracked by using indicators and metrics obtained from different layers. Young *et al.* [141] proposed a solution that relies on measuring the signal strength to determine noises by comparing the observed values with statistically significant correct ones. Mistra *et al.* [142] proposed a centralized approach, a fuzzy inference-based system, to detect the jamming attack by using three inputs received from the sensor node in the network. These inputs are the total packets received during a specific period and the received signal strength (RSS). These values are used to differentiate between the current RSS and the normal RSS. After that, these values are used by the base station to compute the packet drop per terminal (PDPT) and signal-to-noise ratio (SNR), which are further used as inputs for the fuzzy inference system to obtain the jamming index. The jamming index varies from 0 to 100 and is used to determine the intensity of the jamming attack, which can range from a situation of no-jamming to absolute jamming.

Sleep deprivation attack causes energy consumption and reduces the lifetime of devices and sensors. For detecting or minimizing the effects of this attack, Pirretti *et al.* [143] investigated three different methods: the random vote scheme, the round-robin scheme, and the hash-based scheme. The random vote scheme counteracts the selection of a malicious node by randomizing cluster head selection. In the round-robin technique, each node benefits from becoming a cluster head at least once. This method is more scalable than the random vote scheme. Unfortunately, the round-robin scheme introduced an important overhead. However, researchers introduced the hash-based scheme to overcome this problem. In this method, each node generates a random number and then broadcasts its number's hash within the cluster. This information is then used for the cluster head selection. Tapalina *et al.* [125] proposed a hierarchical framework based on distributed collaborative mechanisms to detect the sensor nodes that are affected by the sleep deprivation attack in WSNs. They use a cluster-based mechanism in an energy-efficient manner. The proposed model uses an anomaly detection technique to avoid false intrusion. To mitigate the attack, the proposed model physically excludes malicious nodes from the network and rejects fake packets. Bhattasali *et al.* [144] proposed a framework for mitigating sleep deprivation attacks without using MAC-based protocols. The framework reduces energy consumption by limiting long-distance communications and relies on a cluster-based approach where the cluster is divided

into several sectors. The framework relies on a five-layer model of WSNs.

By *eavesdropping*, an adversary can acquire data and detect sensitive information of a user. Many research efforts have been made to develop techniques that detect intruders inside a WNS. Indeed, these systems prevent the intruder from causing damage to the network or stealing data [145]. Wang *et al.* [146] proposed an Intrusion Detection System (IDS) for WNSs that derive the detection probability by considering the sensing range, the transmission range, and node density. On the other hand, Silva *et al.* [147] explained that by using IDS, we could acquire information related to the attack techniques, helping in the development of prevention systems in distributed WNSs. Recently, researchers introduced new IDS based on ML techniques. Anthi *et al.* [148] proposed a three-layer IDS that used a supervised approach. Indeed, it classifies the IoT device's normal behaviour, identifies the malicious packets, and, finally, classifies the attack. Besides, in the literature, many contributions propose physical layer security solutions to avoid eavesdropping. These physical layer techniques aim to hide the mere existence of a node or the fact that communication is taking place. Security at the physical layer was mainly intended as the use of a spread spectrum technique [149]. Moreover, Soderi *et al.* [150] proposed to combine watermarking with a jamming receiver to draw a secure region around the legitimate receiver.

MITM is a kind of attack where an adversary could listen to the communication between two endpoints or more. The MITM attacker can modify, intercept, change, or replace the communication traffic of the target victim. This type of attack could be executed in different communication channels such as GSM, UMTS, Long-Term Evolution (LTE), Bluetooth, near-field communication (NFC), and Wi-Fi [151]. Cryptography techniques could be applied to the data to avoid this type of attack. Mahalle *et al.* [152] presented a solution designed to safeguard the system against DoS, replay attacks, and MITM attacks. The proposed protocol, known as Identity Authentication and Capability-Based Access Control (IACAC), utilizes the Diffie-Hellman algorithm based on Elliptic Curve Cryptography (ECC) to generate keys. This approach enables devices to authenticate their peers by relying on encrypted secret keys.

Frame injection is a kind of code injection attack classified by OWASP Top 10 [153]. This type of attack has different aspects, which allow hackers to redirect users to other malicious websites used for phishing and similar attacks. Zhiping *et al.* [154] finds that channel state information (CSI) could solve the issue of this attack.

With the *node tampering*, the attacker could damage the sensor node by replacing the entire code or part of its hardware or even electronically interrogating the nodes to gain access and alter sensitive information. For example, routing tables or shared cryptographic keys impact the operation of higher communication layers. This type of attack sends physical alerts to compromised nodes to obtain sensitive data, such as the encryption key. Tiberti *et al.* [155] proposed a defence from these attacks relying on Blockchain-based and computing cryptographically secure hashes for checking the content. Raymond *et al.* [156] introduced a framework in which replay protection and strong link-layer authentication could also mitigate node tampering attacks.

In *malicious code injection*, an attacker could damage the IoT system by modifying the information and sending the wrong data to other nodes, dropping the packets, and stealing the private data and encryption key. We can use anti-virus, firewalls, worm detectors, and IDS, and keep the system up-to-date to mitigate the effect of this type of attack [157].

Emissions Security (EmSec) is concerned with preventing attacks exploiting emissions,

namely radiated or conducted electromagnetic signals. There are different aspects of Emsec attacks. Active and passive Emsec measures are strictly related to ElectroMagnetic Compatibility (EMC) and Radio Frequency Interference (RFI), which can disrupt systems accidentally. The trouble begins when the emitted energy holds sensitive data, so an eavesdropper can interpret and analyze such compromising emanations to steal information; thus, data encryption is essential. It is crucial to realize that these emissions shall be considered as an additional Virtual ElectroMagnetic (VEM) interface [158].

II. Network layer

This layer facilitates data connectivity to perception layer devices to accomplish the functionality of different applications in the application layer. The network layer is considered to be the connectivity provider for other layers. Therefore, there are probable security flaws that could compromise the operations of IoT architecture.

The *sinkhole attack* is one of the most destructive routing attacks in the IoT context. It collapses the network communication by generating additional traffic on the network. Sinkhole attacks compromise nodes, create fake information, and send routing requests to neighbour nodes. Cervantes *et al.* [159] proposed an IDS system for identifying sinkhole attacks on 6LoWPAN networks for IoT. Kashif *et al.* [160] proposed a new protocol, RAEED, for detecting sinkhole and DoS attacks. Ibrahim *et al.* [161] presented a new mechanism for detecting sinkhole attacks using the hop count technique in WSNs.

The *wormhole attack* is an internal attack that listens to the activities on the network without modifying them. There are different techniques used to detect wormhole attacks. Gupta *et al.* [162] presented a detection technique without using any hardware, such as a directional antenna and precisely synchronized clock. This approach is known as a wormhole attack detection protocol using a hound packet. It could detect the wormhole attack in the network or the nodes that were making this type of attack. Hu *et al.* [163] introduced the notion of a packet leash as a general mechanism for detecting and, thus, defending against wormhole attacks. A leash is any information that is added to a packet designed to restrict the packet's maximum allowed transmission distance. Leashes are produced to defend against wormholes over a single-hop wireless transmission. They proposed geographic leashes and temporal leashes. The temporal leash ignores any packets with a limited lifetime. Instead, a geographical leash ignores any messages from an unknown distance. Lai *et al.* [164] proposed the RPL routing protocol-based wormhole detection technique without using any hardware requirements or any particular nodes.

The *Sybil attack* is caused by malicious Sybil nodes which use fake identities to corrupt the IoT functionality and even violate data privacy. The fake identities on the network may result in spamming, launching phishing attacks, or disseminating malware. Demirbas *et al.* [130] used the signal strength measurements to detect this type of attack by deploying detector nodes to compute the location of the sender during message communication. The presence of another message communication with the same sender location but a different sender's identity indicates a Sybil attack. The assumptions of the proposed approach make it usable for static networks.

The IoT deployment architecture requires that every device should be identified uniquely on the network. The message communication for identification should be secure to ensure that the data being transferred to a device in the end-to-end communication reaches the spec-

ified target [165]. ECC can be used to secure *neighbour discovery*. The public key signatures of the ECC are used to identify nodes in the neighbour discovery phase. The symmetric and asymmetric keys management systems are used depending on the application requirements [117].

Privacy attack of cloud-based IoT: Cloud computing is considered a high-risk environment for developers, consumers, and businesses because the environment cannot be defined or controlled. Verification of log messages is used to protect cloud-based IoT from insertion, withholding, modification, and reordering of messages. Encryption, obfuscation, encrypted data processing, trusted platform module, sticky policy, anonymization, data segmentation, trusted third-party mediator, and key management can be used to mitigate the effect of cloud-based attacks and protect the information in the cloud [166].

III. Application layer

This section describes the different security attacks that may occur at the IoT application layer.

SQL injection is the most common web hacking technique. It consists of inserting a SQL query as input data from the client to the application. SQL injection enables an adversary to spoof identity, disclose, modify, delete, or make data unavailable on a system [167]. The OWSAP project [135] gives different recommendations of countermeasures for securing IoT. The security mechanisms include testing interfaces against injection, avoiding weak passwords, and using HTTPS protocols and firewalls. Furthermore, it is crucial to regularly update the firmware or software installed on the device using an encrypted transmission mechanism. The updated files should be obtained from a secure server and undergo proper signing and validation processes before installation. This ensures the integrity and security of the device's software, minimizing the risk of potential vulnerabilities[109].

XSS: is a kind of injection attack in which a malicious script is injected into trusted websites. XSS occurs when an adversary uses a web application to send malicious codes in the form of a browser-side script to another end-user. To mitigate the impact of XSS, OWSAP recommended implementing a Content Security Policy (CSP). CSP is a browser-side mechanism that allows the developer to create source whitelists for client-side resources of the web application, e.g., images, JavaScript, CSS, CSP, etc., through a special HTTP header, notifies the browser to execute or render resources only from that sources [135].

Phishing attacks are concerned with stealing users' credentials and then using them to hack and gain access to IoT devices and steal sensitive data. Phishing attacks send fraudulent transmitters that appear to the end-user as reliable resources. This type of attack can be contrasted by using specific algorithms to classify the spam email, check the contents of the emails, learn more about the limited data set, and develop a model that classifies and predicts whether an email is dangerous [168].

The *Application layer DDoS attack* is designed to attack the application by concentrating on specific issues and vulnerabilities. The DDoS attack involves different types of attacks such as UDP flood, ICMP/PING flood, Ping of Death, SYN flood, and Zero-day DDoS. Yin *et al.* proposed an algorithm to detect DDoS attacks. The main purpose of this algorithm is to detect whether an attack has occurred, find the real DDoS attack, and block the DDoS attack at the source. Thus, the simulation result of the proposed algorithm could detect the IoT devices from which a DDoS attack is launched within a shorter time and mitigate DDoS

attack [169].

In the literature, several contributions offer solutions to the security issues described above. A classification based on the IoT architecture is proposed, specifically addressing mitigations for each layer, including the perception, network, and application layers. A summary of important research results are presented in Tables 2, 3 and 4. In particular, in accordance with the layered architecture proposed in this chapter, each table lists the IoT security issues and countermeasures for the corresponding layer.

This section concludes by mentioning design tools available to developers for verifying the security properties of protocols. These tools belong to the category of formal methods and support the *security by design* development model. An example is provided in [170], where a general framework based on the IoT-Lysa specification language is proposed that relies on formal semantics, taking into account the costs and benefits of using encrypted communications.

The following section presents and discusses energy and memory usage attacks in smart devices in IoT and IoE systems. The next section also presents related work and background studies of energy and memory attacks and discusses possible detection mechanisms used in IoT and IoE smart devices.

2.5 Related Work and Background Reading

2.5.1 Energy Consumption Attacks

Energy-based attacks are often categorised as IoT sensing domain attacks, where the smart devices and sensors are the target [190]. Dabbagh and Rayes in [190] described the sensing domain attacks like vampire attacks, jamming attacks, sinkhole attacks, and selective-forwarding attacks. The vampire attack, among others, is considered an energy-based attack because it aims to destroy the battery of sensors. The researchers also identified four types of vampire attacks based on the techniques used to destroy power: Denial of Sleep, stretch attack, flooding attack, and carousel attack. Patil and Sharma in [191] also described several DoS attacks for wireless sensors. The authors mentioned two attacks that waste the energy of sensors, among others: Denial of Sleep and vampire attacks. Another category of attacks is related to DoS, but they can waste energy indirectly. These are jamming attacks, wormhole attacks, and path-based DoS attacks.

Energy consumption attacks and their analysis will be described in Chapter 3 and 4.

A. Fake Access Points Attacks

One of the most challenging security problems for wireless networks is detecting F-AP attacks. This attack is also called the rogue AP attack or the evil twin attack [192].

Detection of rogue AP attacks in the wireless network of a smart healthcare system is an essential aspect of wireless security [193]. A rogue device detection system using various techniques such as site survey, noise checking, MAC address list checking, and wireless traffic analysis has been proposed in [194]. The authors concentrated on detecting internal rogue devices, such as devices connected via a wireless network and used by employees on a corporate network. However, this approach cannot be applied to IoT devices due to resource constraints.

Table 2. Perception layer security.

Issue	Effect	Countermeasure	Ref.
Jamming attack	– DoS; – Communication blockage.	– Interference detection; – Implement reactive strategies based on the signal-to-interference-plus-noise ratio (SINR) measurements; – Game-theory anti-jamming techniques.	[141] [171]
Sleep deprivation attack	– Energy consumption; – Reduce the lifetime of the sensor.	– Round robin and hash-based schemes for the cluster head selection; – Device authentication.	[172] [143] [173]
Eavesdropping	– Unauthorized interception of private information; – Privacy violation.	– Physical layer security; – Protocol encryption (e.g., HIP, IPSec).	[150] [174]
MITM	– Unauthorized interception of private information; – Communication blockage.	– Physical layer security; – Protocol encryption (e.g., HIP, IPSec) – Authentication scheme (e.g., HIP)	[114] [149] [175] [176] [150] [174]
Frame injection	– Modification of data in transit; – Communication spoofing and blockage.	– Channel state information analysis.	[154]
Node tampering	– Unauthorized access to the hardware; – Damage the hardware.	– Disable testing and debugging tools; – Remove the unused interfaces such as USB ports; – Strengthening hardware design.	[122]
Emissions Security	– Exploiting electromagnetic information leakage; – Completely passive attack.	– Improve electromagnetic shielding; – Apply electromagnetic compatibility norms; – Improve the firmware.	[158] [121]

Mehndi et al. [195] proposed an approach that considers the Mac Address, Service Set

Table 3. Network layer security.

Issue	Effect	Countermeasure	Ref.
Sinkhole attack	– DoS; – Data leakage.	– Authentication of the sender and the receiver by using a hash chain method; – Attack detection is based on the attacker's reputation in the network.	[165] [128] [177] [159]
Wormhole attack	– DoS; – Packets redirection and messages are tunnelled in the wrong direction.	– Apply integrity checks on the forward packets; – Apply cryptography techniques for securing the nodes from the compromised wireless node.	[178] [179] [179]
Sybil attack	– DoS and network disruption; – Spoofing multiple nodes identities.	– Adversary received signal strength indication (RSSI) based detection scheme; – Location verification; – Blockchain to establish the validity and integrity of transactions between nodes.	[180] [181] [130] [182]
Discover neighbour devices	– DoS.	– Neighbour authentication based on a public key signature.	[165]
IP fragmentation attack	– DoS and Disruption; – Incomplete packets can cause operating systems and security appliance vulnerabilities.	– Cryptographic techniques to verify that received fragments belong to the same packet; – Timestamp and nonce added to the fragmented packets at the 6LoWPAN adaptation layer.	[125] [183] [131]
Privacy attack of cloud-based IoT	– Privacy violation;	– Controlling and verifying messages at different levels to protect the system. – End-to-end security, i.e., cryptography for providing secure communication between IoT devices and the cloud.	[117] [132]

Identifier (SSID), and signal strength of the AP to decide whether the AP is rogue or not. In detecting authorized APs, the MAC addresses of all visible APs are matched against a list of authorized APs. Tools such as Ettercap[22], Wireshark[23], and Snort[24] are used for filtering

[22]https://ettercap.github.io/ettercap/
[23]http://www.wireshark.org/
[24]http://www.snort.org/

Table 4. Application layer security.

Issue	Effect	Countermeasure	Ref.
SQL injection	– Spoofing identity; – Destroying data; – Get Admin rights.	– Use of prepared statements with defined queries; – Enforcing least privilege; – Escaping all user supplied input.	[135]
XSS	– Stealing credentials; – Delivering malware to the victim.	– Using frameworks that automatically escape XSS by design; – Enabling a content security policy.	[135]
Phishing attack	– User's credential spoofing; – Gain access to IoT devices, like security cameras.	– Aggregate a set of legitimate information from the end-user that the attacker cannot exploit.	[184]
Third-party software security	– Data leakage.	– Data encryption before saving them in the cloud; – Access control policies.	[185]
DDoS	– Device's resources depletion; – Denying the user of available bandwidth and resources.	– Machine learning detection schemes that detect deviations from normal traffic patterns; – Zero-day attack signature extraction.	[186] [187] [188]
IP misconfiguration	– Decreases system performance and reliability; – Unpredictable behaviour of the attack.	– Prevent misconfigurations.	[189] [138]

instances where the MAC address is spoofed. While Kilincer, Ertam, and Şengür [196] proposed an automated technique for detecting and preventing F-APs attacks in the network of IoT devices. The proposed experiment uses a Single Board Computer (SBC) and a wireless antenna (ODROID module). The operation was about: 1) creating an F-APs, 2) scanning the surroundings using the SBC and WiFi modules, and 3) detecting fake AP broadcasts.The F-APs have been assigned to an unauthorized Virtual Local Area Network (VLAN). This study [196] is limited and focuses on F-APs attack detection and prevention. However, the data collected about the network and some attacks are still possible without connecting.

B. DDoS and EC-DDoS Attacks

DDoS and EC-DDoS attacks are security threats by attackers that enter the WiFi network coverage area and inject many forged packets. Adversaries use this attack for two purposes: restricting usage of the WiFi bandwidth and preventing licensed users from communicating with the licensed AP to paralyze or reduce the WiFi network's performance [197].

Various studies have been dedicated to investigating the repercussions of DDoS attacks on web servers, mainly when these attacks originate from compromised IoT devices. For example, Kambourakis [198], Marzano [199], Tushir [200], and Kolias [201] discussed the outbreak of the Mirai botnet (and its variants), which compromised IoT devices to launch a DDoS attack against data centres. They claim that even naive techniques can be used to take control of such devices and create a massive and highly disruptive army of zombie devices. Liu and Qiu [202] investigated de-authentication and disassociation DDoS attacks involving overwhelming wireless devices with fake de-authentication and disassociation packets. They observed that increasing the attack rate leads to a drop in TCP throughput and a rise in UDP packet loss. Furthermore, they proposed a client-device-based queuing model to demonstrate that the existing IEEE 802.11w standard fails to address de-authentication and disassociation issues at high attack rates. Moreover, there are different approaches to monitoring IoT devices' energy consumption to detect IoT cyberattacks. Tushir *et al.* [200] quantitatively studied the impact of DDoS attacks on smart home IoT devices and their energy consumption. However, they did not present any detection or mitigation solutions.

Despite this, many developed methods exist to detect and prevent DDoS and F-AP attacks in IoT systems. The presented approach in Chapter 3 focused primarily on assessing the impact of the combination of DDoS, EC-DDoS, and F-APs attacks on energy consumption, response time, and connectivity of smart healthcare devices. Therefore, the problem addressed in Chapter 3 is the energy consumption efficiency of smart IoT devices. The chapter focuses on studying the energy consumption of smart devices in various states, such as "Idle," "Active," and "Under Attack." Specifically, a combination of DDoS and F-APs attacks is implemented to impact the energy resources of smart healthcare devices. The proposed energy monitoring mechanisms facilitate the analysis of smart devices' energy behaviour both with and without attacks. The chapter's objective is to better understand the impact of DDoS, EC-DDoS, and F-APs attacks on the energy consumption and connectivity of smart healthcare devices within a wireless network. The results demonstrate the cumulative effect of DDoS and F-APs attacks on smart healthcare devices in different states. Notably, F-APs attacks contribute to 45% of the total effect, while DDoS attacks account for 55%.

Overall, the findings validate the effectiveness of the proposed testbed model in achieving comprehensive monitoring coverage of smart healthcare devices' energy consumption. This solution also aids in developing a detection mechanism against energy attacks in IoT systems. Therefore, in the subsequent chapter, a lightweight detection mechanism will be constructed to identify energy attacks in smart devices as presented in Chapter 4.

2.5.2 Detecting of Energy Consumption Attacks

Detecting energy consumption attacks is one of the essential tasks to be considered as its effects might cause severe problems to the privacy and reliability of the smart device. Since researchers develop many schemes and methods, but due to the constrained environ-

ment, e.g., low computational power and low energy of IoT, these techniques are not feasible. Therefore, an added line of protection that considers resource constraints should be built into IoT devices and networks to defend IoT-based organizations from cyber threats [203].

Various authors have presented detection techniques for energy consumption attacks in IoT systems. In [204], the focus is on promoting the use of more efficient smart devices as a key principle of energy efficiency. Home automation control is vital in achieving efficient and sustainable operations by minimizing energy losses, optimizing energy usage, and effectively managing the system's operational level. In the study [205], various home energy management systems are assessed to identify differences in functionality and quality, focusing on discovering opportunities for energy savings through behavioural and operational approaches. The adoption of energy-efficient scenarios is often influenced by their potential benefits in terms of comfort, convenience, and security. The study [206] introduces a detection framework for IoT systems that relies on energy consumption analysis. The proposed methodology involves analyzing the energy consumption patterns of smart devices and categorizing their attack status, distinguishing between cyberattacks and physical attacks. The framework employs a two-stage approach, using a short time window for initial rough attack detection and a longer time window for more precise attack detection. In the study by Felius *et al.*[207], various techniques were proposed to manage smart home systems and reduce energy consumption. The first approach, feed-forward control, involves real-time monitoring of interference factors and implementing appropriate adjustments based on known parameters. This system compensates directly for external factors such as wind, solar radiation, and internal heat gain. Another method explored is Model Predictive Control (MPC)[207], which predicts the system's future behaviour using models and adjusts the system accordingly. Additionally, Fuzzy Logic Control was introduced, which does not require a complex mathematical model but relies on user experience quality to control the system effectively. The paper by Hoffmann et al. [208] explored energy consumption analysis approaches, but it was concluded that these approaches might not be suitable for devices like smartphones due to significant differences in their typical energy consumption in real-world scenarios. Moreover, the presence of noise in the system caused by unpredictable user and environmental interactions could result in numerous false alarms. Practical tests were conducted, and it was found that the additional power consumed by malicious applications is too minimal to be noticeable, considering the mean error rates of state-of-the-art measurement tools. However, the study did indicate that DDoS attacks could be detected by analyzing the energy consumption patterns of similar devices. In this paper [209], the author presents a method for detecting IoT attacks by analyzing the energy consumption of smart devices, taking into account user preferences related to energy consumption. The primary objective is to improve the accuracy of IoT cyberattack detection and pinpoint the presence of IoT malware on these devices. The study involves analyzing the IoT software opcode sequences to enable the detection of various IoT device performances, including DoS and DDoS attacks. To the best of our knowledge, the work in Chapter 3 is the first to detect energy consumption attacks in smart home devices, depending on measuring the packet reception rates by the smart devices.

To the best of the author's knowledge, the presented work [21] in Chapter 4 is the first work to detect energy consumption attacks in smart home devices, depending on measuring the packet reception rates by smart devices. The author builds a lightweight detection algorithm to detect energy attacks in smart devices. Therefore, Chapter 4 presents a com-

prehensive solution for detecting energy attacks in IoT systems. A monitoring mechanism is developed to track both the packet reception rate and energy consumption of smart devices. Monitoring these parameters enables a deeper understanding of the device's behaviour under different states (Idle, Active, and under attack). The proposed algorithm is designed to detect energy attacks across protocols such as MQTT, TCP, and UDP.

The results of the study demonstrate the effectiveness of the algorithm in providing complete monitoring coverage for detecting energy attacks in IoT systems. The successful detection of energy attacks is achieved by monitoring the packet reception rate and measuring the energy consumption of the smart devices. These findings are significant for implementing packet reception rate monitoring in critical IoT networks.

2.5.3 Memory Consumption Attacks

In Chapter 5, a pioneering approach is presented that elevates the originality compared to existing studies by focusing on detecting memory attacks based on meticulous monitoring of memory usage in smart devices within the IoE and IoT domains. The ubiquity of the IoE has catalyzed an unprecedented level of interconnectedness among people, data, things, and processes [210, 211], necessitating rigorous cybersecurity evaluation akin to traditional features like durability, suitability for purpose, and maintenance [212]. Despite the burgeoning emphasis on IoE security in legislation and common standards, standardized and independent verification of IoE devices remains in its infancy. Several authors have explored memory analysis to prevent and identify attacks in the IoE environment proactively [211]. Memory analysis has garnered considerable attention among malware researchers [213]. Researchers such as Vömel *et al.*[214] have surveyed various memory acquisition and analysis techniques. Rathnayaka *et al.*[215] have observed that successful malware infections leave traces in memory. Zaki et al.[216] have investigated artifacts left by rootkits at the kernel level, including driver and module modifications, SSDT and IDT hooks, and callbacks. Aghaeikheirabady[217] has presented an analysis approach that compares features extracted from memory, such as function calls, DLLs, and registry information, to enhance accuracy. Although the approach achieves an overall accuracy of 98% using Naïve Bayes, it is associated with a high false positive rate exceeding 16%.

Similarly, Mosli *et al.*[218] introduced a technique for malware detection based on extracting API calls, registry, and imported libraries from memory images. Although individual feature experiments achieved up to 96% accuracy using the SVM classifier on registry activities, their subsequent work[219] focused on utilizing process handles in memory for malware detection. The experiment showed that malware commonly used process handles, mutants, and section handles, achieving a modest accuracy slightly above 91% with the random forest classifier. In another study, Duan *et al.*[220] presented an approach to extract live DLL features from memory to detect malware variants. The hidden Naïve Bayes classifier achieved an accuracy of 90%. Additionally, Dai *et al.*[221] proposed a malware detection and classification approach based on converting memory images into fixed-size grayscale images and extracting features using gradient histograms. The neural network classifier achieved an accuracy of 95.2%. Moreover, in a previous work by the authors, API calls from behaviour analysis and memory analysis were combined into a single vector representation for each sample, demonstrating that memory analysis can overcome the limitations of behaviour analysis [222].

In this innovative study, the memory usage of smart devices in both the IoE and IoT environments is diligently monitored to detect memory usage attacks and mitigate resource constraint challenges. A specialized testbed environment is meticulously utilized to measure the memory usage of smart devices before and after attacks on the IoE and IoT environments. The proposed model focuses on continuous memory usage monitoring of smart devices, enabling the detection of memory usage attacks. The model encompasses three crucial steps: a lightweight monitoring mechanism to continuously monitor memory usage, a detection mechanism to identify the initiation and cessation of memory usage attacks, and a mitigation process to prevent further reading and writing to memory, blacklist affected smart devices, and disconnect them from Internet access.

This pioneering approach is meticulously applied and tested on a range of smart devices, including Raspberry Pi and Arduino, each with distinct architectures. Various Python and C libraries are adeptly utilized for monitoring memory usage. The detection and mitigation algorithm has demonstrated remarkable efficiency in detecting memory usage attacks within IoT systems. By continuously monitoring and analyzing the current and previous memory measurements, the algorithm categorizes memory usage in smart devices as normal or abnormal. Moreover, the technique is thoughtfully designed to be lightweight, considering the resource constraints intrinsic to smart devices.

The experiments conducted as part of this study, as showcased in [22], provide empirical evidence of the effectiveness of the proposed approach for detecting and classifying memory usage attacks. These evaluations include an assessment of classification accuracy, memory usage monitoring, and attack detection. The results attest to the robustness and reliability of the proposed mechanism, underscoring its potential to bolster the security posture of IoT and IoE environments significantly. Finally, the algorithm presented in Chapter 5 demonstrates high efficiency in detecting memory usage attacks in smart devices. The memory usage during detection fluctuates between two normal states: *Idle* and *Active*. For instance, memory usage remains below 35% for the Raspberry Pi and less than 16% for the Arduino. Additionally, CPU usage is measured for the Raspberry Pi, registering a final percentage of less than 2.5% during the detection mechanism.

Chapter 3

Analysis of the Impact of Energy Consumption Attacks on Smart Devices

The rapid growth of IoT technology has revolutionized human life by inaugurating the concept of smart healthcare, smart devices, smart city, and smart grid. The security of IoT devices has emerged as a significant concern, particularly in the healthcare domain, where recent attacks have exposed critical vulnerabilities in IoT security. In addition, in IoT networks, the connected devices are vulnerable to attacks such as attacks that affect the resource constraints of healthcare devices, e.g., energy consumption attacks. Therefore, this chapter defines the impact of DDoS and F-APs attacks on WiFi smart healthcare devices and studies the underlying reasons from the perspective of the attacker, victim device, and AP. This work focuses on IoT devices' connectivity and energy consumption when attacked. The main key findings of this chapter are as follows: (i) the minimum and maximum attack rate of DDoS attacks that cause service disruptions on the victim side, and (ii) the minimum-the higher effect of EC-DDoS and F-APs attacks on the energy consumption of the smart healthcare devices.This study highlights the importance of communication protocols, attack rates, payload sizes, and the state of victim devices' ports as key factors influencing the energy consumption of these devices. These findings contribute to a comprehensive understanding of the potential vulnerabilities of IoT devices in smart healthcare environments. They serve as a solid foundation for future research endeavours aimed at developing effective defence solutions to mitigate the impact of energy consumption attacks.

This chapter is organized as follows: Section 3.1 provides detailed insights into the research problems, offering a clear understanding of the motivations and objectives driving the study. Section 3.2 serves as a general introduction to the chapter. In Section 3.3, the attack scenarios and assumptions are explained. Section 3.4 outlines the energy monitoring objectives used in the experiments of this chapter. The testbed scenario, data collection process, F-APs setup, and the most significant influential factors are presented in Section 3.5.

In Section 3.6, different results regarding network scans, disconnections caused by DDoS attacks, energy consumption measurements, and the effects of EC-DDoS and F-APs on energy consumption are described. Lastly, Section 3.7 provides the experimental evaluation, illustrating the impact of energy consumption attacks on smart devices. The content of this chapter is mainly taken from [20].

3.1 Problem Statement, Motivation and Objectives

The problem addressed in this chapter is the efficient, full monitoring of the smart devices' resource constraint problems and detection of resource constraint attacks within IoT systems. We target the detection and mitigation of resource-constraints attacks. The primary goal of the energy monitoring mechanism is to ensure resilient network connectivity. The objective is to detect energy consumption faults in a proactive and efficient manner by continuously monitoring the energy usage of smart devices. This includes monitoring the status of all devices within the network. This chapter provides energy monitoring mechanisms in order to detect energy attacks in smart devices. This study has been done in real smart IoT devices, and the final results show the effect of EC-DDos, DDoS, and F-APs attacks on smart healthcare devices. The monitoring mechanism of energy consumption must have minimal effect on the energy consumption of smart devices within the IoT system. Energy consumption attacks have been analysed on smart healthcare devices because of their important effects on the healthcare community.

Therefore, new threats constantly emerge since IoT healthcare devices operate in an interconnected and interdependent environment. Moreover, as IoT healthcare devices are typically used in an unattended environment, intruders may maliciously access these devices. Eavesdropping can access privately owned information from the communication channel because IoT devices are usually linked through wireless networks. In addition to these security issues, IoT devices cannot afford to incorporate advanced security features because of their limited energy and processing power. Therefore, it is essential to study the effect of malicious attacks on the energy consumption of smart healthcare devices and show their impact, as smart healthcare systems are much more vulnerable and sensitive to their privacy and security. More importantly, considering the massive amount of smart healthcare devices on the market, the impact of energy consumption attacks cannot be neglected. Our main contribution is studying the effect of a practical combination of F-APs and DDoS attacks on smart healthcare devices. The main purpose of choosing DDoS and F-APs attacks as their impact on IoT security is high, so many researchers are working on solutions [223]. Thus, we target the energy consumption of smart devices. In the first step, we design a smart system to measure the current consumption of smart healthcare devices. Also, we build a testbed to monitor the smart devices and capture devices' status, e.g., *On* or *Off*, network traffic, and energy consumption. We identify several critical influential factors, particularly in terms of communication protocol, Attack Rate (AR), payload size, and victim devices' port status. We study the impact of these factors on the victim devices' resource constraints, such as energy consumption. In the second part of the contribution, the attacker disconnects the smart devices from the local AP by sending DDoS attacks. At the same time, we study the effect of EC-DDoS on energy consumption by sending malicious attacks to affect the energy resources of smart devices. Also, we implement real-time energy monitoring on real smart

devices to register the effect of the attack. In the last part of the contribution, we designed the F-AP to force the smart devices to connect to it once it disconnected from the local AP by DDoS attack. The F-AP is designed to automatically send malicious attacks,e.g., DDoS, EC-DDoS, or others, affecting smart devices' energy consumption.

This contribution gives a valuable understanding of the effect of DDoS and F-AP attacks on the energy consumption of smart healthcare devices by presenting a testbed and real energy consumption tests. Energy consumption attacks can destroy smart devices and impact patients' lives. The analysis of this research is used to build detection mechanisms for energy attacks in IoT devices as described in Chapter 4.

3.2 Introduction

IoT smart devices play a vital role in various aspects of human life, such as healthcare and transportation [40, 222]. The IoT has brought about a transformative impact on technology and society, continuously increasing the number of IoT devices. According to a report by Cisco, it is projected that approximately 29.3 billion networked devices will be connected to the Internet by 2023 [224]. As the number and heterogeneity of smart devices are accelerating rapidly, it is becoming challenging to maintain the security of these devices [225]. IoT footprints have been identified in various domains such as manufacturing, agriculture, transportation, electric grid, and healthcare [226]. In an IoT-based healthcare system, security is the primary concern as the data is directly related to human beings [227]. An Intensive Care Unit (ICU) is a special and critically operational hospital department where specialized treatment is given to patients requiring critical medical care. Usually, patients who are acutely unwell or injured severely and require continuous medical care are admitted to the ICU. The equipment and devices concerned in the ICU play a vital role in keeping the patient alive and healthy. In such a scenario, any communication breakdown due to a cybersecurity breach may cause severe effects on a patient's life and even death in some instances [228].

Moreover, smart healthcare devices typically interact through different wireless communication protocols that allow adversaries to perform different attack types. For example, eavesdropping, creating F-APs, DDoS, and EC-DDoS [229]. Attackers use DDoS attacks to launch malicious traffic to damage target smart healthcare devices by affecting their resources and disconnecting them from the legitimate AP. EC-DDoS attacks lead to an increase in the target's energy consumption to destroy it by sending malicious traffic. F-APs attacks force smart healthcare devices to connect to an alternative AP, monitor the transferred packets, and then launch malicious attacks to consume more energy. Typically, most IoT devices have limited processing capabilities, and applying advanced security techniques to each device is challenging [230]. Using them in the smart healthcare system may also give unauthorized access to cybercriminals to monitor patients' private data and exploit sensitive information or send attacks to consume more energy and destroy smart devices [231]. Previous studies have focused on conducting static and dynamic analyses and implementing defences against DDoS and F-AP attacks. However, a significant limitation of existing dynamic analyses is that they are primarily carried out in virtual environments, hindering accurate measurement of resource consumption for compromised devices, especially energy usage [18]. To overcome this challenge, this work addresses the issue by conducting experiments in a controlled environment using real-world devices. This approach enables gath-

ering precise data on the impact of attacks on resource-constrained IoT healthcare devices within a cost-effective experimental setup.

In this chapter, we study the effect of DDoS, EC-DDoS, and F-APs attacks on WiFi connectivity and energy consumption perspectives. The results show the significant damage that could be caused by these attacks and draw attention to the urgent need for effective defence solutions. This chapter can be used as a framework to test smart healthcare devices' security and create security standards for robust, predictable, and tamper-free operations.

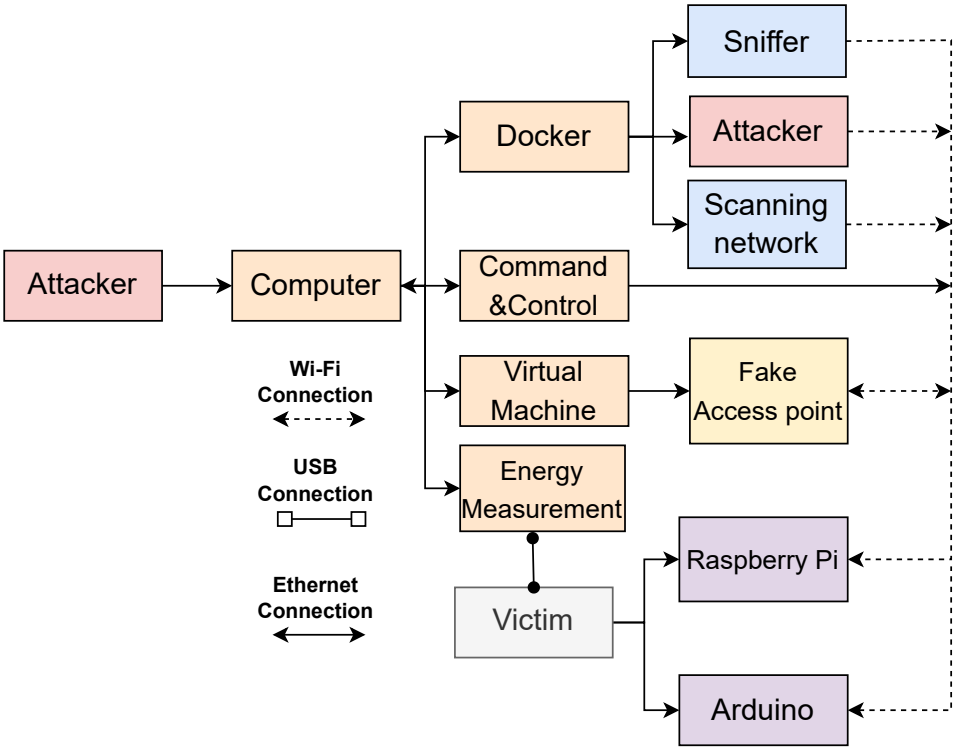

Figure 19. Testing Environment.

3.3 Attack Scenario and Assumption

Attack Scenarios. Figure 20 illustrates the potential scenarios where DDoS, EC-DDoS, and F-APs attacks can be applied. First, F-AP is designed to look like the actual AP. In those scenarios, the attacker can set an F-AP to launch different attacks to affect the energy resources of smart healthcare devices. The F-AP signals could be more vital to the victim than the actual AP. Once disconnected from the actual AP by sending DDoS attacks, the tool forces smart healthcare devices to automatically reconnect to the F-AP, allowing the attacker

to intercept all the traffic to that smart healthcare device, such as MITM attacks. Sniffing tools can also be applied to get or edit the information sent or received from the victim's devices. Additionally, the attackers may use F-AP and EC-DDoS attacks to destroy smart healthcare devices by affecting resource usage, e.g., energy consumption.

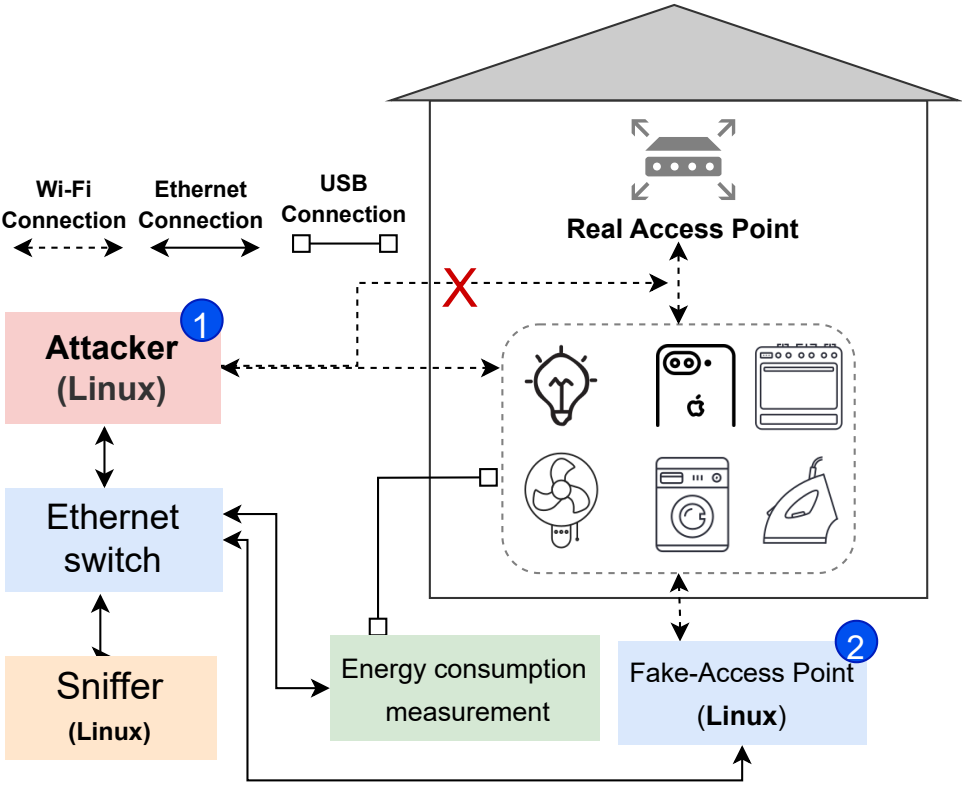

Figure 20. Attacking Scenarios.

Assumptions. The attacks we consider require the attacker to send DDoS attacks to force the device to disconnect from the legitimate AP and affect its resources. Then, the adversary could set up the F-APs attack at different locations. The adversary may set the F-APs at a distance from the victim to avoid being caught. As a result, the smart device is connected to the F-APs created by the attacker. The potential F-APs attack relies on sniffing over the WiFi to capture all packets travelling to and from the monitored smart device. Also, the F-AP is designed to affect the energy consumption of smart devices by automatically sending malicious attacks to connected smart devices.

Therefore, in this chapter, we investigate the effect of F-APs and EC-DDoS attacks on smart healthcare devices' energy consumption by implementing them with malware de-

signed to increase the energy consumption of smart healthcare devices and destroy them.

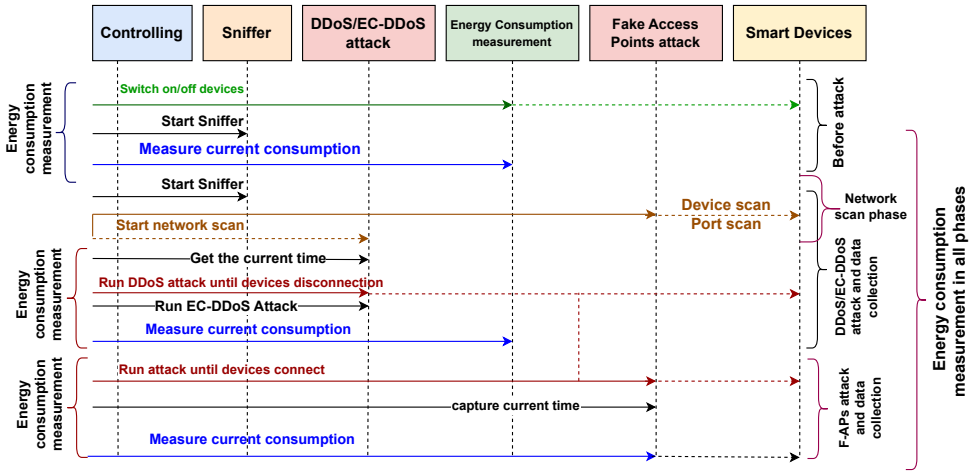

Figure 21. Sequence diagram showing an attacker intercepting and affecting energy measurement of the smart healthcare devices.

3.4 Energy Monitoring Objectives

Energy monitoring mechanisms, in general, aim at detecting and localizing energy attacks, providing the appropriate tools for overseeing the energy state and availability of devices. The necessary corrective energy measures can be taken by mapping symptoms of detected energy problems to possible attack causes. Therefore, we developed a smart circuit using a non-invasive current sensor, as shown in Figure 22, to measure the current consumption of smart healthcare devices. This smart circuit samples voltage, ampere, watt, and current per second. The current consumption values for each smart healthcare device are stored in the DB. In our experiment, we use the Joule (J) values to calculate the energy consumption of smart devices.

Let us describe the energy (E) measurement footprints considering the set of different device statuses in the absence or the presence of the attack.

$$E(d) = f(e(d), n, ATK) \quad \text{and} \quad n \in [0, 1] \tag{3.1}$$

Where (e_d) the energy measurement (e) of the smart device (d) at a point in time in the absence or presence of cyberattacks (ATK), and n is the number of energy measurements in a time interval, "$f(e(d), n) \in [0, 1]$" where 0 is the minimum energy consumption measurement, and 1 presents the maximum energy consumption measurement in the absence or presence of the attack. The smart IoT devices' energy consumption is calculated and analysed before and after attacking the smart devices. The main purpose of this calculation is to study the effect of the energy consumption attack and build a solid foundation for detecting this type of attack as described in Chapter 4.

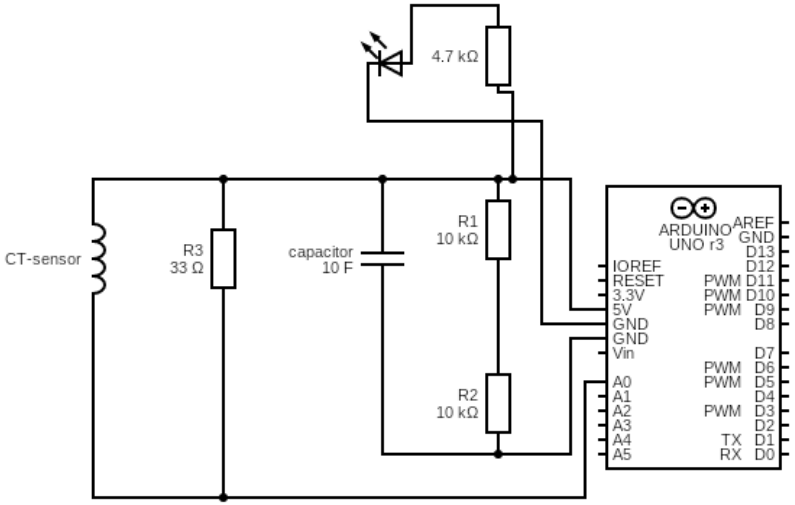

Figure 22. Circuit for measuring current consumption.

3.5 Proposed Testing Environment

In this section, we explore various attack scenarios targeting smart healthcare devices and provide insights into the testbed, network scan, and data collection process. The testbed serves as a platform to investigate the impact of DDoS and F-AP attacks on energy consumption. Additionally, we analyze the status of devices and ports to identify potential vulnerabilities and weak points in the system.

3.5.1 Experiment Setup

There are two types of attacks on IoT devices: internal and external. Internal attacks happen when the adversary has access to the local network; this is possible by hacking the WiFi or gaining access to IoT devices. For example, attackers may gain access to the device by launching internal attacks to access a local Linux-based device remotely or by sending packets from outside the network for the external attack. For instance, if the attacker can force smart healthcare devices to disconnect from the actual AP and connect to the F-APs, then the adversary can send packets to the device outside the local network.

The testbed contains different smart healthcare devices, e.g., Arduino and Raspberry Pi, as proof of concept. Furthermore, three Linux-based images were created using Docker, as shown in Figure 19. These images involve 1) an attacker sending malicious packets to the victim devices, 2) a sniffer for capturing WiFi traffic, and 3) a control system to scan the network and get different information about the port and devices' status.

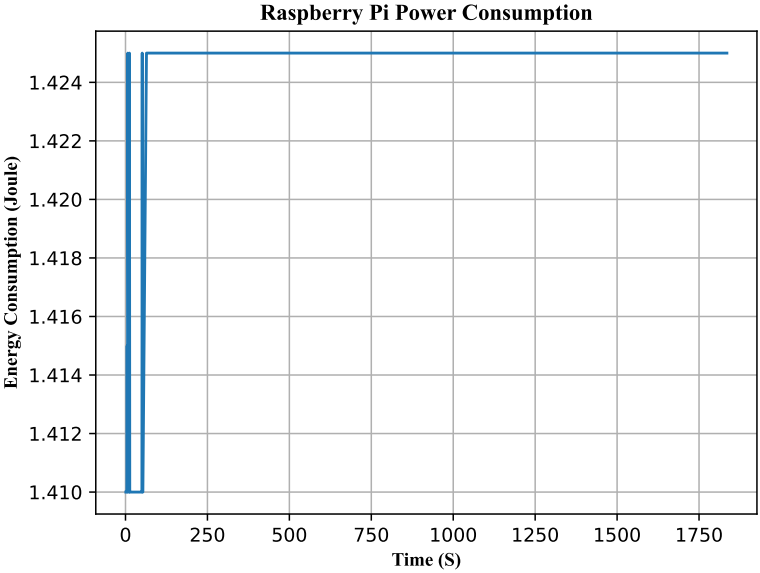

Figure 23. Energy Consumption of the Raspberry Pi (Normal).

Moreover, we created the F-APs using a TP-Link TLWN772N USB adapter and a Linux-based software system. Then, we designed a smart meter to measure the energy consumption of smart healthcare devices using a non-invasive current sensor [232] with an Arduino and some other resistors. Also, we used different software tools for attacking data generation and collection. On the adversary side, we used Nmap[1] to launch a network scan and identify devices' status, such as *online* or *offline*, IP address, and MAC address. Then, we ran a TCP/UDP port scan on the victim devices to identify port status (open/closed, filtered/not filtered, and others). Furthermore, we used hping3[2] to generate DDoS and EC-DDoS attacks by adjusting the AR, source IP address, destination IP address, payload, attack type, flags of TCP sessions (SYN, ACK, FIN, push, or urgent), and port types. In addition, we used hostapd (host access point daemon) and dnsmasq with a TP-Link TLWN772N USB adapter to create the F-APs. Hostapd is a user-space daemon software enabling a network interface card to act as an AP and authentication server. Dnsmasq is a lightweight, easy-to-configure DNS forwarder designed to provide DNS (and, optionally, DHCP and TFTP) services to a small-scale network. The TP-Link TLWN772N is a USB adapter that acts as an F-AP.

We used tshark to evaluate the impact of EC-DDoS and F-APs attacks on the resource constraints of smart healthcare devices and capture WiFi traffic. Figures 20 and 25 show the

[1]https://nmap.org/
[2]https://www.kali.org/tools/hping3/

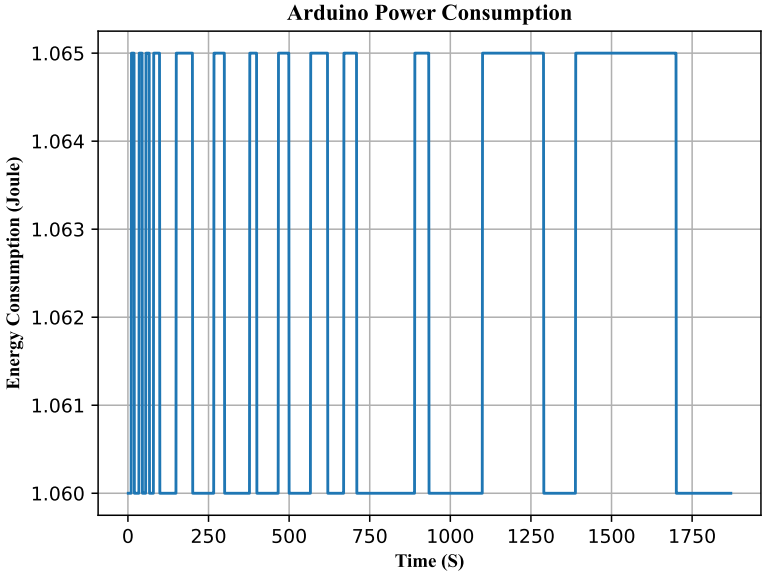

Figure 24. Energy Consumption of the Arduino (Normal).

attacking scenarios. Different stages are used to run our experiment. In the first stage, we measure the energy consumption once the device is turned *On*. Another measurement is used when the device is connected to the AP. Then, we run a network scan to capture the port and device status. Once we ensure that the device is connected to the Internet, we send DDoS and EC-DDoS attacks for two purposes: first, to consume more energy, and second, to disconnect the device from the local AP. Then, we run energy consumption measurements to calculate the energy consumption of the devices and study the devices' behaviours under DDoS and EC-DDoS attacks. Next, we run the F-APs attack on smart devices. We first check whether the devices are already disconnected from the local AP. We then force the smart devices to connect to the F-APs and finally start measuring the energy consumption of smart healthcare devices. Also, we study the behaviour of smart devices in terms of energy consumption and connectivity. The F-AP works as a MITM attack. The F-APs are used for different purposes: 1) monitoring the devices, 2) sending malicious packets to consume more energy, and 3) affecting the CPU usage of the smart healthcare devices.

3.5.2 Collecting Data

We built a smart healthcare test environment by deploying different smart devices, as shown in Figure 25. Data is aggregated from various smart devices for analysis purposes. Moreover, the aggregated data is categorized into behavioural and network data. Behavioural

Algorithm 1 the affecting of F-APs and DDoS attacks on energy consumption

procedure SMARTDEVICE(a) ▷ *consume more energy of (a).*
    Sniff air for network scanning
    E1: energy consumption before the attack
    E2: energy consumption after the attack
    if $a = connected$ **then** ▷ *a is connected to the AP*
        Calculate energy consumption
        Disconnect using DDoS attack
    else if $a = disconnected$ **then** ▷ *from the actual AP*
        Send DDoS attack
        Calculate energy consumption after the attack
    if $E1(\mathrm{a}) < E2(\mathrm{a})$ **then**
        Connect the device to F-APs
        Send malicious attack to consume more energy
    else if $E1(\mathrm{a}) < E2(\mathrm{a})$ **then** ▷ *energy consumption after attack*
        Send another packet of DDoS attack
        Consume more energy and disconnect the devices
    else
        Try to consume more energy
        Calculate energy consumption, AR, survival duration (SD), and threshold(AR)
    while $E1 \neq 0$ **do** ▷ *Consume more energy to destroy the device*
        $E1(\mathrm{a}) \leftarrow E2(\mathrm{a})$

data refers to the status of the smart devices, like *On* or *Off*, and the device's readings. Network data refers to smart healthcare devices' TCP and UDP packet data. We integrate this data to learn typical behaviours in a smart healthcare environment.

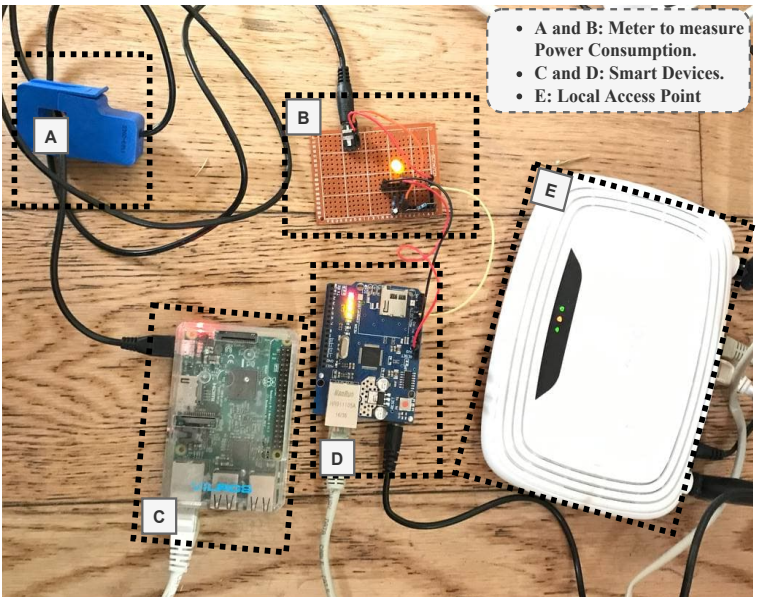

Figure 25. Proof of Concept for Wireless Network smart healthcare devices.

Figure 21 shows different phases of our attack scenario; in the first phase, we control the smart devices by switching them *On* or *Off*; this is essential to calculate the energy consumption of the smart devices before launching any attack. In the second phase, we measure the energy consumption of the smart devices for at least 30 minutes to calculate the average energy consumption before launching any attack; then, we start sniffing the network to get information about ports and devices' status. In the final phase, we launch DDoS and EC-DDoS attacks to impact smart devices' connectivity and energy consumption. Simultaneously, the energy consumption of appliances is measured to show the impact of this attack on the energy consumption of smart healthcare devices. After that, once the devices are disconnected from the legitimate AP using DDoS attacks, we calculate AR, Survival Duration (SD), and the threshold of the AR. Next, we force the devices to connect to the F-APs, where the fourth phase will start. In this phase, we start monitoring and collecting information about the smart devices using F-APs facilities. We then launch malicious attacks through the F-APs to consume more energy and study the behaviours of smart devices' energy consumption under F-APs attacks. Through that, we achieve the main purpose of destroying smart healthcare devices by using energy consumption attacks caused by F-AP and EC-DDoS attacks, as shown in Figure 26.

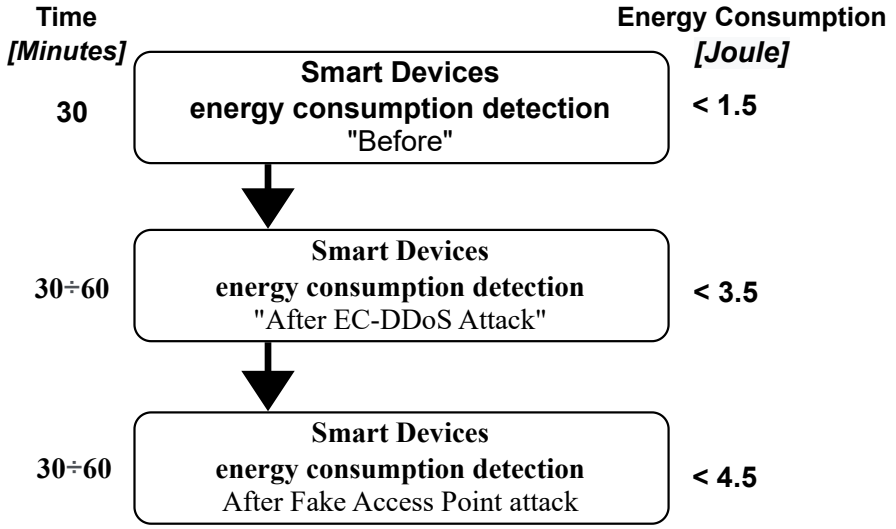

Figure 26. Energy Consumption affects before and after attacking smart healthcare devices.

3.5.3 Setting up a Fake Access Point

We used hostapd (host access point daemon) and dnsmasq with a TP-Link TLWN772N USB adapter to create the F-APs attack, broadcast a fake signal, and capture the victim's packets. The F-AP configures the same SSID, Basic Service Set Identifier (BSSID), broadcast channel, and security settings as the legitimate AP. The scenario with the F-APs attack is reported in Figure 20. The monitor mode of the F-APs is enabled using *airmon-ng start* for capturing attack injections or packets from and to the smart healthcare devices. The F-APs work as an MITM attack to capture packets transferred between the smart devices and the server. Moreover, the F-AP is designed to send malicious attacks to consume more energy of the connected smart healthcare devices[3].

3.5.4 Determining the weak side

In this section, the primary focus lies in examining the impact of disconnections and power consumption. We specifically consider victim devices with varying hardware configurations, such as differences in CPU, WiFi chip, and memory. It is important to note that these devices may exhibit diverse responses when subjected to a given attack. Arduino and Raspberry Pi are considered.

The port state and communication protocol can significantly influence victims' responses during an attack. Accordingly, Three types of attacks are conducted in this study: TCP-SYN,

[3]https://github.com/developerZA/EnergyConsumptionAttack.git

UDP, and Internet Control Message Protocol (ICMP) echo request attacks. For TCP-SYN and UDP attacks, targets with different port states, such as open, closed, filtered, and open-filtered, are selected. In the case of ICMP attacks, the port number is not specified.

Considering that smart IoT devices have limited resources, their behaviour can vary significantly depending on the payload size of the attack packets. Therefore, the payload of attack packets is configured with two settings: 0 B, representing no payload (NP), and 1500 B, representing high payload (HP). This allows us to explore the impact of payload size on the devices' responses to the attacks.

The relationship between DDoS and EC-DDoS attacks and the Attack Rate (AR) has been acknowledged as having a direct impact. In Section 3.6.2, we analyse AR's effect on service disruption by determining the minimum AR required to disconnect devices from the legitimate Access Point (AP). Furthermore, in Section 3.6, we investigate the impact of EC-DDoS and F-APs attacks on the energy consumption of smart healthcare devices.

3.6 Experimental results and analysis

In this section, we describe an experimental workplace to test the effect of DDoS, EC-DDoS, and F-APs attacks on the energy consumption of smart healthcare devices. This experiment focuses on collecting incoming malicious attacks and the usage statistics of a victim device and analyzing the attack effects on the victim devices in terms of energy. The network scan of the smart devices is used to obtain the status of the ports and then to determine the weak side of smart devices by calculating their AR and SD. Moreover, we study the effect of EC-DDoS and F-APs attacks on energy consumption.

3.6.1 Network Scan

The network scan operation involves gathering crucial information about the victim's smart devices, including their online or offline status, IP address, and MAC address [109]. On the other hand, the port scan enables an attacker to determine the status of TCP and UDP ports on the target devices. The ports can be in one of the following states: open, closed, filtered, or open-filtered. For a comprehensive overview, Table 5 presents the port statuses observed for the devices utilized in our testbed.

Table 5. Network scan result in terms of port status for TCP and UDP protocols.

Device	TCP scanned ports	UDP scanned ports
Raspberry Pi	3 open, 998 open-filtered, 65389 filtered and 0 closed ports	4 open and 700 open-filtered and 0 closed ports
Arduino	1 open, 22 filtered, 1000 open-filtered and 0 closed ports	1000 open-filtered ports

.

3.6.2 Attack Rate and DDoS Attacks

In our study, the threshold AR is defined as the minimum AR, measured in Packets Per Second (PPS), that leads to the disconnection of the victim's device from the AP. The SD represents the time duration between the commencement of an attack and the resulting device disconnection caused by the attack. To capture a comprehensive range of attack scenarios, we have configured the maximum attack duration to vary between 8 and 30 minutes. During the experimentation, ICMP, TCP-SYN, and UDP attacks were launched against the victim devices, targeting their open, filtered, and closed ports. This enabled the collection of their respective threshold AR and SD values for analysis.

The AR applied to the Raspberry Pi is between 500 to $10,000$ PPS for both NP and PH. In contrast, the AR sent to the Arduino for NP attacks is between 100 to 800 PPS, as the threshold AR is 800 PPS. We did not use a PH attack against the Arduino because it disconnects with minimal AR. Table 6 reports the average SD in minutes for the smart devices. We can see that the Arduino device disconnects in all cases with different attacks. Instead, the Raspberry Pi disconnects only with low packets. The Raspberry Pi survives with a higher AR than the Arduino, with 20 k packets at NP. The AR of the Arduino is 800 packets at NP and 200 packets at PH. The tshark files show that the Raspberry Pi broadcasts probe requests and sends de-authentication packets to the legitimate AP. The main difference between the smart devices is that the Raspberry Pi has more powerful hardware than the Arduino.

Table 6. Survival Duration (SD) caused by DDoS attack.

Survival Duration	**Raspberry Pi**		**Arduino**	
	NP [Minutes]	*PH [Minutes]*	*NP [Minutes]*	*PH [Minutes]*
SD (ICMP)	7.58	none	3.6	3.13
SD (TCP)	6.2	none	3.3	2.44
SD (UDP)	7.8	none	3.8	2.44

Through the experiment, when we calculate the received AR by the victim devices, we find that the victim devices rarely receive the actual AR sent by the attacker. For example, we sent about 15 k packets to the open ports of the Raspberry Pi; the received packets were about 14544 packets. Also, we can notice that the increase in the average packet rates sent by the attacker causes an approximately logarithmic increase in the received packets by the victim.

3.7 Experimental Evaluation

3.7.1 Energy Consumption Attack and IoT devices

To calculate the energy consumption of the devices versus the incoming attack reception rate of the victim devices, we need to collect the data from both the sensors and the tshark data. Therefore, all data relevant to this experiment is stored automatically in the DB. During the process of packet collection, the attacks are initiated using TCP, UDP, and ICMP flood commands. The topology illustrated in Figure 20 is employed to send the malevolent TCP,

UDP, and ICMP traffic individually to the target device. Throughout this procedure, the victim device records all usage statistics related to energy consumption and connectivity. Each attack simulation persists for a duration of 1 second, summing up to a total of 30 minutes, with corresponding usage statistics being recorded over the same duration.

Figure 27 shows the device's energy consumption when its status is *On* in the absence of attacks on that device. The standard energy consumption of the Raspberry Pi is between 1.410 J and 1.420 J per second. However, the current consumption varies from 1.410 J to more than 3.3 J per second after launching TCP-SYN attacks on open ports of the Raspberry Pi. In contrast, we can notice that the energy consumption increases to more than 3.60 J per second after launching ICMP attacks on open ports of the Raspberry Pi. Additionally, the energy consumption fluctuates between 1.4 J and 3.50 J per second after launching a UDP flood attack because of the overload that might have happened on the Raspberry Pi's open ports.

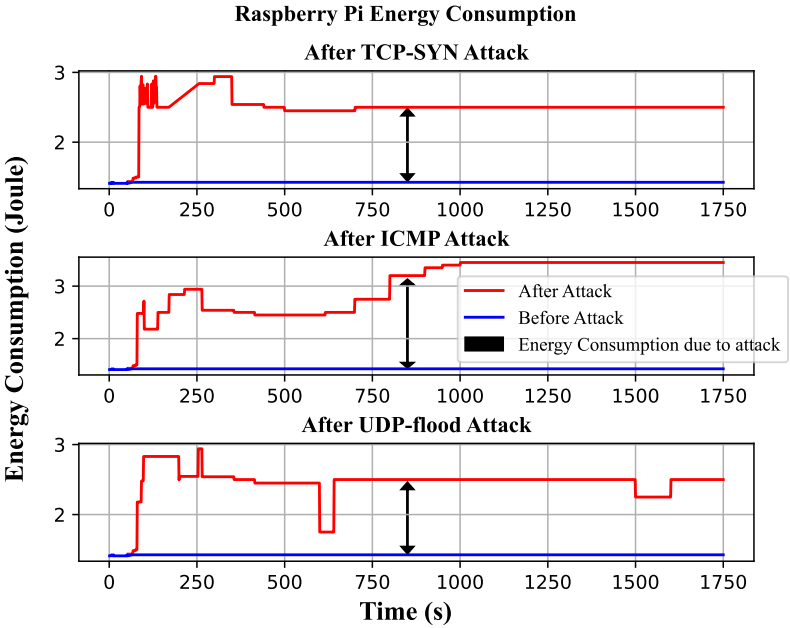

Figure 27. Raspberry Pi Energy Consumption under EC-DDoS Attack.

Figure 28 shows the current consumption of the Arduino when its status is *On* in the absence of attacks. The standard energy consumption of the Arduino is between 1.060 J and 1.065 J per second. In contrast, the energy consumption varies from 1.065 J to more than 1.75 J per second after launching TCP-SYN attacks on NP. At the same time, the energy consumption increases slightly from 1.15 J to 1.25 J per second after sending an ICMP attack. The UDP flood attack causes an increase in energy consumption from 1.25 J to more than 1.50 J per second.

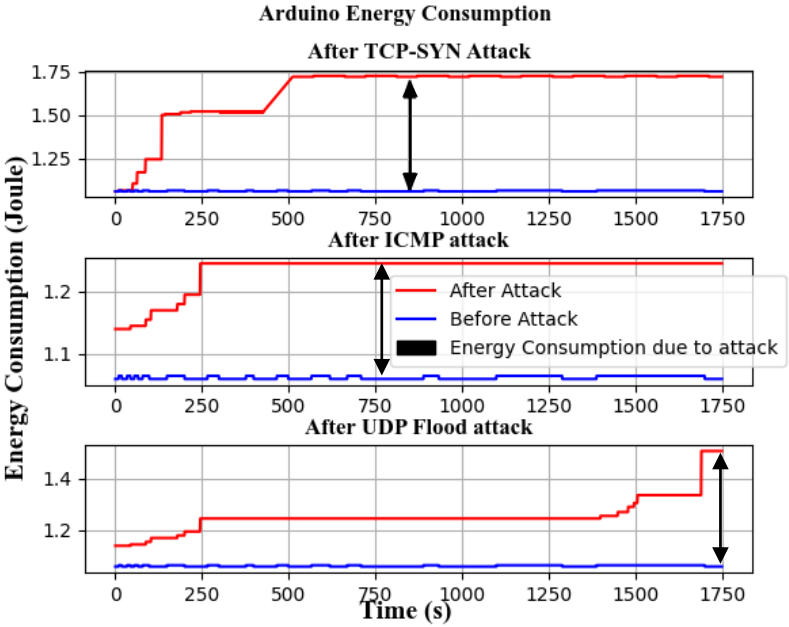

Figure 28. Arduino Energy Consumption under EC-DDoS Attacks.

It is important to note that the EC-DDoS attack rates employed by the attacker in this study remained below the threshold of DDoS attack rates that result in disconnection on smart healthcare devices. In the next section, we study the smart devices' energy consumption behaviour under F-APs attacks.

3.7.2 Energy Consumption and F-APs Attacks

Once the devices are disconnected from the legitimate AP, the F-APs attack takes over its responsibility to consume more energy and monitor them.

The signal of the F-APs is more vital to the victim's smart devices than the legitimate AP. When the devices are disconnected, the signal from the F-APs will be sent to the smart devices to force them to connect to affect their energy resources. Afterwards, the monitoring mode of the F-APs will be enabled to monitor packets transferred from and to the smart devices. At this stage, the sniffer is essential to launch further attacks on the target device and collect information about it, such as IP and port status. The F-AP is designed to be more flexible in sending malicious packets automatically to affect the energy resources once the smart devices are connected.

The required time for the Raspberry Pi to connect to the F-APs is between 3 and 5 minutes. While the Arduino takes 7 to 10 minutes, sometimes we force the Arduino to connect to the F-APs. **Figure 29** shows how the energy consumption of the Raspberry Pi changed

Algorithm 2 Fake Access Points Attack

procedure SMARTDEVICE, F-APS(a,b) ▷ *consume more energy of (a).*
  Sniff air for network scanning
  Measure energy consumption of SH
  if $a \subseteq b$ **then**
    if $a = connected$ **then** ▷ *a is connected to F-APs*
      Sniff air for network scanning
      Send malicious packets
      Calculate energy consumption after the attack
    else if $a = NotConnected$ **then**
      Try to reconnect it to F-APs
  while $a \not\subseteq b$ **do** ▷ *if there are no new devices*
    Sniff air for finding new devices

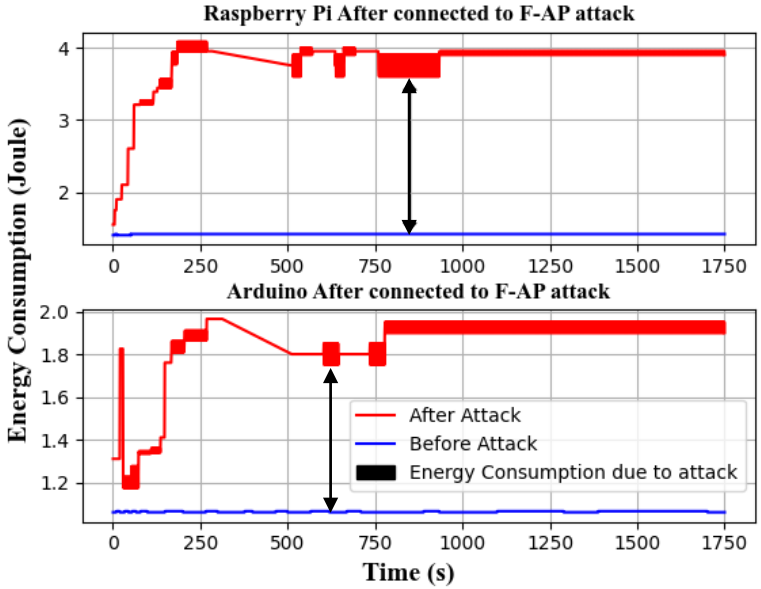

Figure 29. Raspberry Pi and Arduino Energy Consumption under F-APs Attack.

after connecting it to the F-APs; the malicious packets were randomly selected and sent to the Raspberry Pi. The energy consumption increases to more than 4.00 J per second. At the same time, the energy consumption of the Arduino increases slightly to reach more than 2.00 J per second after connecting it to the F-AP. Therefore, we can conclude that the F-APs

attack successfully affects smart healthcare devices' energy consumption.

3.7.3 Results and Discussion

In our experiment, we studied the effect of DDoS, EC-DDoS, and F-APs attacks against the Raspberry Pi and Arduino for about 30 to 60 minutes and measured the energy consumption. During such attacks, the smart devices continuously receive the packets and spend resources processing these packets. Our analysis shows that effective DDoS attacks can be

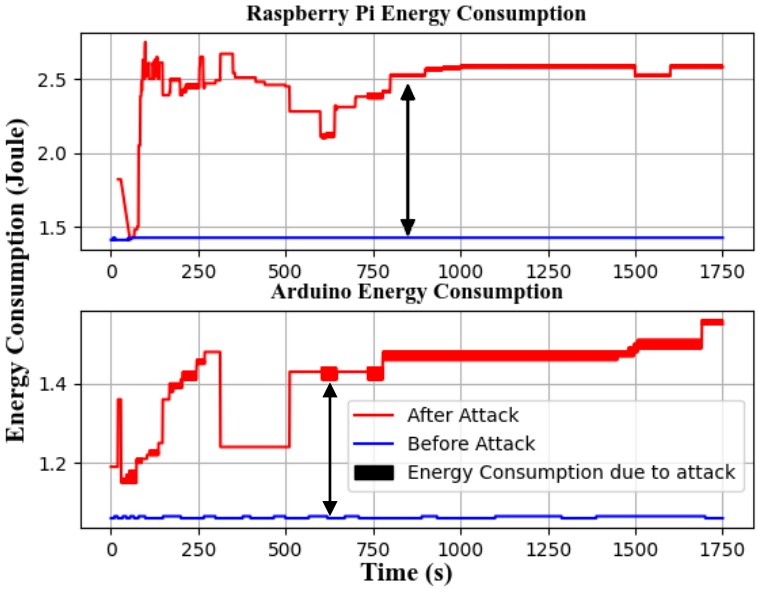

Figure 30. Raspberry Pi and Arduino Energy Consumption under Attacks where the F-APs affect 45% of the energy consumption of the Raspberry Pi and the Arduino, while the affection of EC-DDoS attack is about 55%.

launched at NP if the victim replies to ICMP packets. ICMP or TCP-SYN/UDP attacks could be used on open and closed ports. However, to launch EC-DDoS attacks that cost the victim device's maximum energy without being disconnected from the legitimate AP, the attacker can launch a PH TCP-SYN attack against open ports or ICMP attacks if the device responds to ICMP packets.

Moreover, to force the smart healthcare devices to connect to the F-AP attack, the signals of the latter should appear stronger to the victim than the legitimate APs. The attacker launches malicious attacks through the F-AP to induce maximal energy consumption without being disconnected by considering the threshold of the AR. Figure 30 shows the overall infection of EC-DDoS and F-APs attacks on both devices (Arduino and Raspberry Pi); as

it can be seen, the energy consumption of the Raspberry Pi device varies from 1.42 J to be more than 3 J per second. At the same time, the energy consumption of the Arduino varies from 1.06 J per second to more than 2 J per second. It is observed that DDoS, EC-DDoS, and F-APs attacks significantly impact the energy consumption of IoT devices. When an IoT device experiences a flood of TCP, UDP, and ICMP packets, it leads to significant increases in energy usage, which can ultimately result in the destruction of the IoT devices. This study aims to enhance the comprehension of energy consumption attacks caused by the combined impact of F-AP, DDoS, and EC-DDoS attacks on smart healthcare systems. The analysis of resource consumption in such scenarios provides valuable insights into the impact of DDoS and F-AP attacks on resource-constrained smart healthcare environments. Furthermore, it facilitates future research in developing lightweight defence mechanisms against these types of attacks. As such, the following equation represents the impact of the EC-DDoS attack on energy consumption:

$$ECDDOS = Enormal + \alpha \times P_{attack} \tag{3.2}$$

where $ECDDOS$ represents the energy consumption during a DDoS attack, $Enormal$ represents the energy consumption under normal operating conditions, P_{attack} represents the power consumed specifically due to the attack, and α is a scaling factor that quantifies the impact of the attack on energy consumption.

Chapter 4

Detection of Energy Consumption Cyber Attacks on Smart Devices

With the spread of IoT technologies, there is a growing concern about the security of smart home devices. Smart home devices suffer from resource-constrained problems, and these devices and sensors could be connected to unreliable and untrustworthy networks. Nevertheless, securing IoT technology is mandatory due to the relevant data handled by these devices. Preventing energy attacks and securing the IoT infrastructure is a crucial challenge in modern smart homes. One potential solution to address abnormal behaviour of IoT devices and detect IoT cyberattacks is through energy consumption monitoring.

Moreover, building a lightweight algorithm for securing IoT devices is essential to consider the limitations of its resources. This chapter presents a lightweight technique for detecting energy consumption attacks on smart home devices based on analyzing the received packets by the smart devices. The proposed algorithm considers three different protocols, TCP, UDP, and MQTT, and different device statuses, like *Idle*, *active*, and when it is under attack. Moreover, it considers the resource constraints of the smart devices for detecting abnormal behaviours and sending an alert to the administrator as soon as the attack is detected. The proposed approach effectively detects energy consumption attacks by measuring the packet reception rate of the smart devices for different protocols.

We have organized this chapter as follows: Section 4.1 provides detailed insights into the research problems, offering a clear understanding of the motivations and objectives driving the study. Section 4.2 serves as a general introduction to the chapter. Our proposal, including metrics definition, methodology, and the detection algorithm, is described in Section 4.3. In Section 4.4, we present the testbed scenario used to test the algorithm, along with the final results of detecting energy consumption attacks for different protocols using packet reception rate measurement. Finally, in Section 4.5, we present the results and discussions. The content of this chapter is mainly derived from [21].

4.1 Problem Statement, Motivation and Objectives

The problem addressed in this research is detecting the smart devices' resource constraints problems. This chapter targets the detection and mitigation of energy consumption attacks. We aim to provide a lightweight detection mechanism that considers the smart devices' resource constraints. This chapter provides a detection mechanism to detect energy attacks in smart devices. This study has been done in real smart IoT devices, and the final results show high efficiency in detecting energy consumption attacks in smart home devices.

Consequently, security is the main issue that restricts the adoption of IoT in social life. Many researchers have been working to make the IoT a more reliable and secure technology so that it can be adopted in society to make some aspects of human life more manageable and convenient. Since researchers develop many schemes and methods, but due to the constrained environment, e.g., low computational power and low energy of IoT, these techniques are not feasible. Therefore, an added line of protection that considers resource constraints should be built into IoT devices and networks to defend IoT-based organizations from cyber threats. Our main contribution is building a lightweight algorithm to detect energy consumption attacks in smart homes deployed directly at sensors. It applies real-time packet rate measurement to discriminate between smart devices' normal and abnormal packet reception rate behaviours. Therefore, normal behaviour is determined by evaluating the packet reception rate and energy consumption of smart devices during attack-free intervals. Conversely, any significant deviation from this established normal behaviour in the packet reception rate signifies abnormal behaviour. In this work, we consider three different protocols such as TCP, UDP, and MQTT. We also consider the different device statuses, such as *Idle*, *active*, and when it is under attack, to evaluate the best detection of energy consumption attack. We simulate the detection algorithm and assess the results by applying the proposed algorithm to the smart devices themselves, such as the Raspberry Pi[1]. We measure the current consumption of the smart device to monitor the energy while measuring the packet reception rate to discriminate between normal and abnormal behaviours. Therefore, this algorithm design is a protection strategy for IoT devices to maintain their integrity, seamlessly make them available to legitimate users, and protect them from energy consumption attacks by considering their resource constraints.

4.2 Introduction

The IoT can incorporate many heterogeneous devices such as cameras, smart meters [233], vehicles, and others transparently while providing open access to various data generated by such devices to provide new services to citizens and companies [234]. The IoT paradigm can be extremely massive and complex. It may contain tens of thousands of sensors, actuators, and gateways. Devices can communicate with gateways via different protocols, whereas gateways may connect with the internet and cloud-based apps via a similarly diverse range of protocols [235]. IoT technology's services find applications in many domains such as automotive, medical aids, smart grids, and many others [236]. The relevant data exchanged between smart IoT devices are more vulnerable to attacks since they are often deployed in a hostile and insecure environment [237].

[1]https://www.raspberrypi.com/documentation/

In this complex architecture, data can be processed by various heterogeneous entities. Data transmission, security, and integrity are key aspects to be considered. As a result, protocols and technologies are required to provide data security, access management, and flow data transmission [238]. Many recent studies have been conducted to cope with security issues in the IoT paradigm [239] [240]. Some of these studies concentrate on security issues at a particular layer, whereas other approaches aim at providing end-to-end security [241]. Several methods and protocols have been suggested, primarily concerned with reducing energy consumption and increasing the network lifetime [242] [243]. Therefore, security solutions are mandatory to protect IoT devices from intruder attacks. This paper aims to secure low-resource IoT devices, such as smart home devices, against energy consumption attacks [244]. In smart homes, detecting energy consumption attacks is required to protect the energy from vulnerability threats that could access the home network and attack the smart devices. Monitoring the energy consumption of IoT devices provides a potential method for identifying devices engaged in energy-intensive attacks. Furthermore, an energy consumption analysis-based approach offers increased security, particularly in scenarios where the device's kernel has already been compromised. Once a device has been compromised, ensuring data integrity becomes challenging, and its trustworthiness cannot be guaranteed [209] [203]. One of the most significant areas of study today revolves around the efficient utilization of energy resources. Approximately one-third of the total energy consumption is attributed to specific losses, where energy is unintentionally consumed [245]. Moreover, there is an anticipated increase in energy consumption in the future. Growing awareness of the importance of energy conservation and efficiency has also contributed to the development of the modern smart home concept [203]. Initially, the concept of a smart home focused on connecting sensors and devices over a network to enable remote access, monitoring, and control of the living environment, providing convenience to users. However, in the current stage, it also encompasses optimizing energy usage in buildings and addressing malware and IoT cyber-attack detection within smart home infrastructures. Monitoring the energy consumption of IoT devices serves as a potential method for detecting attacks that require significant energy consumption [243], such as DDoS attacks [246] and crypto-mining attacks [203].

In this chapter, we build a lightweight algorithm that considers the resource constraints for smart devices to detect energy consumption attacks. The algorithm is used to monitor the packet reception rate of the smart devices on different protocols. In this algorithm, we used the following protocols: TCP, UDP, and MQTT, as they are popular protocols used nowadays with IoT systems [247, 248]. We also consider different devices' statuses, such as *Idle*, *active*, and when they could be under attack. The algorithm automatically fetches the packets' reception rate and divides them into different behaviours, such as normal and abnormal, depending on the presence and absence of the energy consumption attacks. At the same time, the energy consumption of the smart devices is measured to determine the packet reception rate's behaviour and to specify whether the packet reception rate's behaviour is normal or abnormal. This algorithm successfully detected energy consumption attacks in smart home devices with a cost-efficient experimental setup.

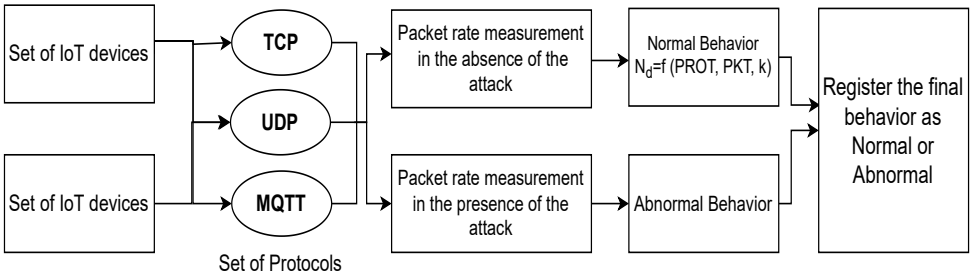

Figure 31. Packet Reception Rate measurement in the absence and presence of the attack.

4.3 Packet Monitoring Mechanisms

4.3.1 Proposed Algorithm

In this section, we present the algorithm to detect energy consumption attacks in smart home devices by monitoring the packet rate received by the smart devices. The algorithm considers different protocols like TCP, UDP, and MQTT and different device statuses such as *active*, *Idle*, and *under attack*. The proposed algorithm is depicted in Figure 61.

The algorithm has three phases, 1) *collecting phase* where the algorithm collects samples of the number of received packets for different statuses when the device is Idle, active, or under attack and divides the collected packets from different protocols, e.g., TCP, UDP, and MQTT, into normal or abnormal behaviours; 2) *calculating phase*, which calculates the collected samples and compares the final results of the fetching packets with the energy measurements to determine whether the state of packets measurements is caused by an energy consumption attack, then divides the final results into (normal, or abnormal behaviour); and 3) *detection phase* where the algorithm applies different conditions to classify if there is an energy consumption attack or not. We build the algorithm inside the Python scripts to automatically fetch packets and analyse normal and abnormal behaviours.

In the detection stage of the proposed technique, the packet reception rate of IoT devices for different protocols is measured and analyzed. If the IoT device has abnormally high received packets, it may have carried out an energy consumption attack. Therefore, smart devices should stop listening to the received packets of such a port. Simultaneously, there should be a counter (x) on the total time that the smart device stops listening; if it exceeds (x) times, then the algorithm should register it as abnormal behaviour. Our algorithm considers the ($x = 3$) times. However, in our specific scenario, we chose 3 times as a threshold to discern whether the rise in packet reception rate is due to an attack or simply normal behaviour by checking the energy measurement of the smart device.

Algorithm 3 A Technique to Detect Energy Consumption Attack

```
Input: PROT, PKT, d
Output: Normal or Abnormal
if N(d) = f(PROT, PKT, k) then
    return to monitor packet rate
else
    Make the device stop licensing for x time
    counter = counter + 1
    if counter > 3 then
        registers the device as having abnormal behaviour
        check energy consumption
    else
        return to monitor packet rate
```

4.3.2 Packet Measurements

To effectively build a technique to detect energy consumption attacks in IoT systems, it is necessary to take into account the different protocols used for different IoT devices. This algorithm considers three different protocols, e.g., TCP, UDP, and MQTT. Also, it considers different device statuses, e.g., *Idle*, *active*, and when it is under attack.

With the aim of IoT energy consumption attack detection at the learning stage, the packet reception rate of each IoT device in the IoT network in the absence or the presence of an IoT energy consumption attack is measured at a specific interval and at equal sub-intervals of time. Based on these measurements, the number of normal $N(d)$ received packets of IoT devices are constructed, part of them labelled as *normal behaviour* and entered into the database (DB) to deal with them later on.

Let us describe the normal (N) packet reception rate measurement in the absence of energy consumption attacks.

$$N(d) = f(PROT, PKT, k) \quad \text{and} \quad k \in [0, 1] \tag{4.1}$$

The expression $N(d) = f(PROT, PKT, k)$ represents a function f that calculates the normal behavior $N(d)$ based on following inputs:

Where $N(d)$ is the normalized receiving packets of an IoT device (d), where d is a certain smart home device, (PKT) represents the received packets at a point in time in the absence of an energy consumption attack for a specific protocol ($PROT$), and K is the number of packet measurements within a specific time, $n(d) = f(PROT, PKT, k) \in [0, 1]$ where 0 is the minimum received packets, and 1 is the maximum received packets by the smart devices for a specific protocol. Therefore, the function f takes these parameters $(PROT, PKT, k)$ and produces a numerical output $N(d)$ and sets the final decision of this output to normal or abnormal decision.

4.3.3 Energy Measurements

With the aim of IoT energy consumption attack detection at the learning stage, the energy consumption measurements of each IoT device in the IoT network in the presence or the absence of IoT energy consumption attacks are measured at a specific interval and at equal sub-intervals of time. Based on these measurements, the number of received packets of IoT devices on $N(d)$ are constructed, part of them labelled as *normal behaviour* and others as *abnormal behaviour* and entered into the DB to deal with them later on. The energy consumption measurement is essential at the first stage as it is used to determine the behaviour of the packet reception rate as normal or abnormal.

To achieve this objective, we infected IoT devices with malicious attacks capable of executing IoT energy consumption attacks, e.g., flooding attacks. Subsequently, we measured the energy consumption of each IoT device under different statuses, both in the presence and absence of IoT cyberattacks, at specific intervals and equal sub-intervals of time. In this experiment, we designed a smart circuit using a non-invasive current sensor [2] with Arduino, capacitors, and other resistors to measure the current consumption of smart home devices. This smart circuit samples voltage, ampere, watt, and current per second. In our experiment, we use the Joule (J) values to calculate the energy consumption of smart devices, as shown in Figure 34.

Let us describe the energy (E) measurement footprints considering the set of different device statuses in the absence or the presence of the attack.

$$E(d) = f(e(d), PROT, k) \quad \text{and} \quad k \in [0, 1] \tag{4.2}$$

The expression $E(d) = f(e(d), PROT, k)$ denotes a function f that calculates the energy consumption $E(d)$ of a smart device (d). Here are the components of this expression:

Where (e_d) the energy measurement (e) of the smart device (d) at a point in time in the absence or presence of cyberattacks for a specific protocol ($PROT$), and K is the number of energy measurements in a time interval, $f(e(d), k) \in [0, 1]$ where 0 is the minimum energy consumption measurement, and 1 presents the maximum energy consumption measurement in the absence or presence of the attack. In essence, the function f takes into account the current energy consumption $e(d)$ and the used protocol $PROT$ to compute the overall energy consumption $E(d)$ of the smart device (d). Therefore, we calculated the energy consumption for every 3 minute for a specific smart device; the time for each energy consumption measurement is also registered and entered into the DB. It's important to note that we set the measurement of the energy to 3 minutes, but it also could take on any value, and these parameters could adapt to the particular scenario being evaluated. However, in our specific scenario, we set it to 3 minutes to optimize resource usage, considering the hardware capabilities and network bandwidth. This parameter is highly adjustable based on the system's capabilities. We opted for 3 minutes in this experimental setup to streamline resource usage. The measurements collected over this duration are sent in a single request to the DB, effectively conserving resources.

[2] https://tinyurl.com/mrxyvr46

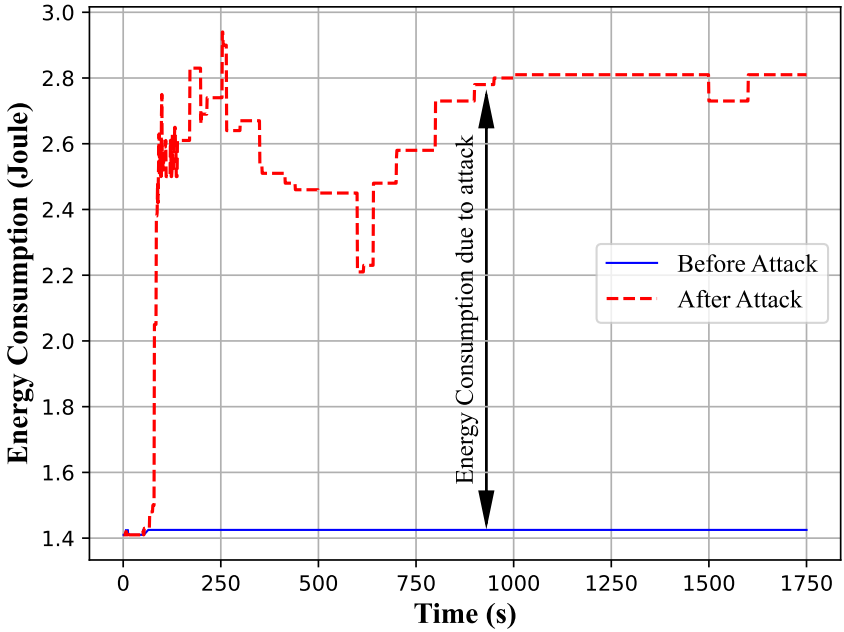

Figure 32. Energy consumption measurement of normal and abnormal behaviours of the Raspberry Pi device.

4.3.4 Calculation of normal and abnormal behaviours

In order to calculate the packet reception rate for each IoT device in normal and abnormal cases, we have divided the code into different parts:1) The first part is to fetch the packet reception rate for each protocol separately, depending on the set of protocols used in our system, 2) We measure the packet reception as shown in Equation 4.1 for the active smart devices with the absence of the attack and for each protocol separately and register the final results as normal behaviours, 3) Then, we measure the current consumption of the smart device in the case of normal behaviour as shown in Equation 4.2 and monitor the packet reception rate with the energy consumption when the status of the smart device is *On* with the absence of the attack. The monitoring mode continuously fetches the packet, calculates energy for about 30 minutes, and stores the final results for every 3 minutes in the DB, 4) For calculating abnormal behaviours, we send malicious attacks to consume energy for about 30 minutes to the active smart devices. At this time, we start calculating the energy consumption and the packet reception rate for each protocol separately. Then, we compare the final results with the normal behaviours of such a device. In case of abnormal behaviour, we store the final result for every 3 minutes in the DB as abnormal behaviours, 5) For printing the

final results and displaying the normal with the abnormal behaviours, we fetch the stored data from the DB and start calculating the normal with abnormal behaviours, 6) In case there is abnormal behaviour with fetching the packets compared to normal behaviours, we notify the system administrator to register the entire case as abnormal behaviour.

4.4 Implementation and Analysis

4.4.1 Experimentation and Discussion

In this section, we describe the testbed scenario that we used to test the algorithm. Also, we show the final results of detecting energy consumption attacks for different protocols using packet reception rate measurement.

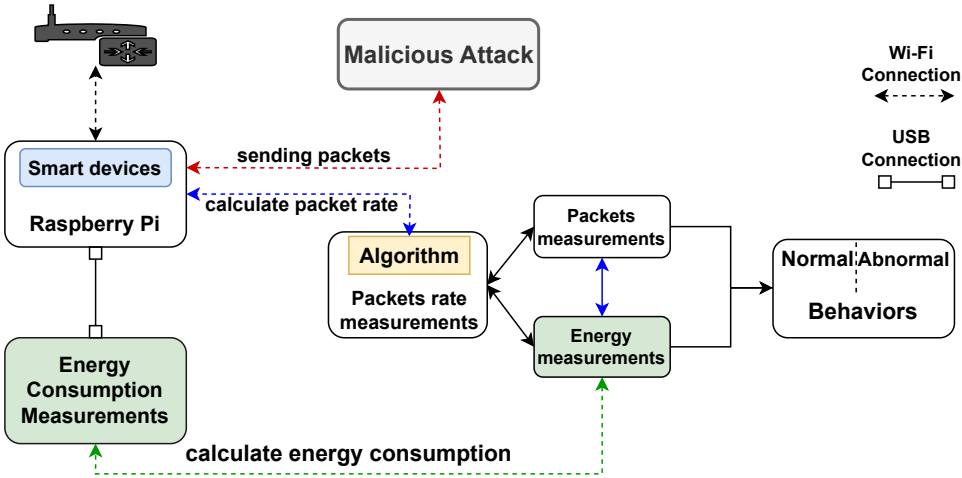

Figure 33. Testing Environment.

The Testbed Scenario

We used Raspberry Pi as a smart home device in this experiment. We used different software tools for attacking data generation and collection. On the adversary side, we used *Nmap*[3] to launch a network scan and identify devices' status, such as *online* or *offline*, IP address, and MAC address. Different tools are used to generate malicious attacks on the victim side, such as *hping3*[4]. We designed a smart circuit using a non-invasive current sensor with Arduino, capacitors, and other resistors to measure the current consumption of smart home devices. This smart circuit samples voltage, ampere, watt, and current per second. In

[3]https://nmap.org/
[4]https://www.kali.org/tools/hping3/

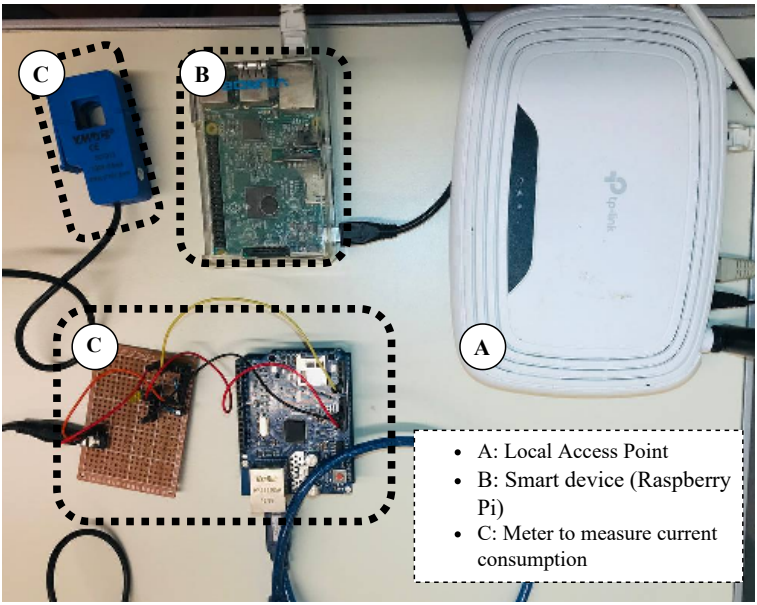

Figure 34. Testbed scenario showing the devices used in our experiment and the sensor used to measure the energy consumption.

our experiment, we use the Joule (J) values to calculate the energy consumption of smart devices.

We analyze the packet rate received by the smart home device to detect energy consumption attacks. We built a program using *pyshark*[5] to sniff and fetch packets automatically and store the final results in the DB[6].

Table 7. Packets analysis depends on protocol type and energy consumption.

PROT	Normal Behavior		Abnormal Behavior	
	Packet	E [J]	Packet	E [J]
TCP	$2000 \div 6000$	≤ 1.42	> 6000	> 1.42
UDP	$2000 \div 6000$	≤ 1.42	> 6000	> 1.42
MQTT	$2000 \div 6000$	≤ 1.42	> 6000	> 1.42

The total average received packets by the smart devices is calculated by estimating the average rate of the received packets in 30 minutes compared to the abnormal behaviour.

[5]https://pypi.org/project/pyshark/
[6]https://github.com/developerZA/ATechniuqeToDetectEnergy.git

We divided the packet reception rate into different slots. For every 3 minute, we calculated the average of the received packets in the absence of the attack and stored the final results in the DB as normal behaviour. The same calculation is applied to the smart home device when it is under attack. Then, the final results are stored in the DB for further calculations. The detection system keeps monitoring the received packets, and in case there are abnormal behaviours received by such a device, we register that case as abnormal behaviour.

To calculate the average received packets by the smart devices of the TCP protocol. We analyzed all the received packets and divided them into different types, such as packets received, re-transmission, and acknowledged, as shown in Figure 36. However, Figure 1(a) in 35 illustrates an attacker targeting smart devices through the access point. The primary function of TCP/IP is to ensure reliable data transmission between two hosts, typically a receiving smart device and a transmitting attacker. In this experiment, we launched a TCP/SYN attack, causing a flood of packets towards the victim devices to consume more resources.

In our experiment, we need the average of the received packet by the smart devices that cause an increase in energy consumption. Then, we used the final calculation to detect energy consumption attacks.

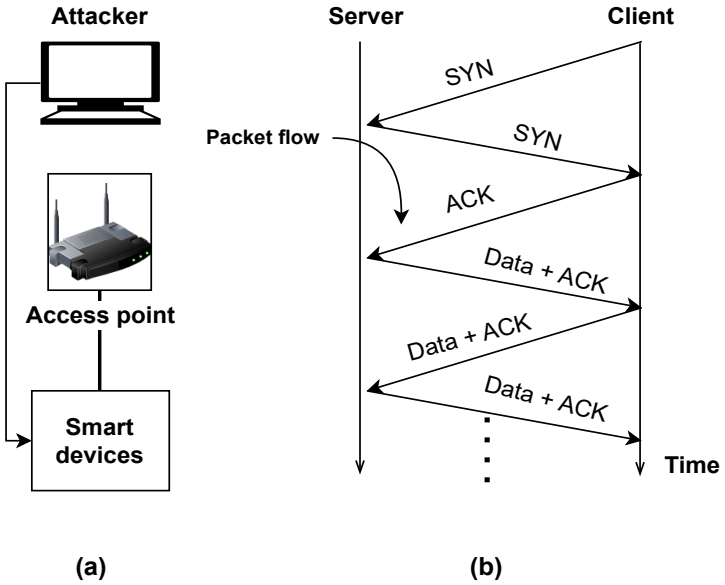

Figure 35. (a) Network with an attacker and smart device and (b) TCP/IP connection timing diagram

Through the 30 minutes in the absence or the presence of the attack, we study the received packets by the smart devices for different protocols such as TCP, UDP, and MQTT. Also, we study the total number of times the smart devices stopped listening to understand if energy consumption attacks source the received packets. The normal average of the re-

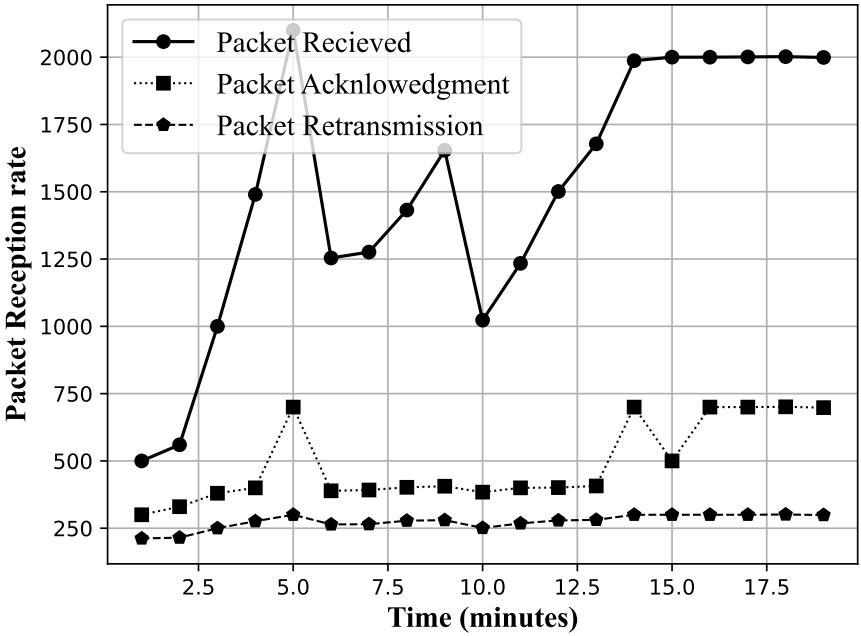

Figure 36. Packet received, re-transmission, and acknowledged for the TCP protocol.

ceived packets for TCP protocol in 30 minutes fluctuates between 2 k and 5 k packets, as shown in Figure 37.

In this experiment, for every 3 minute, we calculated the normal and abnormal behaviour. So, for the first 3 minutes, the normal behaviour of the received packet is less than 5 k packets, while the abnormal received packets in the first 3 minutes in the presence of the attack are more than 6 k packets. The detection system registers the first case of the first 3 minutes as abnormal behaviour. We also calculate the normal and abnormal behaviours for the total of 30 minutes by calculating the average of the packet reception rates of the normal behaviours and comparing it with the average of the abnormal behaviours to register the entire case of the 30 minutes as normal or abnormal behaviour.

In the case of UDP protocol, it is not easy to customize the actual receiving packet as the state of such a port cannot be confirmed by network scan using Nmap[7] because the port does not send any response. So, in our calculation, as shown in Figure 38, we calculate the normally received packets of UDP protocol by the smart device. We monitor the packet reception rate of the smart devices for 30 minutes to check the normal and abnormal receiving packets of the Raspberry Pi. The normal behaviour of the receiving packets is between 1 k

[7]https://nmap.org/

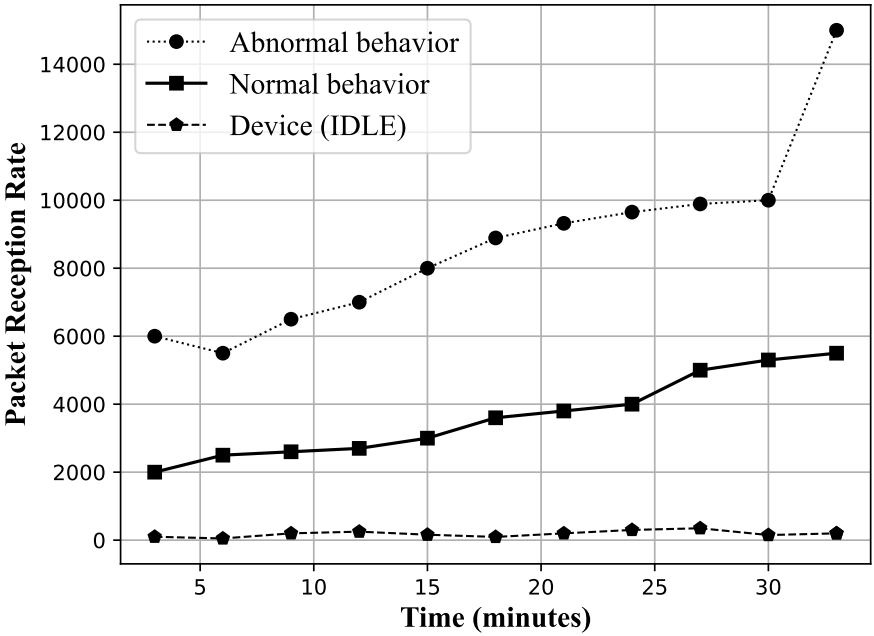

Figure 37. Packet reception rate of normal and abnormal behaviours of the TCP protocol.

and 3 k packets. In contrast, the abnormal behaviour of the received packets by the smart device is between 9 k and more than 12 k packets. Figure 39 shows the behaviour of the subscribed packet's rate of the MQTT protocol. We study different behaviours of this protocol by registering the number of published and subscribed packets of the smart home device. To detect an energy consumption attack in the case of the MQTT protocol, we consider the number of subscribed packets as they affect the energy resources of the smart devices. Therefore, the normal behaviours of the MQTT protocol are registered to be less than 6 k packets, while the abnormal behaviours reached more than 8 k packets. In this algorithm, we also consider the case where we do not have to specify the protocol by calculating the average received packets for all the used protocols. We find that the normal behaviour of the packet reception rate of the Raspberry Pi is between 1500 packets and less than or equal to 6 k packets. The abnormal behaviour of the total received packets is between 7 k and more than 12 k packets, as shown in Figure 40.

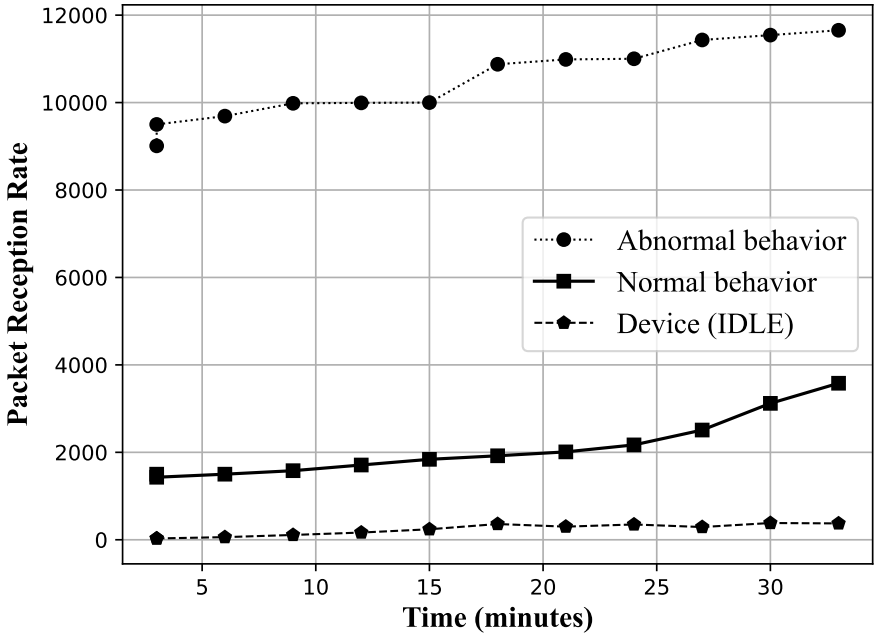

Figure 38. Packet reception rate of normal and abnormal behaviours of the UDP protocol.

4.5 Results and Analysis

In this experiment, an IoT device was deliberately infected with malicious software to conduct various flooding attacks on a target within an isolated network. The energy consumption footprints and packet reception rate measurements of these IoT devices were recorded during the experiments, both in normal operating conditions and when the device was subjected to malicious attacks. Each energy consumption footprint and packet reception rate measurement were taken at intervals of 5 seconds over a period of 3 minutes, capturing the device's behaviour during both attack and normal operation scenarios. A total of 30 minutes of calculation measurement of received packets and the energy consumption footprints of both in the presence of attacks and normal functioning smart devices were built. It's important to note that 30 minutes of measurement could take on any value, and these parameters could adapt to the particular scenario being evaluated. However, in our specific scenario, we set it to 3 minutes to optimize resource usage, considering the hardware capabilities and network bandwidth. This parameter is highly adjustable based on the system's capabilities. We opted for 3 minutes in this experimental setup to streamline resource us-

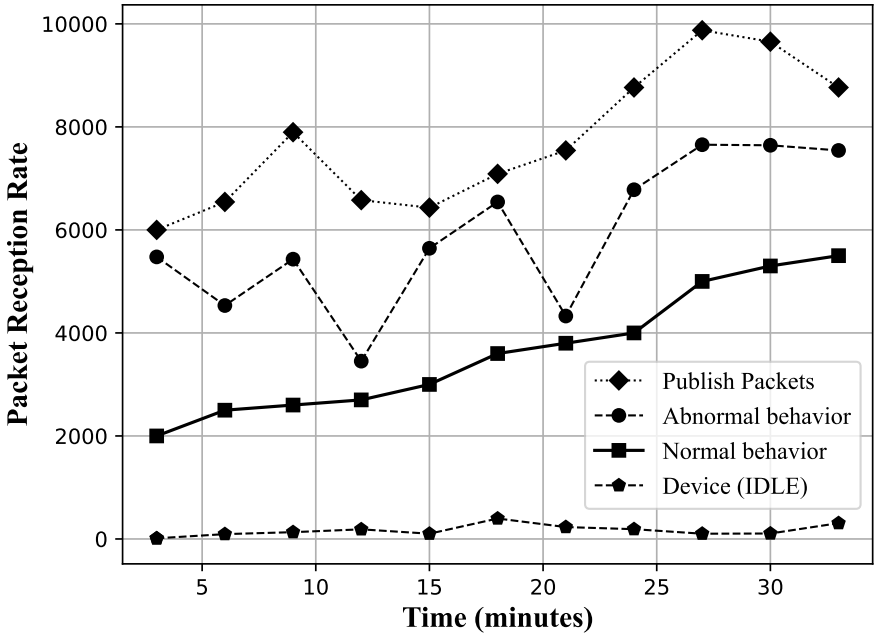

Figure 39. Packet subscribed rate of normal and abnormal behaviours of the MQTT protocol.

age. The measurements collected over this duration are sent in a single request to the DB, effectively conserving resources.

The results of this experiment showed high efficiency of energy consumption attack detection based on the packet reception rate analysis. At the same time, the analysis of the packet reception rate for different protocols was considered. As it can be seen from Figure 40, the abnormal behaviour registered once the packets reached more than 6 k packets for different protocols. This analysis is done for different types of protocols and different devices' statuses.

This experiment shows high efficiency in detecting energy consumption attacks as it is not expensive to implement in a smart home device and considers the smart device's resource constraint. Compared to calculating the energy consumption of the devices for detecting energy consumption attacks in smart homes. Additionally, the energy consumption during the detection of the energy attack fluctuates between 1.2 J and 1.5 J, indicating a high efficiency in energy utilization during the attack detection.

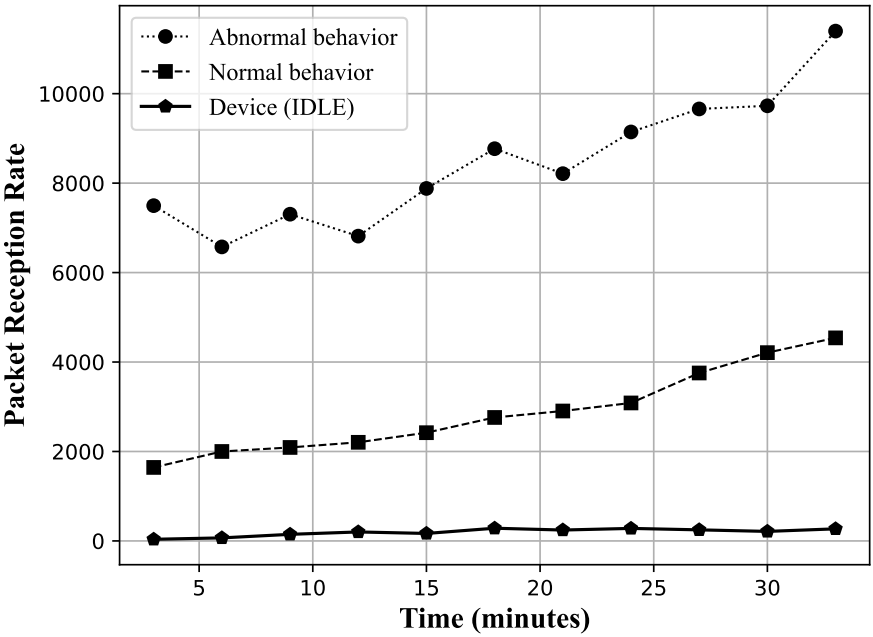

Figure 40. General cases (Normal behaviour Vs Abnormal behaviour) for TCP, UDP, and MQTT altogether, where the effect of each protocol in normal behaviour is as follows: TCP effect is about 45 %, and UDP affect about 30 %, and the MQTT effect is about 20 %. While the impact of TCP is about 40 %, MQTT is about 40 %, and 20 % of UDP is in the presence of the attack.

Chapter 5

Mitigating and Analysis of Memory Usage Attack in IoE system

Marketing forces toward smart homes are also accelerating the spread of IoE devices in households. An obvious risk of the rapid adoption of these smart devices is that many lack controls for protecting the privacy and security of end users from attacks designed to disrupt lives and incur financial losses. In today's context, the smart home system encompasses the management of essential life support processes in various settings, ranging from small-scale systems like commercial offices, apartments, and cottages to larger, highly automated complexes such as commercial and industrial establishments. One of the critical tasks to be solved by the concept of a modern smart home is the problem of preventing the usage of IoE resources. Recently, there has been a rapid increase in attacks on consumer IoE devices.

Memory corruption vulnerabilities constitute a significant class of vulnerabilities in software security through which attackers can gain control of an entire system. Numerous memory corruption vulnerabilities have been found in IoE firmware already deployed in the consumer market. The objective of this chapter is to analyze and provide an in-depth explanation of resource usage attacks. To facilitate the dynamic analysis of these attacks, a cost-effective simulation environment is developed. Additionally, controlled resource usage attacks are conducted on resource-constrained victim IoE devices, allowing for the monitoring of their resource consumption, including CPU and memory utilization. We also build a lightweight algorithm to detect memory usage attacks in the IoE environment. The result shows high efficiency in detecting and mitigating memory usage attacks by detecting when the intruder starts and stops the attack.

We have structured this chapter as follows: Section 5.1 outlines the main motivation and objectives of this chapter. Section 5.2 provides a general introduction to the chapter. In Sections 5.3 and 5.4, we offer a comprehensive description of the threat and testbed scenarios. Section 5.5 presents a static analysis of resource usage attacks, showcasing both normal and

abnormal memory and CPU usage of the smart devices. Next, in Section 5.6, we present our proposal, which includes metrics definition, methodology, and the detection algorithm. Finally, we present the results and discussions in Section 5.7. The content of this chapter heavily draws from [22].

5.1 Motivation and Objectives

IoE is a fast-growing field with capabilities to revolutionize the whole industry. As per market trends, more than 20 billion of smart devices will be deployed in the next five years [249]. These interconnected devices will be generating sensitive data which needs to be protected. The field of IoE is making leaps and bounds technologically. There are multiple limitations while deploying IoE devices daily, e.g., battery life and lightweight computation. Therefore, building a novel security mechanism aims to protect the functionalities and privacy of sensitive IoE network environments, including healthcare, smart cities, etc. However, due to the substantial number of nodes in the environment and their restricted computing capabilities, securing smart nodes in the IoE environment is essential to protect the data and make the devices available to end-users. A lightweight mitigation technique should be considered to protect smart devices from resource-constraint attacks such as DoS, DDoS, and other malicious attacks. Our main contribution is building a lightweight technique to detect memory usage attacks in smart devices deployed directly at sensors. It applies real-time memory usage calculation to discriminate between different memory usage, e.g., read/write to memory. In this work, we consider different behaviours on the memory of smart devices. We measure the memory usage when there is read and write, under or without the attack, to evaluate the best detection of memory usage attack. We simulate the mitigation technique and assess the results by applying the proposed technique to smart devices, such as the Raspberry Pi[1] and Arduino. We measure the current memory usage of the smart device to monitor the memory usage to discriminate between normal and abnormal behaviours. Therefore, this algorithm design is a protection strategy for IoE devices to maintain their integrity, seamlessly make them available to legitimate users, and protect them from memory attacks by considering their resource constraints.

5.2 Introduction

IoE is an extension of IoT, which aims to connect network devices with specialized sensors or actuators through the Internet [250]. These sensors and actuators enable the detection and response to environmental changes such as light, temperature, sound, and vibration. By incorporating additional components, IoE significantly expands the capabilities of IoT, providing enhanced experiences for businesses, individuals, and countries. Unlike traditional IoT, IoE leverages data and processes to create more meaningful and valuable interactions with the environment[251][252], as depicted in Figure 41. The ultimate goal of IoE is to boost operational efficiency, offer new business opportunities, and improve the quality of our lives. Better to relate to this idea; take the scenario of a person uncertain about closing a gas valve

[1]https://www.raspberrypi.com/documentation/

at home. An IoE solution allows a user to automatically check the gas valve's status and close it remotely if necessary [212] [253].

Despite the potential benefits of IoE, it also brings significant security risks to its users. With the increasing number of IoE devices and their growing importance in our daily lives, the connection between the physical world and cyberspace introduced by IoE amplifies the vulnerability of smart devices to cyber attacks. Attacks on IoE systems can directly impact the well-being and safety of end users, as exemplified by the potential threat of an attacker intentionally causing a gas leak in a gas value scenario [254] [255]. The lack of

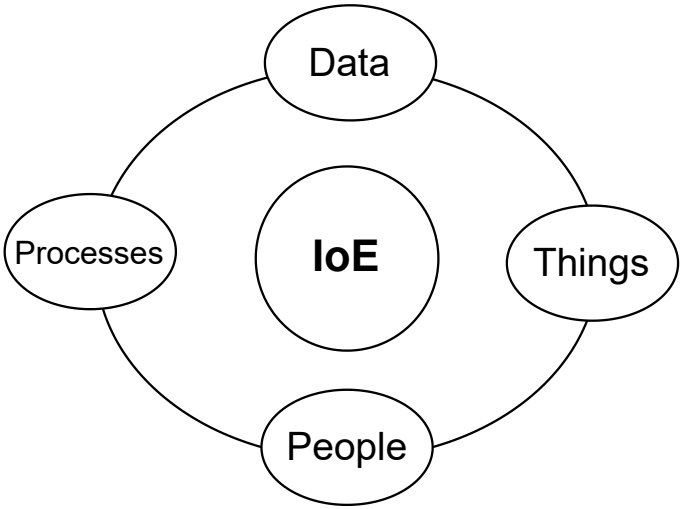

Figure 41. The Definition of Internet of Everything (IoE).

awareness regarding the quantity and characteristics of IoE devices around us is a cause for concern, especially considering the potential security risks they pose. Recent security incidents have highlighted the vulnerabilities of IoE systems. One notable incident is the DDoS attack against Dyn in 2016, which involved the Mirai botnet comprising around $100,000$ IoE hosts such as cameras and routers. The attack on Dyn's DNS caused a widespread outage of major websites like Netflix and CNN [256] [257]. Furthermore, attacks targeting the resources of sensors and actuators can render smart devices inaccessible to end-users. These incidents underscore the importance of addressing security challenges in the IoE landscape.

In light of these evolving threats, it is crucial to increase awareness of the potential security risks associated with IoE [258]. This can be achieved through systematic risk assessments and the use of effective visualizations to educate end users. Home users, in particular, are at a higher vulnerability level as they are increasingly surrounded by IoE devices such as hands-free speakers, baby monitors, and security cameras. However, they often lack the necessary resources and expertise to identify and address IoE-related threats, leaving them exposed to potential security risks. It is therefore important to empower home users with the knowledge and tools to remediate these risks and minimize their impact [252]. There-

fore, in this chapter, we mainly focus on analyzing the memory usage attack in smart devices and mitigating the effect of this attack by building a lightweight algorithm to detect memory usage attacks by calculating the memory usage of the smart device.

To accomplish this goal in home networks, we first identify memory usage attacks in smart home devices. Next, we analyze the effect of the attack by sending malicious attacks to affect the resources of the smart devices and calculate its effect on memory usage. We then elicit and document threats in the form of threat scenarios. Once specified, we build a lightweight algorithm to detect and mitigate the effect of the attack on memory usage.

5.3 Testbed Scenario

We used Raspberry Pi and Arduino as smart home devices in this experiment. We used different software tools for attacking data generation and collection. On the adversary side, we used *Nmap*[2] to launch a network scan and identify devices' status, such as online or offline, IP address, and MAC address. Different tools can generate malicious attacks on the victim side, such as *hping3*[3]. We used *tshark* tool [4] to evaluate the impact of memory usage attacks on smart devices and capture WiFi traffic.

We also created a module inside the smart device to monitor memory usage and register all memory behaviours in the DB. The monitoring mode registers the behaviour of the smart devices once it is *Idle*, *active*, and *under attack*. Different stages are used to run our experiment. In the first stage, we monitor the memory usage once the device is Idle, Active, and under attack. Then, we run a network scan to capture the port and device status. Once we ensure that the device is connected to the Internet, we send memory usage attacks for two purposes: first, to affect the memory, and second, to consume more memory usage and study the behaviour of the attack. Then, we run memory usage monitoring to calculate the memory usage of the devices and study the devices' behaviours before and after the attack.

5.4 Threat Scenarios and Threat Model

This section provides a brief overview of the design of memory usage attacks on IoE smart devices. The objective of a memory usage attack is to disrupt the functioning of a smart device by launching malicious attacks, such as DoS or DDoS attacks, specifically targeting the device's memory. This type of attack focuses on vulnerable IoE and embedded devices, which often lack robust built-in security protections and face resource-constraint challenges.

5.4.1 Threat Scenario

The smart devices of IoE suffer from low computation problems such as low energy and memory. The resource-constraints problems encourage attackers to attack these devices by flooding the smart devices with malicious attacks. In this chapter, we consider a scenario

[2]https://nmap.org/
[3]https://www.kali.org/tools/hping3/
[4]https://www.wireshark.org/

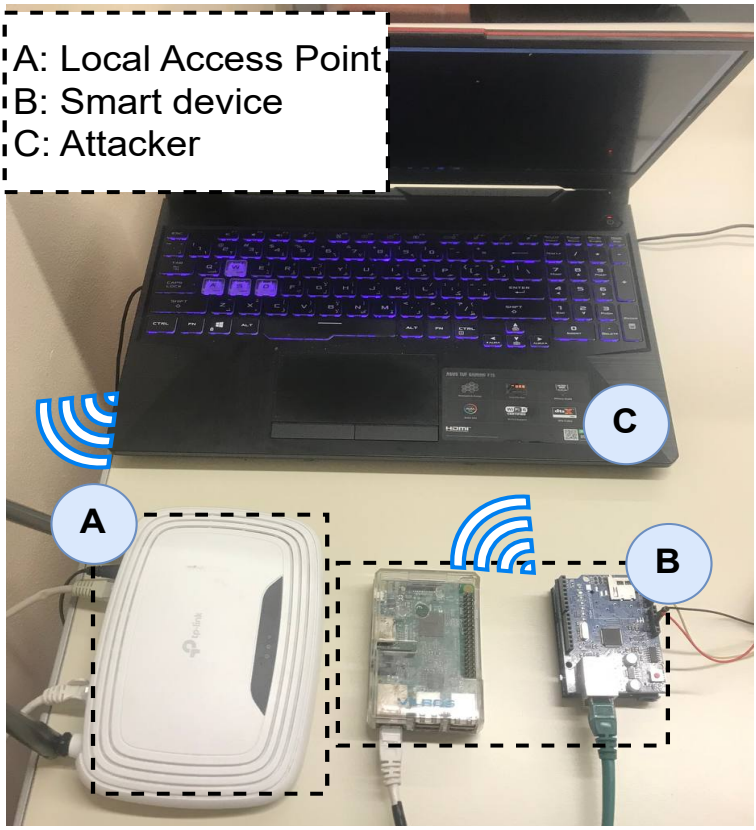

Figure 42. Testbed scenario showing the devices used in our experiment.

where the attacker has infiltrated the control network and possesses the ability to communicate with the smart devices. This can occur either through insider threats, where individuals with authorized access abuse their privileges, or through external hackers. Once inside the control network, the attacker has a plethora of attack options at their disposal, including malicious attacks. The focus of this chapter is to examine memory usage attacks specifically targeting smart devices in IoE systems.

The threat scenario used in our experiment was first to scan the network and get different information about the port and devices' status. For scanning the network, we install *Nmap* on Kali-Linux. In this scenario, the attacker can send a malicious attack to the smart device to affect its resources in terms of memory. The IP address, port, and device status are stored in the DB for further calculation. After scanning the network, we start the monitoring mode of the smart device's memory usage once the device is Idle and active, and when we send a malicious attack using *hping3* tools to the smart device. In this case, we study the memory behaviour before and after attacking the smart devices. We also store all informa-

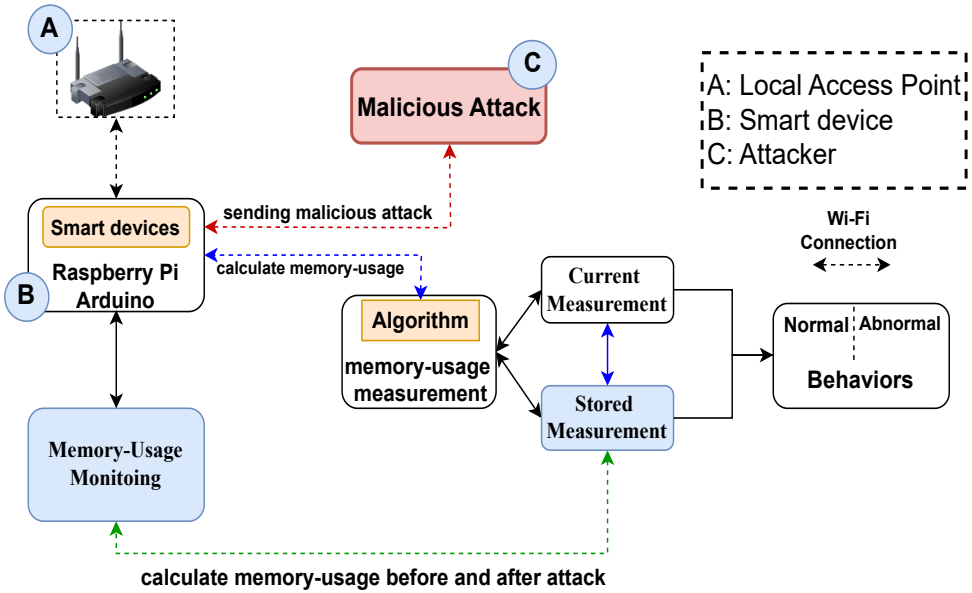

Figure 43. Testing Environment.

tion about the memory, such as memory in total, memory usage, and CPU usage before and after attacking the smart devices.

In particular, the source code consists of three different parts:

- Memory usage attack: This module commences with the DoS and DDoS attack to send malicious attacks to the smart devices and affect their memory.
- Scanner: This module scans the network and gets different information about the smart devices. Also, it sends the IP address of the attacked smart devices for further calculation
- Memory-monitoring-mode: This module monitors the memory usage of the smart devices when it is Idle, active, and under attack. The monitoring mode helps to register different memory behaviours for detecting such attacks.

5.4.2 Threat Model

This chapter presents a model for attacks on the memory usage of smart devices in IoE systems. The model provides a clear understanding of the possible attack vectors. Additionally, a lightweight algorithm [5] is developed for detecting these attacks on smart devices. In this model, we denote the attacker as ATK, the smart devices as d, and the memory usage

[5]https://github.com/developerZA/MitigationMemoryAttack.git

as MEM. Each attack originates from the attacker ATK and targets a specific smart device d. This relationship can be represented as follows:

$$ATK \mapsto mem \rightarrow D \tag{5.1}$$

where $atk \in ATK, d \subset D, mem \in MEM$. The notation $\mapsto$ maps the attacker (ATK) to the victim's (D) memory (mem). For calculating the memory usage and CPU usage of the smart

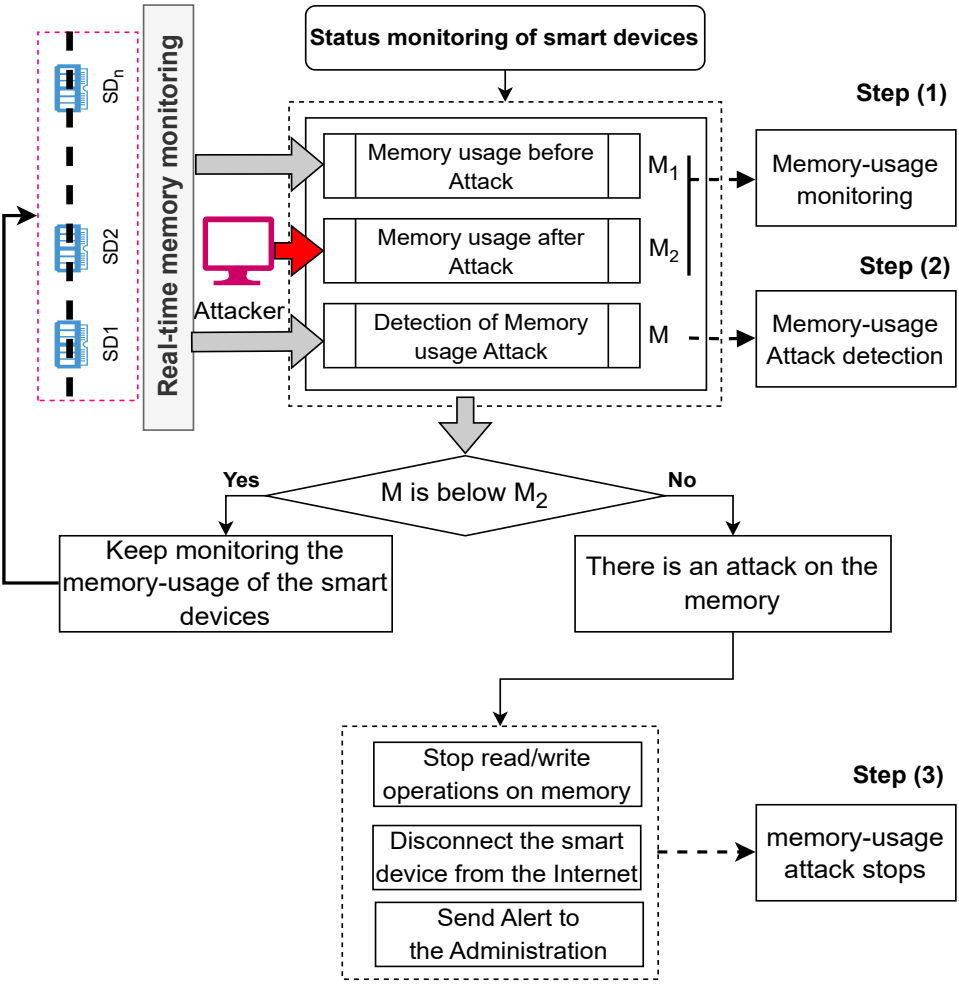

Figure 44. Schematic diagram of the proposed method.

devices before and after attacking the memory, the following equation math represents this calculation. Let us describe the memory usage measurement (MEM) footprints considering

the set of different device statuses in the attack's absence or presence.

$$MEM(d) = f(mem(d), ATK, n) \quad \text{and} \quad n \in [0, 1] \tag{5.2}$$

The expression ($MEM(d) = f(mem(d), ATK, n)$) represents a function f that calculates a memory usage measurement denoted as $MEM(d)$ for a specific IoT device (d) with or without the attack ATK.

Where (mem_d) the memory usage measurement (mem) of the smart device (d) at a point in time in the absence or presence of cyberattacks for a specific attack (ATK), and n is the number of memory usage measurements in a time interval, $f(mem(d), ATK, n) \in [0, 1]$ where 0 is the minimum memory usage measurement, and 1 presents the maximum memory usage measurement in the absence or presence of the attack. In essence, the function f takes into account the current memory usage measurement $mem(d)$ and also considers if there is an attack ATK to compute the overall normal memory usage measurement $MEM(d)$ of the smart device (d).

The CPU ($CPU(d)$) usage measurement is also calculated for the Raspberry Pi device as follows:

$$CPU(d) = f(cpu(d), ATK, n) \quad \text{and} \quad n \in [0, 1] \tag{5.3}$$

The expression ($CPU(d) = f(cpu(d), ATK, n)$) represents a function f that calculates the CPU usage measurement denoted as $CPU(d)$ for a specific IoT device (d) with or without the attack ATK.

Where (cpu_d) is the CPU usage measurement (cpu) of the smart device (d) at a point in time in the absence or presence of cyberattacks for a specific attack (ATK), and n is the number of CPU usage measurements in a specific time, $f(cpu(d), ATK, n) \in [0, 1]$ where 0 is the minimum CPU usage measurement, and 1 presents the maximum CPU usage measurement in the absence or presence of the attack. Therefore, the function f takes into account the current CPU usage measurement $cpu(d)$ and also considers if there is an attack ATK to compute the overall normal CPU usage measurement $CPU(d)$ of the smart device (d).

We do not calculate the CPU usage for the Arduino, as it is a microcontroller. We focus only on the maximum memory usage through or without the attack using a particular library called *MemoryFree* and *pgmStrToRAM*. And we also calculate *micros()* or *millis()* before and after sending the malicious attack. We also calculate the thread time for different statuses of the smart device, e.g., Idle, Active, and under attack.

5.5 Static Analysis of Resource Usage Attack

The smart devices used in this experiment were infected with malicious software used to carry out different malicious attacks on a target on an isolated network. During the experiments, the memory usage footprints of the smart devices were obtained under normal operating conditions, as well as when these smart devices carry out cyberattacks. Each memory usage footprint was obtained by taking measurements after 5 s within 1 minute when the smart device performs an attack and normal operation. A total of 10 minutes of calculation measurement of memory usage footprints of both in the presence of attacks and normal functioning smart devices were built. It's important to note that taking the measurement

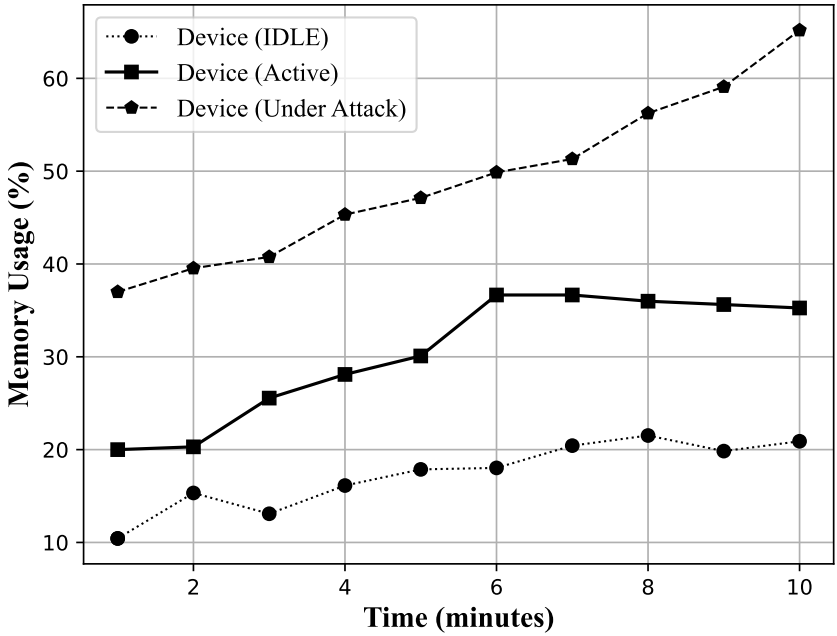

Figure 45. Raspberry Pi (Memory Usage Before and After the Attack).

footprint could take on any value, and these parameters could adapt to the particular scenario being evaluated. In our particular scenario, we configured it for a total of 1 minute to optimize resource usage, considering hardware capabilities and network bandwidth. This setting is highly flexible. For this experimental setup, we chose 1 minute intervals to consolidate measurements in a single request to the DB, effectively conserving resources.

During the process of packet collection, the attacks are initiated using the same flood commands for TCP and UDP. The topology, as illustrated in Figure 43, is utilized where the malicious TCP and UDP traffic is sent separately to the victim device, while all relevant usage statistics are recorded on the victim device. Each attack is simulated for a duration of 1 minute, and the corresponding usage statistics are recorded for the same duration. In the first period of 10 minutes, no attacks are sent, and all usage statistics are recorded and stored in the database. The same procedure is repeated once the second period begins after the malicious attacks have been launched.

The result of this experiment shows the memory usage footprint when the device is Idle, Active, and under attack. Therefore, the normal usage of the memory of the Raspberry Pi device in the absence of the attack fluctuates between 10% to 36%. This percentage is divided between two different states of the smart device; when it is Idle, the percentage

is between 10 and 20%, and when it is Active, the percentage is more than 25% but less than 37% as shown in Figure 45. Moreover, the percentage of memory usage changed after sending the malicious attack, and the percentage changed to be more than 66% per minute. We also calculate the CPU usage of the smart devices to check the CPU status before and after attacking the memory of the smart devices. Figure 46 shows the normal CPU usage for the Idle and Active statuses of the Raspberry Pi device. The normal CPU usage for Idle devices is between 0.55% to 0.88%. The memory usage of the Active smart devices is between 0.88% to 1.50%. At the same time, the CPU usage is more than 1.5% once we send the malicious attack to the smart devices. We also calculate the memory usage of another smart device (Arduino).

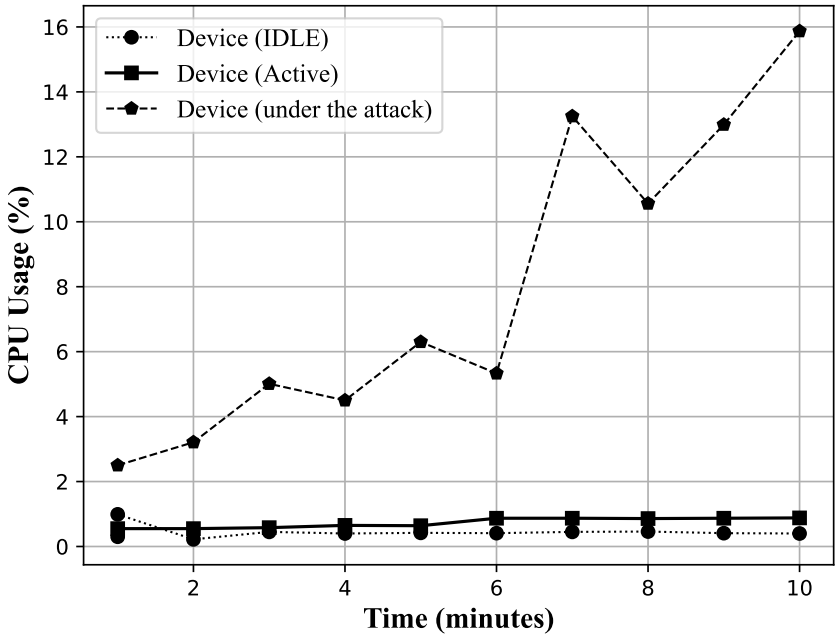

Figure 46. Raspberry Pi (CPU Usage Before and After the attack).

The main purpose of using two different devices is to show how the algorithm works for different devices which implement different architectures. For printing the memory usage of the Arduino device, we used a specific library to get the free usage memory for different statuses of the smart device, e.g., Idle, Active, and under attack. Therefore, the memory usage for the first status, as shown in Figure 47, fluctuates between 8.1% to 11%, and for the Active status, it is between 11% to less than 16%. The memory usage percentage changes to more than 17% and less than 50% once we send a malicious attack to the smart device.

The results and analyses of this experiment assisted us in understanding the impact of

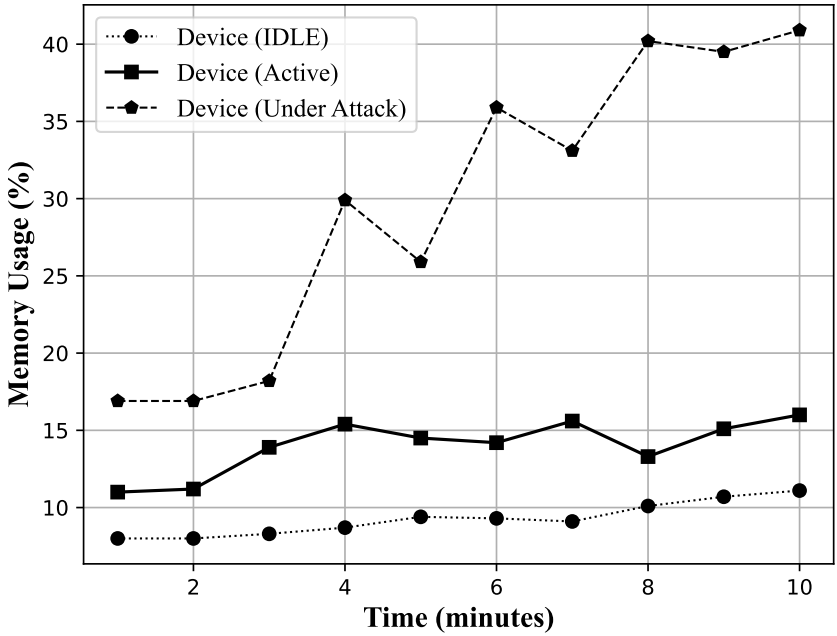

Figure 47. Arduino (Memory Usage Before and After the attack).

the memory usage attack on smart devices and building a lightweight algorithm to protect these devices from such an attack. The following section describes the detection algorithm and presents some results.

5.6 Threat Mitigation

This chapter introduces a detection mechanism and response to cyber-attacks on smart devices' memory usage. We also propose a lightweight algorithm to detect such memory changes inside smart devices by monitoring memory usage. Once the attack is detected, the algorithm will force the memory to stop listening to such an attack (e.g., stop reading and writing to memory). We also disconnect the victim devices from the Internet automatically. We implement this algorithm in the smart devices themselves. The presented mechanism records the response of the attack and memory usage for different states such as Idle, Active, and under attack. The detection algorithm detects any breach in the memory usage of smart devices.

5.6.1 Proposed Algorithm

The attacker aims to consume more memory usage of the smart device, and the monitoring mode of the presented algorithm updates and registers all different cases of memory behaviours before and after the attack. We record the change on memory for every 3 second for 1 minute. According to the data obtained from the testbed, the attacker can change the memory usage within 67% of wrong values during 10 minutes in total.

This Algorithm 4 takes the recorded readings from the DB for each smart device in the IoE system. The variable $Diff$ stores the subtraction of previous (*before sending such attack*) and current (*after sending malicious attack*) smart devices reading. For instance, the maximum memory usage of the smart devices for Idle and Active smart devices are given in Figure 42. The variable $Diff$ stores the difference between the previous and current memory usage readings. For instance, the maximum sudden memory usage change expected in the memory of the smart device is given by subtracting the value of the maximum memory usage when the device is under attack minus the minimum memory usage when the device is Active and Idle before sending any attack.

$$Reading_{Threshold} = Max_{usage(MEM)} - Min_{usage(MEM)} \tag{5.4}$$

Once the variable $Diff$ exceeds the expected value, the variable $T1$ (referred to as "Timer") is reset, and we proceed to verify whether an alert message has been sent to the administration. If not, we increment the $counter1$ variable, which keeps a record of the number of times the difference between the previous and current memory usage readings exceeds the maximum allowable value. Once the $counter1$ variable surpasses the maximum allowable value, it triggers the sending of an alert message, indicating that the memory usage of the specific smart device is under a memory usage attack. At this stage, we take further action by suspending all reading and writing operations to and from memory and disconnecting the smart device from the Internet. The IP addresses of all victim devices are then stored in the blacklist of our database to implement this action effectively.

Through experimentation, we consider the scenario when the attacker stops the attack. When the $Diff$ value is less than $Reading_{threshold}$ value, we compare whether the variable counter is greater than $zero$, and then we increase $T1$. We can assume the attack stops if the variable is greater than the $Time_{Threshold}$ variable. Finally, we reset the alert: $counter1$ and $T1$ variables.

After detecting the memory usage attack of such a device (d), we put all the victim devices on a blacklist. Then, once the attack is detected on such a device, we first stop any operation on the memory, e.g., read and write on memory. We disconnect the Internet connection of the smart device (d) to prevent any further attack on the smart device's memory usage. The next section presents different results regarding detecting memory usage attacks.

Therefore, the mitigation is summarized in the following steps:

1. add the victim smart devices' IP to a black-list;
2. stop any reading/writing to the smart device;
3. disconnect the smart device from the Internet.

Algorithm 4 A Technique to Detect Memory Usage Attack

Input: $d, Diff, C1, T1, Alert$
Output1: Normal($M1$)
Output2: Abnormal ($M2$)
Final Result: Output1 *or* Output2
$M1 : Reading_{memory-usage}$
$MEM(d) = f(mem(d), ATK, n)$
$M2 : Reading_{memory-usage}$
$Reading_{Threshold} = Max_{usage(MEM)} - Min_{usage(MEM)}$
$Diff = M1 - M2$
if $M2 = M1$ **then**
    if $Diff > Reading_{Threshold}$ **then**
        $ResetT1$
        if $Alert ==' On'$ **then**
            monitor memory
        else
            $C1 = C1 + 1$
            if $C1 > Max_{(memory-usage)}$ **then**
                $Alert ==' On'$
                Attack detected
                Detect the main source (X)
                Stopped Reading/Writing on Memory from (X)
                Disconnect the smart device (d) from the Internet
            else
                Return back to monitor memory
    else
        if $C1 > 0$ **then**
            $T1 = T1 + 1$
            if $T1 > Threshold_T$ **then**
                Reset Alert
                Reset C1
                Reset C1
                Attack stopped
            else
                return back to monitor memory

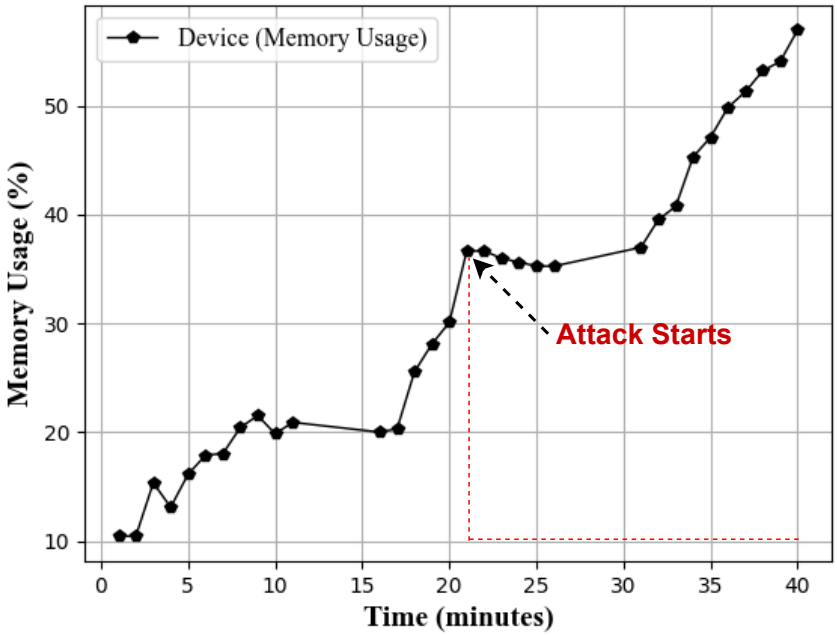

Figure 48. Raspberry Pi (Detecting the memory usage Attack.)

5.7 Experimentation and Discussion

5.7.1 Results

We ran malicious attacks on the smart device to check the memory usage before and after the attack. Figure 48 shows the mechanisms of our algorithm to fetch the attack once it is started. The monitoring mode of the memory usage sends memory usage readings to the algorithm, and inside the algorithm, there is a statistics comparison between normal and abnormal cases. As described in Section 5.4, we first check the behaviour of the smart devices once there is an attack, and we register all different cases for memory usage, e.g., Idle, Active, and under attack. The main purpose of this analysis is to study the attack first and then to build a mitigation mechanism to detect memory attacks.

Figure 48 and 49 shows the presented results of detecting the attack once it starts; we can notice that the attack starts when the memory usage is greater than 37%, and the $Diff$ variable is greater than the expected memory usage value. At this stage, the smart device d is passed through different operations, e.g., stop reading/writing on d, disconnect d from the Internet to stop any further attack, and send an alert to the administration about the status

of the smart device.

The detection algorithm also notified the administration once the attack stopped. This stage will help with further operations. Through this experiment, we also studied the be-

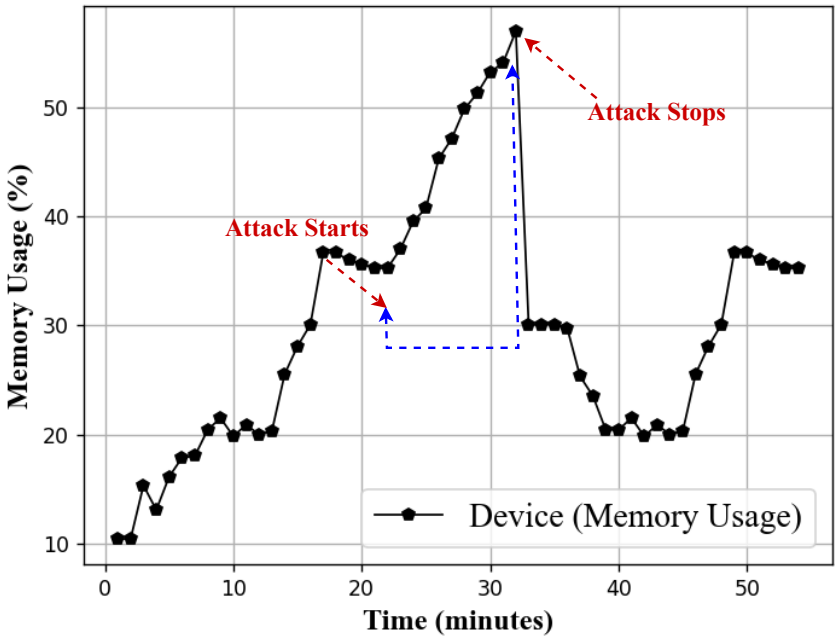

Figure 49. Raspberry Pi (Detecting the memory usage once the attack starts and when it stops).

haviour of the CPU usage of the Raspberry Pi device under the same attack. Figure 50 shows the behaviour of the CPU usage before and after attacking the memory of the smart devices. We also applied the same detection algorithm to study the behaviour of the mitigation algorithm on the CPU and whether this algorithm detects the attack or not.

The same calculation is applied to the Arduino, and the detection algorithm records different variables about the attack once it is started and stopped. Figure 51 shows the recorded results of detecting the attack. We can notice that the attack started when the memory usage percentage increased to be more than 16%, and for detecting the attack when it is stopped, once the memory usage percentage decreased to be less than 20%. Once the system detects that the attack on the smart device d has ceased, it may take actions such as disconnecting the smart device from the Internet or stopping the actual attack at the main source.

The algorithm also stores all victim devices' IPs in the black-list, so when there is an attack on the smart device, we disconnect the smart device to prevent any further attack. We

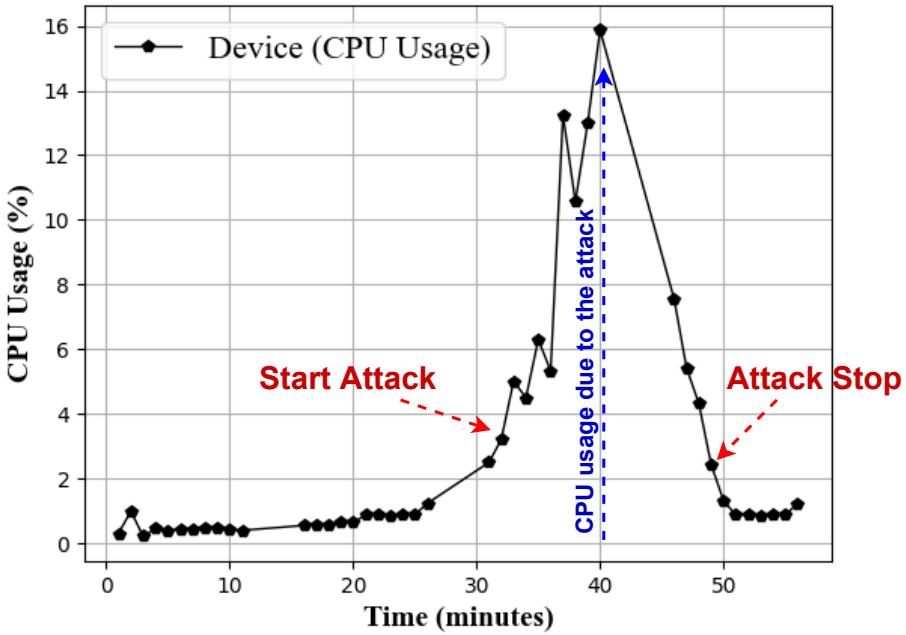

Figure 50. CPU Usage during the attack when it started and stopped (Raspberry Pi).

also prevent further access to the database until the administration team solves the issue. Finally, this algorithm shows high efficiency in detecting memory usage attacks in smart devices. The memory usage during detection fluctuates between two normal states: *Idle* and *Active*. For instance, memory usage remains below 35% for the Raspberry Pi and less than 16% for the Arduino. Additionally, CPU usage is measured for the Raspberry Pi, registering a final percentage of less than 2.5% during the detection mechanism.

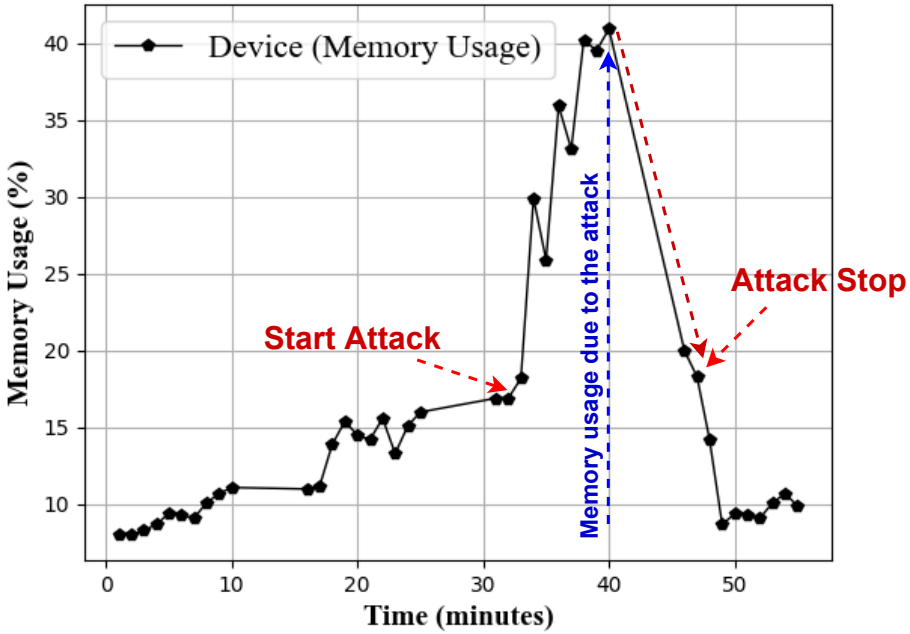

Figure 51. Arduino (Detecting the memory usage once the attack starts and when it stops).

Chapter 6

Conclusion and Future Developments

6.1 Conclusions

Nowadays, the security of IoT has become a crucial topic for researchers. By connecting billions of things to the Internet, IoT and IoE created a plethora of applications that impact every aspect of human life. Mission-critical, time-sensitive applications require robust connectivity and strict reliability constraints. However, IoT devices face challenges due to resource constraints, resulting in problems. Therefore, ensuring continuous device availability and communication reliability are critical factors in guaranteeing a constant, confident, and reliable flow of application data.

The lack of a detection mechanism to identify resource constraint attacks would significantly impact the performance of devices and networks. In this thesis, we have introduced optimized methods for monitoring and detecting such attacks in real smart devices. The primary objective of these monitoring mechanisms is to analyze the impact of resource constraint attacks and develop detection and mitigation methods to minimize their severity. Throughout our research, we have focused on various topics, and the following points summarize our key findings and results:

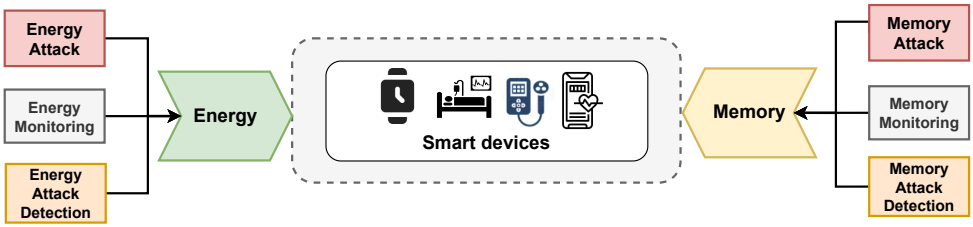

Figure 52. Thesis Conclusions.

6.1.1 Analysis of the impact of Energy Consumption Attacks on Smart Devices

Chapter 3 studied the impact of EC-DDoS and F-APs attacks on the resource usage of different smart healthcare devices and, more specifically, on energy consumption. First, Docker images were utilized to collect data, scan the smart devices' networks, and sniff the network. Then, the calculations of the AR, SD, and threshold of the AR were conducted on the victim's side. The main purpose of the calculation is to study the effect of DDoS attacks on the connectivity of smart healthcare devices. Influential factors such as ports, device state, attack type (i.e., protocols used), and AR were also examined. Furthermore, the analysis focused on examining the effects of DDoS, EC-DDoS, and F-APs attacks on the energy consumption of smart devices. Specifically, the F-APs attack was designed to affect the energy resources of the smart devices by automatically sending malicious attacks to the connected smart healthcare devices. This research provides a deeper understanding of the impact of DDoS, EC-DDoS, and F-APs attacks on the energy consumption and connectivity of smart healthcare devices within wireless networks. The analysis results of this chapter show the behaviour of smart devices under energy consumption attacks, which helps future works to build the best detection mechanisms that consume less energy and detect or mitigate energy consumption attacks with IoT systems. Furthermore, future extensions of this work may consider exploring additional scenarios. For instance, the effects of combining DDoS attacks and F-APs on the memory usage of smart healthcare devices could be investigated.

6.1.2 Detection of Energy Consumption Cyber Attacks on Smart Devices

The IoT is an Internet of smart objects where smart objects communicate with each other. IoT objects are deployed in an open medium with dynamic topology. Due to a lack of infrastructure and centralized management, IoT presents serious vulnerabilities to security attacks, such as energy consumption attacks, as smart devices suffer from resource constraints. Therefore, security is an essential prerequisite for the real-world deployment of IoT. In Chapter 4, a new technique is proposed for detecting energy consumption attacks in smart home devices based on the IoT devices' packet rate analysis. This technique considers the received packets related to the IoT devices for different protocols such as TCP, UDP, and the subscribed packets of the MQTT protocol. Therefore, with the aim of energy consumption attack detection, the packet reception rate of the IoT devices is calculated and analyzed for each protocol separately or all the protocols simultaneously. The algorithm considers different protocols and device statuses, demonstrating high efficiency in detecting energy consumption attacks in smart home devices compared to other algorithms that use the current energy consumption measurement to detect this attack. As this algorithm is easy to use and not expensive to implement, it also considers the resource constraints of smart devices.

The key observations made from Chapter 4 present a thorough understanding of the packet reception rate of IoT devices within a home wireless environment. And how the energy consumption attacks could be detected depending on measuring the packet rate received by the smart devices. Future research will focus on identifying the main sources contributing to high energy consumption in smart home environments by detecting the attack type.

6.1.3 Mitigating and Analysis of Memory usage attack in IoE system

The IoE is the beginning of a new era of technology in Internet-based smart communication and connecting smart devices. The security of IoE pillars is important as some suffer from resource constraints problems. Chapter 5 proposed an approach that can detect and classify memory usage attacks using memory-based features extracted from the memory usage of the smart device. The approach represents a mitigation method to detect the attack once it appears in the memory usage of the smart devices. First, memory usage is monitored using a specific tool implemented in *Python* script and *C* language to fetch different data about memory usage. Then, all the fetched data is stored in the DB for further calculation. Second, the behaviour of the attack is studied, and memory usage readings are recorded before and after the attack. In this work, both static and dynamic analyses of the memory usage attack are conducted. In particular, we have conducted all the experiments in an isolated and cost-efficient experimental setup. It is observed that malicious attacks, e.g., flooding attacks, have a significant impact on the resources of the IoE smart devices. When an IoE edge device is flooded with malicious attacks, there are significant increases in CPU and memory usage. This analysis helps in building the detection algorithm. The detection method relies on monitoring the memory usage to compare different variables of the memory reading. It is also able to detect the attack on time once it happens. Moreover, it can detect if the intruder stops the attack or not. We also build an alert message inside the algorithm to send different notifications to the administration once the attack is detected. Moreover, all victim devices are disconnected from the Internet, and all read/write operations to and from memory are also stopped. In the future, we will focus on detecting the main sources of memory usage attacks in the IoE environment.

6.2 Future developments

This thesis aims to investigate resource constraints attack smart devices within the Internet of Things and Internet of Everything with a particular focus on analyzing the effect of resource constraint attacks. The study discussed the impact of resource-constrained attacks and the development of lightweight algorithms for detecting resource-constrained attacks, such as energy and memory attacks. Along this line of research, several research areas can be potentially identified. To gain a comprehensive outlook, future developments can be broadly divided into four categories (i) network challenges, (ii) data security challenges, (iii) physical layers challenges, and (iv) industrial scenario challenges.

6.2.1 Network challenges

The rapid development of IoT devices and networks in various forms generates enormous amounts of data, which in turn demand careful authentication and security. Therefore, another approach could be applied to develop an algorithm to detect the main sources of resource constraint attacks in IoT and IoE systems since smart devices suffer from resource constraint problems. Therefore, it is essential to detect the main sources of resource constraint problems and mitigate the attack at the early stage. An IDS is a security detection

system that can monitor the network and detect threats [259]. Artificial Neural Network (ANN) intrusion detection could be used to gather and analyze information from various parts of the IoT network and identify resource constraints attacks [260, 18].

Consider an IoT network of n sensor nodes, where $n-1$ nodes function as clients, and one node acts as a server relay for data analytics. The network traffic can be captured non-intrusively using a network tap to avoid altering live traffic. The server node acknowledges the data received from the sensor nodes and responds with relevant data, enabling the sensor nodes to adapt and respond to events, as depicted in Figure 53 on the left side.

In this context, the attack could be from an external intruder. The attacker only targets the server node, as it analyzes, logs, and responds to the sensor nodes. To implement this experiment, we can send different malicious attacks that affect the resources of the smart devices. The IDS system will try to identify the main sources of resource constraint attacks, and ANN will be able to detect resource constraint attacks in the future by learning from the current behaviours. The detection could be able to classify normal and threat patterns. The ANN model could be used to validate against a simulated IoT network, demonstrating over $n\%$ accuracy. It could be used to identify successfully different types of attacks and showed good performances in terms of true and false positive rates. We could also introduce different types of attacks to test the method's reliability and improve the framework's accuracy. Therefore, this idea could be used to present a resource constraint threat analysis of the IoT and uses an ANN to combat these threats. A multi–layer perceptron [261, 18], a type of supervised ANN, could be used to train using internet packet traces, then assess its ability to thwart resource constraint attacks [262, 18]. This idea focuses on classifying normal and threat patterns on an IoT Network.

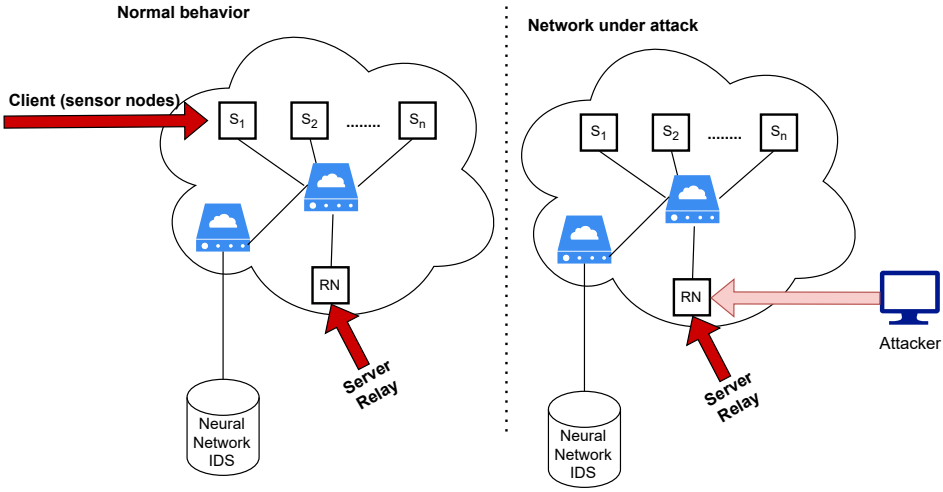

Figure 53. Experimental architecture to mitigate the main sources of resource-constraints attacks in IoT or IoE systems.

The application of IDS based on Tiny Machine Learning (TinyML) has recently gained

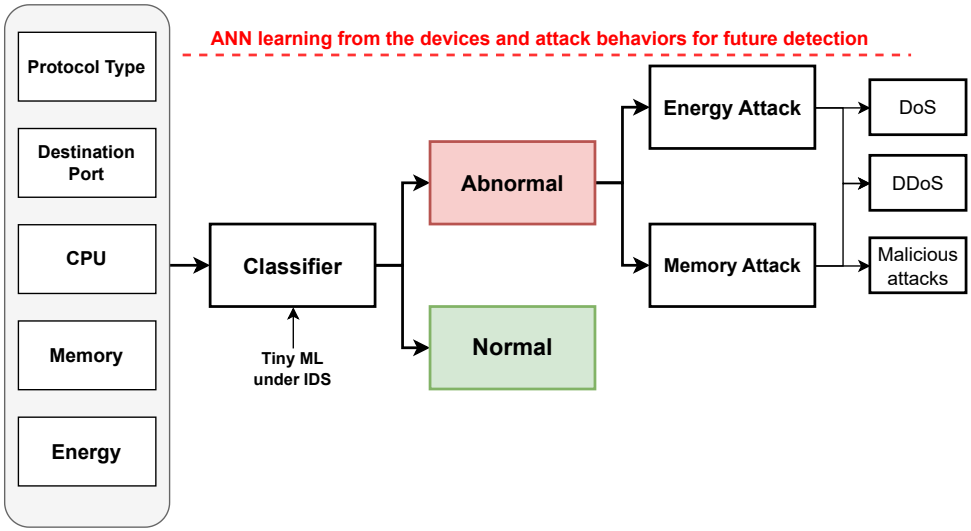

Figure 54. Detection Model.

notable attention because of its distinctive attributes and capabilities, making it applicable in various IoT contexts. Recent research has advocated for utilizing TinyML-based IDS to enhance IoT security. Various researchers leverage TinyML for fault detection in IoT system smart devices. For instance, in [263], the author applied TinyML to improve Gas Metal–Oxide–Semiconductor (GMOS) sensor outcomes. Their setup included a GMOS sensor detecting ethanol and acetone, an Arduino board, and LEDs indicating gas detection. The training was conducted using Edge Impulse (EI). The neural network comprised an input layer with 24 features, 30 neurons for one hidden layer, and 20 neurons for another, with three classes in the output layer. The detection success rate for each gas (ethanol, acetone, and no gas) in the test set was 100%. Resource usage comprised 1.7 KB of RAM and 19.5 KB of flash memory, with a latency of 1 ms. In this paper [264], the author advances the existing jamming detection and classification techniques by proposing an efficient IoT approach based on TinyML. A deep learning model is trained and deployed on an IoT edge device, specifically a Raspberry Pi, using TensorFlow lite. The model, constructed with TensorFlow, comprehensively covers two prevalent jamming types, constant and periodic, alongside the normal channel state. The Raspberry Pi is linked to a Software Defined Radio (SDR) for real-time WiFi channel sensing, capturing Received Signal Strength (RSS) readings. The TinyML model evaluates these readings to identify the presence and type of jamming. Furthermore, an extensive testing campaign is conducted to thoroughly assess and demonstrate the efficacy of the proposed TinyML-based edge detection approach. Therefore, utilizing TinyML in the context of detecting resource-constrained attacks on devices is a promising approach to conserving energy and memory while timely identifying potential attacks.

6.2.2 Industrial challenges

The IIoT is revolutionizing the operations of numerous industrial enterprises, elevating their capabilities to new heights. By seamlessly integrating the physical and digital realms with minimal human intervention, IIoT has a profound impact on the economy and modern business landscape [265, 30]. The data generated by the IIoT is leveraged by artificial intelligence systems to perform intelligent tasks, such as optimizing the efficiency of interconnected machines, error correction, and preventive maintenance. However, the widespread integration of IIoT comes with the risk of facing sophisticated security threats at different levels of the connectivity and communications infrastructure it encompasses. Ensuring availability, confidentiality, and integrity becomes challenging due to the diverse and complex nature of IIoT infrastructures. As a result, potential mistrust in network operations and concerns about privacy breaches or the loss of crucial personal data and sensitive information of network end-users may arise [30]. Machine learning algorithms can be applied to efficiently detect benign and malicious nodes in IIoT networks to secure IIoT networks from various attacks efficiently. The contributions of applying a machine learning algorithm in IIoT are to effectively detect benign and malicious nodes in an IIoT network and provide a novel method for using information through transfer learning while avoiding catastrophic forgetting of previously learned data. We can design a model that can add new information to an already-trained network without starting training from scratch again. The designed model could be combined with other algorithms, e.g., distributed sleep scheduling algorithm, to boost energy efficiency. Furthermore, we can train the designed model on extensive datasets involving many attack scenarios to enhance the model's effectiveness.

6.2.3 Secure big-data transmissions

In the modern era, numerous big-data applications have emerged, encompassing diverse areas like large-scale monitoring of smart cities, healthcare, agriculture, and more. However, the transmission of big data presents significant challenges due to various factors. Chief among them is the unavailability of services and protocols capable of efficiently handling data transmission in the range of 70 to 160 Tbits per second [266] [267], making the process complex. Additionally, the high density of traffic renders real-time monitoring of data transmission practically infeasible. To tackle these hurdles, the development of Deep Learning (DL) offers a promising solution by providing security protection for the vast volumes of data being transmitted. DL exhibits practical features that can be further refined to effectively manage and process such large-scale data transmission.

6.2.4 Data security challenges

Due to the increase of smart devices on the Internet and the number of transmitted data that has increased significantly, data security issues have also increased. Thus, securing smart devices' data is essential. Therefore, the transmission must occur at high rates and offer reliability through high secrecy, low packet loss, and small delay. Furthermore, those smart devices must be affordable to justify their implementation on an IoT system-wide scale, thus having low power consumption and the most cost-efficient embedded processing unit possible [268]. Since the main source of information security in today's landscape

is provided through cryptography, the secrecy constraint [269] can negatively affect most of these criteria. As a result of the growth in the availability of portable and connected equipment with high processing capabilities, the safety measures implemented need to match this computational power with proportionally more prolonged and more complex keys so as not to be vulnerable to brute-force attacks from well-equipped malicious devices [268], [270]. However, this approach is not sustainable because it produces increasingly long authentication routines due to the increased computational overhead and processing cost due to the implemented security algorithms. Moreover, the current solution is not suitable for the resource constraints problems of smart devices in IoT systems. Therefore, we can apply Physical Layer Security (PLS) techniques as additional protection to increase the secrecy of wireless communications in IoT and IoE environments. As the name suggests, PLS could be applied at the physical layer, making it an alternative that can be used with low processing cost compared to cryptography, which is more oriented towards the computational side of the network stack on the application layer.

However, Physical Layer Authentication (PLA) can also be used to prevent malicious users from spoofing IoT devices' information, and the first critical step is effective authentication [271]. And PLA employs unique characteristics inherent to wireless signals and physical devices and is promising in the IoT due to its flexibility, low complexity, and transparency to higher layer protocols [272]. Therefore, PLS and PLA could be used to ensure IoT devices' data confidentiality, availability, and reliability [271].

6.2.5 Physical layer

PLS [273] is an emerging technique proposed to enhance wireless transmissions by exploiting the physical characteristics of the wireless channel. It presents low computational cost and overhead by injecting an interfering signal into potential eavesdroppers' wiretap channels. The key principle of PLS is to permit the secure transmission of confidential data using efficient signal-processing techniques [271]. Moreover, PLS has been recognized as a possible approach for achieving confidentiality at the physical layer by utilizing the inherent randomness of wireless communications. It can be used to provide secure wireless communications without using a key to encrypt them. Also, DL has emerged as a viable option to address various security concerns and enhance the performance of conventional PLS [211] techniques in wireless networks [274] [275]. DL is a strong data exploration technique that can be used to learn normal and abnormal behaviour of 5G and beyond wireless networks in an insecure channel paradigm. Also, since DL techniques can successfully predict future instances by learning from existing ones, they can successfully predict new attacks, which frequently involve mutations of earlier attacks. Thus, motivated by the benefits of DL and PLS, we can combine PLS and DL to solve various security concerns in 5G, 6G, smart devices resource constraints in IoE, and beyond networks, e.g., supply chain risk, spectrum sharing, network slicing, and others [211]. Furthermore, DL appears to be viable for addressing security issues and designing PLS techniques for 5G and beyond networks. Especially, pre-5G networks partially addressed privacy concerns by storing user data in databases owned by mobile operators [276]. Additionally, 5G and beyond networks will confront new security issues due to the rise in User Equipment (UE), services, heterogeneity of connected UEs, high privacy concerns, and new requirements to support various IoT technologies. Additionally, most 5G apps are decentralized, allowing UEs to join or leave the network whenever

they choose [277]. Therefore, PLS is a strong data exploration technique that can be used to learn the normal and abnormal behaviour of wireless networks based on how UEs and base stations communicate with each other. We can also use the combination of PLS and DL to predict and detect memory attacks in smart devices from the viewpoint of physically attacking the memory [278]. We can also use the PLS and DL to predict the main sources of resource constraint attacks before they happen on the smart devices by sending alerts of the different behaviours of the smart devices [279] [211].

Chapter 7

Appendix

This section includes crucial details about the databases and tables used in the experiments of Chapters 3, 4, and 5. These elements are vital for data collection, organization, and analysis, contributing to the study's findings. Additionally, it provides information about the energy consumption tools utilized to measure smart devices' energy usage accurately and the technologies used to measure and monitor the memory usage of smart devices.

7.1 Technologies

The fake access point created by the author in Chapter 3 is implemented on a virtual machine by using tp-link (*TL-WN722N*) as shown in Figure 55. We run malicious attacks, network scans, and sniffers on different Docker images. In particular, the control panel docker contains all the scripts that manage the population of malicious attacks. It also includes the network scan to scan the network and ports of the smart devices. Another docker image has a sniffer to capture the WiFi traffic of the IoT system. We use *hping3*, *tshark*, and *Nmap* inside the docker images for running different things, e.g., malicious attacks, sniffing and fetching the packets automatically and for launching network scans and identifying devices status.

For monitoring the memory of Chapter 5, we used different libraries in *Python* and C *languages*. For example, we used *os*, *psutil*, and *memory-monitoring*, and for the *C* language, we used *MemoryFree* and *pgmStrToRAM*.

The main technologies used for implementing the lightweight algorithms are *MySQL*, *Python*, *C*, and *Matlab*.

7.2 Database

The database used to store all information related to the smart devices we use the MySQL database. MySQL is an open-source SQL relational database management system that's developed and supported by Oracle. The server of the database uses to store data securely and return it in response to another request by the software applications. Figures 56, 57, and 58

Figure 55. TP-Link (*TL-WN722N*) USB Adapter

shows the database's structure that contains all the smart devices data for Chapters 3, 4, and 5. The description of each table is described as follows:

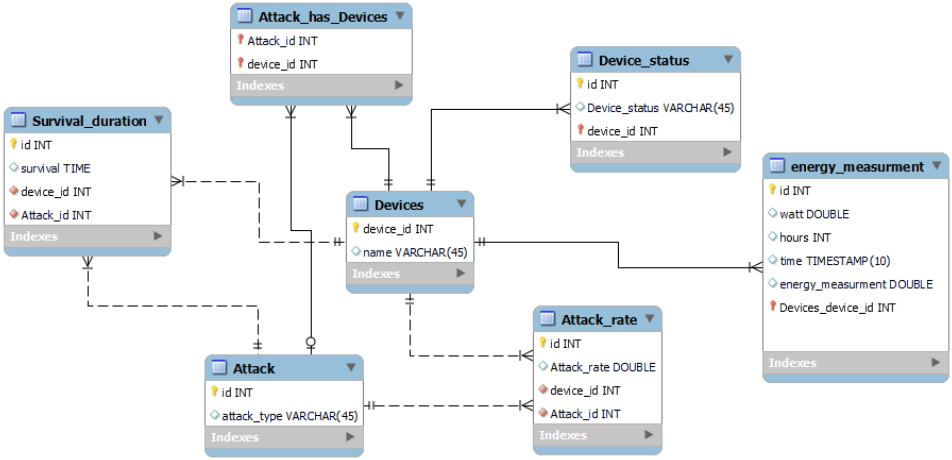

Figure 56. Chapter 3 Database Schema.

1. *Devices* table has all the devices information: *deviceId* represents the smart device ID used in the expirement; *name* is the name of the smart device.
2. *DevicesStatus* table contains all the devices statuses, e.g., *Idle*, *Active*, *under attack*; *deviceId* is a foreign key from *Device* table represents the devices used by the experiment.
3. *Attack* table contains all the information related to the malicious attacks: *attakType* refers to the type of attack used against the smart devices, e.g., DDoS, F-APs, and others.
4. *energy measurement* table saves all the information of all the smart devices energy measurement: *watt* refers to current energy measurement; *hours* contains a number of

hours used to measure the energy of the smart device; *time* contains the information about the starting and ending time of measuring the energy; *device id* is the smart device.

5. *Attack rate* table contains all the information about the attack rate calculations: *attack rate* stores the attack rate of such a smart device (*device id*).
6. *Survival duration* table stores the survival duration of the smart devices: *Survival* is the survival duration of the smart device (*device id*).
7. *Protocol* table stores the protocols used for the experiment of Chapter 4.
8. *Packet reception rate* table stores the packet received by the smart devices: *packet* is the number of packets received by the smart device (*device id*) and for a specific protocol (*protocol id*) within a specific type of attack (*attack id*).

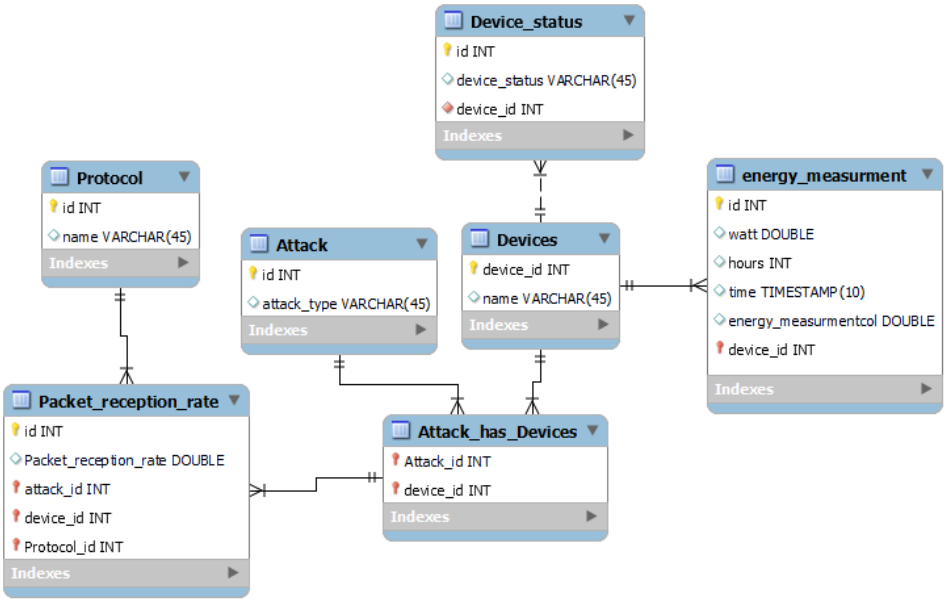

Figure 57. Chapter 4 Database Schema.

9. *memory usage* table contains all information about the memory usage of the smart devices (*device id*) for different statuses: *memory usage* stores the memory usage of the smart device within a specific time (*time*) with or without the attack (*attack id*).
10. *CPU usage* table contains all information about the CPU usage of the smart devices (*device id*) for different statuses: *CPU usage* stores the CPU usage of the smart device within a specific time (*time*) with or without the attack (*attack id*).
11. *blacklist* table stores all the information about the victim smart devices: *ip* is the IP of the victim smart device; *date* the date of inserting the smart device (*device id*) into the blacklist;

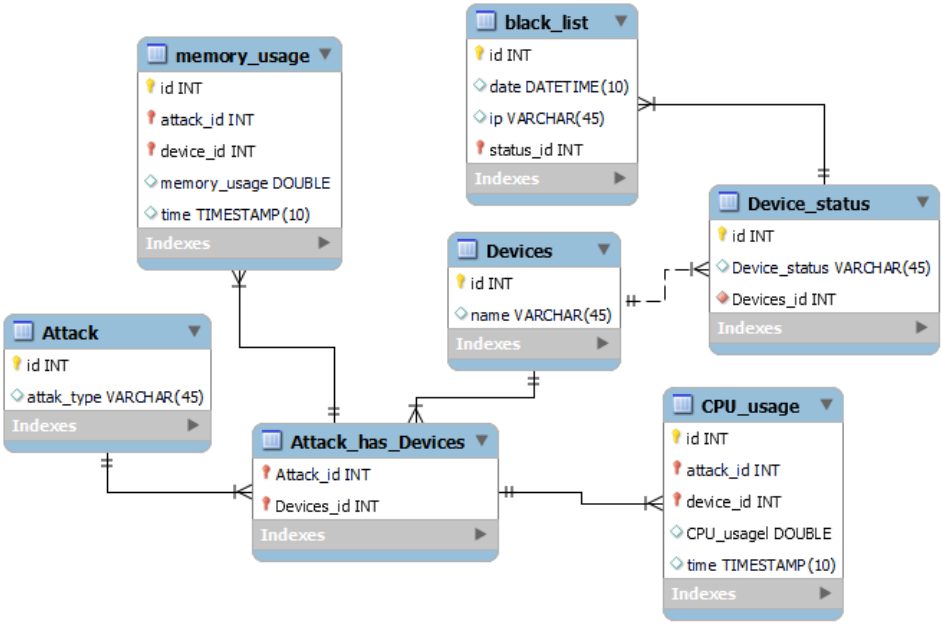

Figure 58. Chapter 5 Database Schema.

7.3 Power Measurements

We developed a smart circuit using a non-invasive current sensor, as shown in Figure 60, to measure the current consumption of smart healthcare devices. This smart circuit samples voltage, ampere, watt, and current per second. The current consumption values for each smart healthcare device are stored in the DB. In our experiment, we use the Joule (J) values to calculate the energy consumption of smart devices.

Within this smart circuit, we can measure the current consumption of the smart device in n seconds. So in our experiment in chapter 3, we measure the energy consumption for every (3) seconds in a total of (30) minutes.

To run this smart circuit, we used Arduino UNO with four types of resistors; one of the resistors is 330 Ω linked with led to show once the start of measuring the energy consumption by the smart devices. Other three resistors sized 10k Ω connected with the circuit of the smart sensor. We also used a capacitor 10 uF.

We also need to install *EmonLib.h* library for reading data from the non-invasive current sensor.

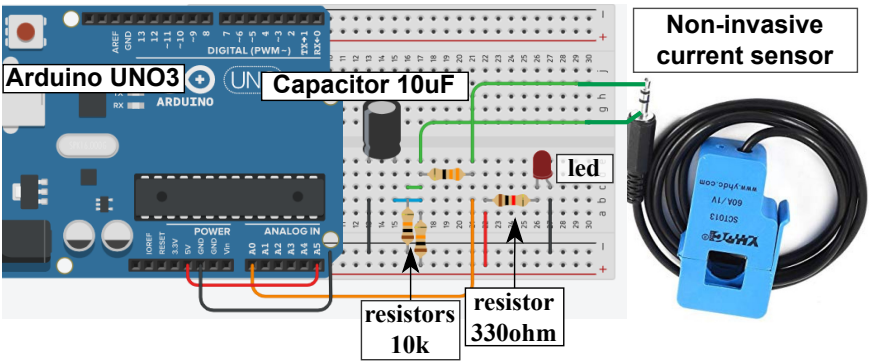

Figure 59. Circuit for measuring current consumption.

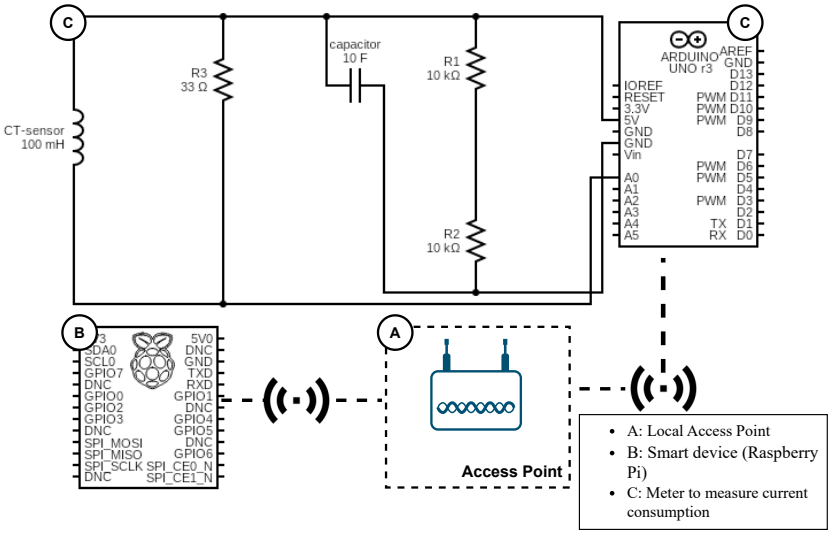

Figure 60. Schematic Sketch for Figure 34 from chapter 4.

Table 8. Components for Power Consumption Measurements.

Component	Value
Arduino UNO3	—
Resistor	330 Ω
Resistor	10 k Ω
. Resistor	10 k Ω
Capacitor	10 uf
LED	—
Non-invasive current sensor	30 A

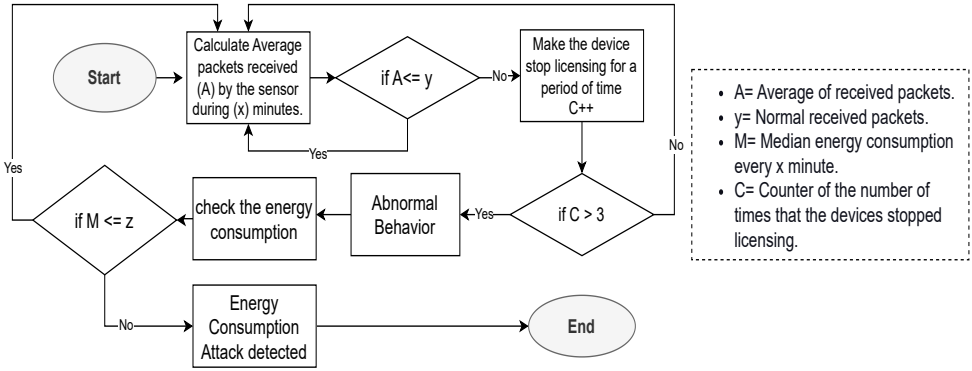

Figure 61. A Technique to Detect Energy Consumption Attack for the algorithm in Chapter 5.

Bibliography

[1] Kevin Ashton. "That 'Internet of Things' Thing". In: 1999.

[2] Gajjala Savithri, Bhabendu Kumar Mohanta, and Mohan Kumar Dehury. "A Brief Overview on Security Challenges and Protocols in Internet of Things Application". In: *2022 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*. 2022, pp. 1–7. DOI: 10.1109/IEMTRONICS55184.2022.9795794.

[3] Amit Kumar Sikder et al. "A Survey on Sensor-Based Threats and Attacks to Smart Devices and Applications". In: *IEEE Communications Surveys and Tutorials* 23.2 (2021), pp. 1125–1159. DOI: 10.1109/COMST.2021.3064507.

[4] Nadia Chaabouni et al. "Network Intrusion Detection for IoT Security Based on Learning Techniques". In: *IEEE Communications Surveys and Tutorials* 21.3 (2019), pp. 2671–2701. DOI: 10.1109/COMST.2019.2896380.

[5] David Reinsel, John Gantz, and John Rydning. "Data age 2025: The evolution of data to life-critical. Don't focus on big data; focus on the data that's big". In: *International Data Corporation (IDC) White Paper* (2017).

[6] Mohammad Mezanur. "Internet-of-Things (IoT) Security Threats: Attacks on Communication Interface". In: 2020.

[7] A Rehash Rushmi Pavitra, I Daniel Lawrence, and P Uma Maheswari. "To Identify the Accessibility and Performance of Smart Healthcare Systems in IoT-Based Environments". In: *Using Multimedia Systems, Tools, and Technologies for Smart Healthcare Services*. IGI Global, 2023, pp. 229–245.

[8] Duan Yan-e. "Design of intelligent agriculture management information system based on IoT". In: *2011 Fourth International Conference on Intelligent Computation Technology and Automation*. Vol. 1. IEEE. 2011, pp. 1045–1049.

[9] Fan Wu et al. "WE-Safe: A wearable IoT sensor node for safety applications via LoRa". In: *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*. IEEE. 2018, pp. 144–148.

[10] Fan Wu, Taiyang Wu, and Mehmet Rasit Yuce. "An internet-of-things (IoT) network system for connected safety and health monitoring applications". In: *Sensors* 19.1 (2018), p. 21.

[11] Basma Hassan. "Monitoring the Internet of Things (IoT) Networks". PhD thesis. Dec. 2019.

[12] Antonio J Jara, Latif Ladid, and Antonio Fernandez Gómez-Skarmeta. "The Internet of Everything through IPv6: An Analysis of Challenges, Solutions and Opportunities." In: *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.* 4.3 (2013), pp. 97–118.

[13] Miloš Stanisavljević, Alexandre Schmid, and Yusuf Leblebici. *Reliability of nanoscale circuits and systems: methodologies and circuit architectures*. Springer Science & Business Media, 2010.

[14] Roberto Minerva, Abyi Biru, and Domenico Rotondi. "Towards a definition of the Internet of Things (IoT)". In: *IEEE Internet Initiative* 1.1 (2015), pp. 1–86.

[15] David Metcalf et al. "Wearables and the internet of things for health: Wearable, interconnected devices promise more efficient and comprehensive health care". In: *IEEE pulse* 7.5 (2016), pp. 35–39.

[16] Rebecca Jamwal et al. "Smart home and communication technology for people with disability: a scoping review". In: *Disability and Rehabilitation: Assistive Technology* 17.6 (2022), pp. 624–644.

[17] Tanveer Ahmad et al. "Data-driven probabilistic machine learning in sustainable smart energy/smart energy systems: Key developments, challenges, and future research opportunities in the context of smart grid paradigm". In: *Renewable and Sustainable Energy Reviews* 160 (2022), p. 112128.

[18] Yalin Liu et al. "Unmanned aerial vehicle for internet of everything: Opportunities and challenges". In: *Computer communications* 155 (2020), pp. 66–83.

[19] Chrysi K. Metallidou, Kostas E. Psannis, and Eugenia Alexandropoulou Egyptiadou. "Energy Efficiency in Smart Buildings: IoT Approaches". In: *IEEE Access* 8 (2020), pp. 63679–63699. DOI: 10.1109/ACCESS.2020.2984461.

[20] Zainab Alwaisi, Simone Soderi, and Rocco De Nicola. "Energy Cyber Attacks to Smart Healthcare Devices: A Testbed". In: *Bio-inspired Information and Communications Technologies*. Ed. by Yifan Chen, Dezhong Yao, and Tadashi Nakano. Cham: Springer Nature Switzerland, 2023, pp. 246–265. ISBN: 978-3-031-43135-7. DOI: https://doi.org/10.1007/978-3-031-43135-7_24.

[21] Zainab Al-Waisi, Simone Soderi, and Rocco De Nicola. "Detection of Energy Consumption Cyber Attacks on Smart Devices". In: EAI-SPRINGER, 2023.

[22] Zainab Al-Waisi, Simone Soderi, and Rocco De Nicola. “Mitigating and Analysis of Memory Usage Attack in IoE system”. In: EAI-SPRINGER, 2023.

[23] Rajkumar Buyya and Amir Vahid Dastjerdi. *Internet of Things: Principles and paradigms*. Elsevier, 2016.

[24] Enzo Baccarelli et al. “Fog of Everything: Energy-Efficient Networked Computing Architectures, Research Challenges, and a Case Study”. In: *IEEE Access* 5 (2017), pp. 9882–9910. DOI: 10.1109/ACCESS.2017.2702013.

[25] Mohammed H Alsharif et al. “Green IoT: A review and future research directions”. In: *Symmetry* 15.3 (2023), p. 757.

[26] Sadiat Jimo, Tariq Abdullah, and Arshad Jamal. “IoE Security Risk Analysis in a Modern Hospital Ecosystem”. In: *Cybersecurity in the Age of Smart Societies: Proceedings of the 14th International Conference on Global Security, Safety and Sustainability, London, September 2022*. Springer. 2023, pp. 451–467.

[27] Muhammad Sajid, Ali Harris, and Shaista Habib. “Internet of Everything: Applications, and Security Challenges”. In: *2021 International Conference on Innovative Computing (ICIC)*. IEEE. 2021, pp. 1–9.

[28] P Gomathi, S Baskar, and P Mohamed Shakeel. “Concurrent service access and management framework for user-centric future internet of things in smart cities”. In: *Complex & Intelligent Systems* 7.4 (2021), pp. 1723–1732.

[29] Ismail Butun, Patrik Österberg, and Houbing Song. “Security of the Internet of Things: Vulnerabilities, attacks, and countermeasures”. In: *IEEE Communications Surveys & Tutorials* 22.1 (2019), pp. 616–644.

[30] Minhaj Ahmad Khan and Khaled Salah. “IoT security: Review, blockchain solutions, and open challenges”. In: *Future Generation Computer Systems* 82 (2018), pp. 395–411. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2017.11.022.

[31] Athar Ali Khan, Mubashir Husain Rehmani, and Abderrezak Rachedi. “Cognitive-Radio-Based Internet of Things: Applications, Architectures, Spectrum Related Functionalities, and Future Research Directions”. In: *IEEE Wirel. Commun.* 24.3 (2017), pp. 17–25.

[32] Fayaz Akhtar, Mubashir Husain Rehmani, and Martin Reisslein. “White space: Definitional perspectives and their role in exploiting spectrum opportunities”. In: *Telecommunications Policy* 40 (4 2016). ISSN: 03085961. DOI: 10.1016/j.telpol.2016.01.003.

[33] Hamed HaddadPajouh et al. “A survey on internet of things security: Requirements, challenges, and solutions”. In: *Internet of Things* 14 (2021), p. 100129.

[34] Jie Lin et al. “A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications”. In: *IEEE internet of things journal* 4.5 (2017), pp. 1125–1142.

[35] John Garrity. "Harnessing the Internet of Things for global development". In: *Available at SSRN 2588129* (2015).

[36] Mahdi H Miraz et al. "A review on Internet of Things (IoT), Internet of everything (IoE) and Internet of nano things (IoNT)". In: *2015 Internet Technologies and Applications (ITA)* (2015), pp. 219–224.

[37] Janet L Holland and Sungwoong Lee. "Internet of everything (IoE): Eye tracking data analysis". In: *Harnessing the Internet of Everything (IoE) for Accelerated Innovation Opportunities*. IGI Global, 2019, pp. 215–245.

[38] Fatima Hussain. *Internet of things: Building blocks and business models*. 978-3. Springer, 2017.

[39] Mahdi H Miraz et al. "Internet of nano-things, things and everything: future growth trends". In: *Future Internet* 10.8 (2018), p. 68.

[40] Pintu Kumar Sadhu, Venkata P Yanambaka, and Ahmed Abdelgawad. "Internet of things: Security and solutions survey". In: *Sensors* 22.19 (2022), p. 7433.

[41] Jong-Hoon Lee et al. "Flexible solid-state hybrid supercapacitors for the internet of everything (IoE)". In: *Energy & Environmental Science* (2022).

[42] Meltem Civas et al. "Universal transceivers: Opportunities and future directions for the internet of everything (IoE)". In: *arXiv preprint arXiv:2107.01028* (2021).

[43] Mohamed Elawady, Amany Sarhan, and Mahmoud AM Alshewimy. "Toward a mixed reality domain model for time-Sensitive applications using IoE infrastructure and edge computing (MRIoEF)". In: *The Journal of Supercomputing* 78.8 (2022), pp. 10656–10689.

[44] Anu Raj and Shiva Prakash. "Internet of Everything: A survey based on Architecture, Issues and Challenges". In: *2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*. 2018, pp. 1–6. DOI: 10.1109/UPCON.2018.8596923.

[45] Md Milon Islam et al. "Internet of Things: Device Capabilities, Architectures, Protocols, and Smart Applications in Healthcare Domain". In: *IEEE Internet of Things Journal* 10.4 (2022), pp. 3611–3641.

[46] Vasavi Avula et al. "The internet of everything: a survey". In: *2021 13th International Conference on Computational Intelligence and Communication Networks (CICN)*. IEEE. 2021, pp. 72–79.

[47] Favour Adenugba et al. "Smart irrigation system for environmental sustainability in Africa: An Internet of Everything (IoE) approach". In: *Mathematical biosciences and engineering* 16.5 (2019), pp. 5490–5503.

[48] Alex Vakaloudis and Christian O'Leary. "A framework for rapid integration of IoT Systems with industrial environments". In: *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*. IEEE. 2019, pp. 601–605.

[49] Anam Sajid, Haider Abbas, and Kashif Saleem. "Cloud-assisted IoT-based SCADA systems security: A review of the state of the art and future challenges". In: *IEEE Access* 4 (2016), pp. 1375–1384.

[50] Hasan Ali Khattak et al. "Perception layer security in Internet of Things". In: *Future Generation Computer Systems* 100 (2019), pp. 144–164. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2019.04.038. URL: https://www.sciencedirect.com/science/article/pii/S0167739X19304194.

[51] Sachin Kumar, Prayag Tiwari, and Mikhail Zymbler. "Internet of Things is a revolutionary approach for future technology enhancement: a review". In: *Journal of Big data* 6.1 (2019), pp. 1–21.

[52] A. H. Mohd Aman et al. "A Survey on Trend and Classification of Internet of Things Reviews". In: *IEEE Access* 8 (2020), pp. 111763–111782.

[53] Store Arduino Arduino. "Arduino". In: *Arduino LLC* (2015).

[54] Michael Margolis. *Arduino Cookbook - Recipes to Begin, Expand, an Enhance Your Projects:Covers Arduino 1.0 (2. ed.)* O'Reilly, 2012. ISBN: 978-1-449-31387-6. URL: http://www.oreilly.de/catalog/9781449313876/index.ht.

[55] Muhammad Sajjad et al. "Raspberry Pi assisted face recognition framework for enhanced law-enforcement services in smart cities". In: *Future Generation Computer Systems* 108 (2020), pp. 995–1007.

[56] Abhijit Bhowmik, Md Saef Ullah Miah, et al. "IoT (Internet of Things)-Based Smart Garbage Management System". In: *AIUB Journal of Science and Engineering (AJSE)* 19.1 (2020), pp. 33–40.

[57] Z. Zhang, Z. Pi, and B. Liu. "TROIKA: A General Framework for Heart Rate Monitoring Using Wrist-Type Photoplethysmographic Signals During Intensive Physical Exercise". In: *IEEE Transactions on Biomedical Engineering* 62.2 (2015), pp. 522–531.

[58] S. Becirovic and S. Mrdovic. "Manual IoT Forensics of a Samsung Gear S3 Frontier Smartwatch". In: *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. 2019, pp. 1–5. DOI: 10.23919/softcom.2019.8903845.

[59] M. A. Razzaque et al. "Middleware for Internet of Things: A Survey". In: *IEEE Internet of Things Journal* 3.1 (2016), pp. 70–95. DOI: 10.1109/JIOT.2015.2498900.

[60] Preeti Agarwal and Mansaf Alam. "IoT Cloud Platforms: an Application Development Perspective". In: *CoRR* abs/1810.12292 (2020). arXiv: 1810.12292. URL: http://arxiv.org/abs/1810.12292.

[61] Steve Buchanan and John Joyner. "Azure Arc-Enabled Kubernetes: Getting Started". In: *Azure Arc-Enabled Kubernetes and Servers*. Springer, 2022, 267:293.

[62] K. Sharma and R. Nandal. "A Literature Study On Machine Learning Fusion With IOT". In: *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*. 2019, pp. 1440–1445.

[63] Madhusanka Liyanage et al. *IoT security: Advances in authentication*. John Wiley and Sons, 2020.

[64] Fadele Ayotunde Alaba et al. "Internet of Things security: A survey". In: *Journal of Network and Computer Applications* 88 (2017), pp. 10–28. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2017.04.002. URL: shorturl.at/jtUV6.

[65] Jonathan de C Silva et al. "Management platforms and protocols for internet of things: A survey". In: *Sensors* 19.3 (2019), p. 676.

[66] Venetis Kanakaris and George A Papakostas. "Internet of things protocols-a survey". In: *International Journal of Humanitarian Technology* 1.2 (2020), pp. 101–117.

[67] Silvio Quincozes, Tubino Emilio, and Juliano Kazienko. "MQTT protocol: fundamentals, tools and future directions". In: *IEEE Latin America Transactions* 17.09 (2019), pp. 1439–1448.

[68] Biswajeeban Mishra and Attila Kertesz. "The use of MQTT in M2M and IoT systems: A survey". In: *IEEE Access* 8 (2020), pp. 201071–201086.

[69] M Veeramanikandan and Suresh Sankaranarayanan. "Publish/subscribe based multi-tier edge computational model in Internet of Things for latency reduction". In: *Journal of parallel and distributed computing* 127 (2019), pp. 18–27.

[70] Dan Dinculeană and Xiaochun Cheng. "Vulnerabilities and limitations of MQTT protocol used between IoT devices". In: *Applied Sciences* 9.5 (2019), p. 848.

[71] Muhammad Ashar Tariq et al. "Enhancements and challenges in coap—a survey". In: *Sensors* 20.21 (2020), p. 6391.

[72] Neven Nikolov. "Research of MQTT, CoAP, HTTP and XMPP IoT communication protocols for embedded systems". In: *2020 XXIX International Scientific Conference Electronics (ET)*. IEEE. 2020, pp. 1–4.

[73] Godfrey A Akpakwu, Gerhard P Hancke, and Adnan M Abu-Mahfouz. "CACC: Context-aware congestion control approach for lightweight CoAP/UDP-based Internet of Things traffic". In: *Transactions on Emerging Telecommunications Technologies* 31.2 (2020), e3822.

[74] Sharu Bansal and Dilip Kumar. "Distance-based congestion control mechanism for CoAP in IoT". In: *IET Communications* 14.19 (2020), pp. 3512–3520.

[75] Shih-Ping Hsu et al. "The Design and Implementation of a Lightweight CoAP-based IoT Framework with Smart Contract Security Guarantee". In: *2020 12th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*. IEEE. 2020, pp. 1–6.

[76] Eyhab Al-Masri et al. "Investigating messaging protocols for the Internet of Things (IoT)". In: *IEEE Access* 8 (2020), pp. 94880–94911.

[77] M Faiqurahman, MM Madani, and DR Akbi. "Performance of XMPP-Based Gateway for IoT Device Communication Services". In: *J. Teknol. dan Sist. Komput* 7.4 (2019), pp. 127–133.

[78] Chi-Shiang Cho et al. "Building on the distributed energy resources IoT based IEC 61850 XMPP for TPC". In: *2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS)*. IEEE. 2019, pp. 61–66.

[79] Carlos A Garcia, Jose E Naranjo, and Marcelo V Garcia. "Analysis of AMQP for Industrial Internet of Things Based on Low-Cost Automation". In: *Brazilian Technology Symposium*. Springer. 2019, pp. 235–244.

[80] Purvi Bhimani and Gaurang Panchal. "Message delivery guarantee and status update of clients based on IOT-AMQP". In: *Intelligent Communication and Computational Technologies*. Springer, 2018, pp. 15–22.

[81] Girija P Naik and A Umesh Bapat. "A Brief Comparative Analysis on Application Layer Protocols of Internet of Things: MQTT, CoAP, AMQP and HTTP". In: *Int. J. Comput. Sci. Mob. Comput* 9.9 (2020), pp. 135–141.

[82] Sherali Zeadally, Farhan Siddiqui, and Zubair Baig. "25 years of Bluetooth technology". In: *Future Internet* 11.9 (2019), p. 194.

[83] Kang Eun Jeon et al. "Ble beacons for internet of things applications: Survey, challenges, and opportunities". In: *IEEE Internet of Things Journal* 5.2 (2018), pp. 811–828.

[84] Yehia R Hamdy and Ahmed I Alghannam. "Evaluation of ZigBee topology effect on throughput and end to end delay due to different transmission bands for IoT applications". In: *Journal of communications software and systems* 16.3 (2020), pp. 254–259.

[85] Hamza Zemrane, Youssef Baddi, and Abderrahim Hasbi. "Internet of Things industry 4.0 ecosystem based on zigbee protocol". In: *Advances on Smart and Soft Computing*. Springer, 2021, pp. 249–260.

[86] Christopher W. Badenhop et al. "The Z-Wave routing protocol and its security implications". In: *Computers and Security* 68 (2017), pp. 112–129. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2017.04.004. URL: https://www.sciencedirect.com/science/article/pii/S0167404817300792.

[87] Ala Al-Fuqaha et al. "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications". In: *IEEE Communications Surveys and Tutorials* 17.4 (2015), pp. 2347–2376. DOI: 10.1109/COMST.2015.2444095.

[88] Hussam Kadhim. "Controlling Home and Office Appliances with the Bluetooth of Smartphone". In: *International Journal of Computer Applications* 152 (Oct. 2016), pp. 975–8887. DOI: 10.5120/ijca2016911821.

[89] Amira Zrelli. "Simultaneous Monitoring of Temperature, Pressure and Strain through Brillouin Sensors and a Hybrid BOTDA/FBG for Disasters Detection Systems". In: *IET Communications* 13 (Nov. 2019). DOI: 10.1049/iet-com.2018.5260.

[90] Amira Zrelli and Tahar Ezzedine. "Collect Tree Protocol for SHM system using wireless sensor networks". In: *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*. 2017, pp. 1797–1801. DOI: 10.1109/IWCMC.2017.7986556.

[91] Cenk Gündoğan et al. "Designing a LoWPAN convergence layer for the Information Centric Internet of Things". In: *Computer Communications* 164 (2020), pp. 114–123.

[92] Meet K Shah and LK Sharma. "Study on 6LoWPAN routing protocols with SD aspects in IoT". In: *2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC) I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), 2018 2nd International Conference on*. IEEE. 2018, pp. 60–65.

[93] František Zezulka et al. "Communication systems for industry 4.0 and the iiot". In: *IFAC-PapersOnLine* 51.6 (2018), pp. 150–155.

[94] IETF. *Multicast DNS*. 2013. URL: https://www.rfc-editor.org/info/rfc6762.

[95] IETF. *DNS-Based Service Discovery*. 2013. URL: https://www.rfc-editor.org/info/rfc6763.

[96] Leki Chom Thungon et al. "A lightweight authentication and key exchange mechanism for IPv6 over low-power wireless personal area networks-based Internet of things". In: *Transactions on Emerging Telecommunications Technologies* 32.5 (2021), e4033.

[97] David Airehrour, Jairo A Gutierrez, and Sayan Kumar Ray. "SecTrust-RPL: A secure trust-aware RPL routing protocol for Internet of Things". In: *Future Generation Computer Systems* 93 (2019), pp. 860–876.

[98] José V. V. Sobral et al. "Routing Protocols for Low Power and Lossy Networks in Internet of Things Applications". In: *Sensors* 19.9 (2019), p. 2144.

[99] x. "IEEE Standard for Low-Rate Wireless Networks". In: *IEEE Std 802.15.4-2020 (Revision of IEEE Std 802.15.4-2015)* (2020), pp. 1–800. DOI: 10.1109/IEEESTD.2020.9144691.

[100] J. P. García-Martín and A. Torralba. "On the Combination of LR-WPAN and LPWA Technologies to Provide a Collaborative Wireless Solution for Diverse IoT". In: 2019, pp. 1–4. DOI: 10.1109/WiMOB.2019.8923566.

[101] Upendra Singh et al. "A survey on LTE/LTE-A radio resource allocation techniques for machine-to-machine communication for B5G networks". In: *IEEE Access* 9 (2021), pp. 107976–107997.

[102] Een Kee Hong, Je Myung Ryu, and Elyse Jee Hyun Lee. "Entering the 5G Era". In: (2021).

[103] P.P. Ray. "A survey on Internet of Things architectures". In: *Journal of King Saud University - Computer and Information Sciences* 30.3 (2018), pp. 291–319. ISSN: 1319-1578. DOI: https://doi.org/10.1016/j.jksuci.2016.10.003. URL: shorturl.at/cdiZ3.

[104] Bo Cheng et al. "Situation-Aware IoT Service Coordination Using the Event-Driven SOA Paradigm". In: *IEEE Transactions on Network and Service Management* 13.2 (2016), pp. 349–361. DOI: 10.1109/TNSM.2016.2541171.

[105] Abu Sarwar Zamani et al. "An Appraise of Web Service based on SOA as a step Towards Cloud Computing, Big Data and IoT". In: *Annals of the Romanian Society for Cell Biology* 25.6 (2021), pp. 18720–18727.

[106] Zijiang Zhu et al. "Quality of e-commerce agricultural products and the safety of the ecological environment of the origin based on 5G Internet of Things technology". In: *Environmental Technology & Innovation* 22 (2021), p. 101462.

[107] Viet Minh Nhat Vo and Van Hoa Le. "Model of dynamic clustering-based energy-efficient data filtering for mobile RFID networks". In: *ETRI Journal* 43.3 (2021), pp. 427–435.

[108] Amee A. Patel and Sunil J. Soni. "A Novel Proposal for Defending against Vampire Attack in WSN". In: *2015 Fifth International Conference on Communication Systems and Network Technologies*. 2015, pp. 624–627. DOI: 10.1109/CSNT.2015.94.

[109] Bhagyashri Tushir et al. "A quantitative study of ddos and e-ddos attacks on wifi smart home devices". In: *IEEE Internet of Things Journal* 8.8 (2020), pp. 6282–6292.

[110] Mohamed Abdel-Basset, Gunasekaran Manogaran, and Mai Mohamed. "RETRACTED: Internet of Things (IoT) and its impact on supply chain: A framework for building smart, secure and efficient systems". In: *Future Generation Computer Systems* 86 (2018), pp. 614–628. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2018.04.051. URL: shorturl.at/nRTV1.

[111] João Santos et al. "An IoT-based mobile gateway for intelligent personal assistants on mobile health environments". In: *Journal of Network and Computer Applications* 71 (2016), pp. 194–204. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2016.03.014. URL: shorturl.at/rzMOV.

[112] Shailendra Rathore and Jong Park. "Semi-supervised learning based distributed attack detection framework for IoT". In: *Applied Soft Computing* 72 (July 2018). DOI: 10.1016/j.asoc.2018.05.049.

[113] Anca Jurcut et al. "Security Considerations for Internet of Things: A Survey". In: *SN Computer Science* 1 (4 2020). ISSN: 2662-995X. DOI: 10.1007/s42979-020-00201-3.

[114] A. Hameed and A. Alomary. "Security Issues in IoT: A Survey". In: *2019 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*. 2019, pp. 1–5.

[115] Fadele Ayotunde Alaba et al. "Internet of Things security: A survey". In: *Journal of Network and Computer Applications* 88 (2017), pp. 10–28. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2017.04.002.

[116] Sabrina Sicari et al. "Security, privacy and trust in Internet of Things: The road ahead". In: *Comput. Networks* 76 (2015), pp. 146–164.

[117] Jun Zhou et al. "Security and Privacy for Cloud-Based IoT: Challenges". In: *IEEE Communications Magazine* 55.1 (2017), pp. 26–33.

[118] Muhammad Shadi Hajar, M Omar Al-Kadri, and Harsha Kumara Kalutarage. "A survey on wireless body area networks: architecture, security challenges and research opportunities". In: *Computers & Security* 104 (2021), p. 102211.

[119] IOT. *Internet of Things - Architecture IoT-A*. 2013. URL: https://cordis.europa.eu/project/id/257521.

[120] ENISA. *Good Practices for Security of Internet of Things in the context of Smart Manufacturing*. Nov. 2018. URL: shorturl.at/glsX0.

[121] Ross Anderson. *Security engineering: a guide to building dependable distributed systems*. John Wiley and Sons, 2020.

[122] The National Cyber Security Centre. *The Cyber Security Body of Knowledge*. Oct. 2019. URL: shorturl.at/fgiU1.

[123] Bilal Shabandri and Piyush Maheshwari. "Enhancing IoT Security and Privacy Using Distributed Ledgers with IOTA and the Tangle". In: 2019, pp. 1069–1075. DOI: 10.1109/SPIN.2019.8711591.

[124] Umair Riaz et al. "A novel embedded system design for the detection and classification of cardiac disorders". In: *Comput. Intell.* 37.4 (2021), pp. 1844–1864. DOI: 10.1111/coin.12469. URL: https://doi.org/10.1111/coin.12469.

[125] Tapalina Bhattasali and Rituparna Chaki. "A Survey of Recent Intrusion Detection Systems for Wireless Sensor Network". In: *CoRR* abs/1203.0240 (2012).

[126] M. Frustaci et al. "Evaluating Critical Security Issues of the IoT World: Present and Future Challenges". In: *IEEE* 5.4 (2018), pp. 2483–2495. DOI: 10.1109/JIOT.2017.2767291.

[127] All. *The 5 Worst Examples of Iot Hacking and Vulnerabilities in Recorded History*. 2017. URL: shorturl.at/mprW3. (accessed: 25.02.2022).

[128] Mauro Conti et al. "SPLIT: A Secure and Scalable RPL routing protocol for Internet of Things". In: IEEE. 2018, pp. 1–8.

[129] Tiago Gomes et al. "A 6LoWPAN accelerator for Internet of Things endpoint devices". In: *IEEE Internet of Things Journal* 5.1 (2017), pp. 371–377.

[130] Murat Demirbas and Youngwhan Song. "An RSSI-based Scheme for Sybil Attack Detection in Wireless Sensor Networks". In: *WOWMOM*. IEEE Computer Society, 2006, pp. 564–570.

[131] René Hummen et al. "6LoWPAN fragmentation attacks and mitigation mechanisms". In: *WISEC*. ACM, 2013, pp. 55–66.

[132] Aamir Hussain et al. "Security framework for IoT based real-time health applications". In: *Electronics* 10.6 (2021), p. 719.

[133] CyberMDX. *cyber attack*. 2017. URL: shorturl.at/gK146. (accessed: 25.02.2022).

[134] Cert analysis. *Analysis Cert analysis on iot botnet and ddos attack*. 2016. URL: shorturl.at/yDEQS. (accessed: 25.02.2022).

[135] OWSAP. *OWASP, Top IoT Vulnerabilities*. 2020.

[136] Mookyu Park, Haengrok Oh, and Kyungho Lee. "Security risk measurement for information leakage in IoT-based smart homes from a situational awareness perspective". In: *Sensors* 19.9 (2019), p. 2148.

[137] Lo'ai Tawalbeh et al. "IoT Privacy and security: Challenges and solutions". In: *Applied Sciences* 10.12 (2020), p. 4102.

[138] ENISA. *Baseline Security Recommendations for IoT*. Nov. 2017. URL: shorturl.at/ijlr1.

[139] John Matherly. “Complete Guide to Shodan: Collect. Analyze. Visualize. Make Internet Intelligence Work for You”. In: *Leanpub* (2016).

[140] Granville. *Facebook and Cambridge Analytica what You Need to Know as Fallout Widens*. 2017. URL: shorturl.at/jrwz0. (accessed: 25.02.2022).

[141] Maxwell Young and Raouf Boutaba. “Overcoming Adversaries in Sensor Networks: A Survey of Theoretical Models and Algorithmic Approaches for Tolerating Malicious Interference”. In: *IEEE Commun. Surv. Tutorials* 13.4 (2011), pp. 617–641.

[142] Sudip Misra, Ranjit Singh, and S. V. Rohith Mohan. “Information Warfare-Worthy Jamming Attack Detection Mechanism for Wireless Sensor Networks Using a Fuzzy Inference System.” In: *Sensors* 10.4 (2010), pp. 3444–3479. DOI: http://dx.doi.org/10.3390/s100403444.

[143] Matthew Pirretti et al. “The Sleep Deprivation Attack in Sensor Networks: Analysis and Methods of Defense”. In: *IJDSN* 2 (Sept. 2006), pp. 267–287. DOI: 10.1080/15501320600642718.

[144] Tapalina Bhattasali, Rituparna Chaki, and Sugata Sanyal. “Sleep Deprivation Attack Detection in Wireless Sensor Network”. In: *International Journal of Computer Applications* 40 (Feb. 2012), pp. 19–25. DOI: 10.5120/5056-7374.

[145] Aliya Tabassum and Wadha Lebda. “Security Framework for IoT Devices against Cyber-attacks”. In: *SFI*. Nov. 2019, p. 5. DOI: 10.5121/csit.2019.91321.

[146] Y. Wang et al. “Intrusion Detection in Homogeneous and Heterogeneous Wireless Sensor Networks”. In: *IEEE Transactions on Mobile Computing* 7.6 (2008), pp. 698–711. DOI: 10.1109/TMC.2008.19.

[147] Ana Paula Silva et al. “Decentralized intrusion detection in wireless sensor networks”. In: *DIDIW*. Jan. 2005, pp. 16–23. DOI: 10.1145/1089761.1089765.

[148] E. Anthi et al. “A Supervised Intrusion Detection System for Smart Home IoT Devices”. In: *IEEE Internet of Things Journal* 6.5 (2019), pp. 9042–9053. DOI: 10.1109/JIOT.2019.2926365.

[149] Tommaso Pecorella, Luca Brilli, and Lorenzo Mucchi. “The Role of Physical Layer Security in IoT: A Novel Perspective”. In: *Inf.* 7.3 (2016), p. 49.

[150] S. Soderi et al. “Physical layer security based on spread-spectrum watermarking and jamming receiver”. In: *Transactions on Emerging Telecommunications Technologies* 28.7 (2017).

[151] M. Conti, N. Dragoni, and V. Lesyk. “A Survey of Man In The Middle Attacks”. In: *IEEE Communications Surveys Tutorials* 18.3 (2016), pp. 2027–2051. DOI: 10.1109/COMST.2016.2548426.

[152] P.N. Mahalle et al. “Identity authentication and capability based access control (IACAC) for the internet of things”. In: *J. Cyber Security Mobility* 1 (Oct. 2012), pp. 309–348.

[153] OWASP. *OWASP Web Security Testing Guide*. 2020. URL: https://owasp.org.

[154] Zhiping Jiang et al. “PHYAlert: identity spoofing attack detection and prevention for a wireless edge network”. In: *Journal of Cloud Computing* 9 (Dec. 2020). DOI: 10.1186/s13677-020-0154-7.

[155] W. Tiberti et al. “A Lightweight Blockchain-based Technique for Anti-Tampering in Wireless Sensor Networks”. In: *2020 23rd Euromicro Conference on Digital System Design (DSD)*. 2020, pp. 577–582. DOI: 10.1109/DSD51259.2020.00095.

[156] D. R. Raymond and S. F. Midkiff. “Denial-of-Service in Wireless Sensor Networks: Attacks and Defenses”. In: *IEEE Pervasive Computing* 7.1 (2008), pp. 74–81. DOI: 10.1109/MPRV.2008.6.

[157] J. Deogirikar and A. Vidhate. “Security attacks in IoT: A survey”. In: *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*. 2017, pp. 32–37. DOI: 10.1109/I-SMAC.2017.8058363.

[158] S. Soderi et al. “Near-field measurements for safety related systems and jamming attack”. In: *Progress In Electromagnetics Research B* 62.1 (2015), pp. 289–302.

[159] C. Cervantes et al. “Detection of sinkhole attacks for supporting secure routing on 6LoWPAN for Internet of Things”. In: *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. 2015, pp. 606–611. DOI: 10.1109/INM.2015.7140344.

[160] K. Saghar et al. “RAEED: A formally verified solution to resolve sinkhole attack in Wireless Sensor Network”. In: *2016 13th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. 2016, pp. 334–345. DOI: 10.1109/IBCAST.2016.7429899.

[161] Md Abdullah, M. Rahman, and Mukul Roy. “Detecting Sinkhole Attacks in Wireless Sensor Network using Hop Count”. In: *International Journal of Computer Network and Information Security* Vol. 7 (Feb. 2015), pp. 50–56. DOI: 10.5815/ijcnis.2015.03.07.

[162] S. Gupta, S. Kar, and S. Dharmaraja. “WHOP: Wormhole attack detection protocol using hound packet”. In: *2011 International Conference on Innovations in Information Technology*. 2011, pp. 226–231. DOI: 10.1109/INNOVATIONS.2011.5893822.

[163] Yih-Chun Hu, A. Perrig, and D. B. Johnson. "Wormhole attacks in wireless networks". In: *IEEE Journal on Selected Areas in Communications* 24.2 (2006), pp. 370–380. DOI: 10.1109/JSAC.2005.861394.

[164] Gu-Hsin Lai. "Detection of wormhole attacks on IPv6 mobility-based wireless sensor network". In: *EURASIP Journal on Wireless Communications and Networking* 2016 (Nov. 2016). DOI: 10.1186/s13638-016-0776-0.

[165] Rabia Riaz, Ki-Hyung Kim, and H. Farooq Ahmed. "Security analysis survey and framework design for IP connected LoWPANs". In: *ISADS*. IEEE Computer Society, 2009, pp. 29–34.

[166] Syrine Sahmim and Hamza Gharsellaoui. "Privacy and Security in Internet-based Computing: Cloud Computing, Internet of Things, Cloud of Things: a review". In: *Procedia Computer Science* 112 (2017). Knowledge-Based and Intelligent Information and Engineering Systems: Proceedings of the 21st International Conference, KES-20176-8 September 2017, Marseille, France, pp. 1516–1522. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2017.08.050.

[167] L. Ma et al. "Research on SQL Injection Attack and Prevention Technology Based on Web". In: *2019 International Conference on Computer Network, Electronic and Automation (ICCNEA)*. 2019, pp. 176–179. DOI: 10.1109/ICCNEA.2019.00042.

[168] M. Baykara and Z. Z. Gürel. "Detection of phishing attacks". In: *2018 6th International Symposium on Digital Forensic and Security (ISDFS)*. 2018, pp. 1–5. DOI: 10.1109/ISDFS.2018.8355389.

[169] D. Yin, L. Zhang, and K. Yang. "A DDoS Attack Detection and Mitigation With Software-Defined Internet of Things Framework". In: *IEEE Access* 6 (2018), pp. 24694–24705. DOI: 10.1109/ACCESS.2018.2831284.

[170] Chiara Bodei, Stefano Chessa, and Letterio Galletta. "Measuring security in IoT communications". In: *Theoretical Computer Science* 764 (2019). Selected papers of ICTCS 2016 (The Italian Conference on Theoretical Computer Science (ICTCS), pp. 100–124. ISSN: 0304-3975. DOI: https://doi.org/10.1016/j.tcs.2018.12.002.

[171] N. Namvar et al. "Jamming in the Internet of Things: A Game-Theoretic Perspective". In: *2016 IEEE Global Communications Conference (GLOBECOM)*. 2016, pp. 1–6.

[172] Abdul Wahab Ahmed et al. "A comprehensive analysis on the security threats and their countermeasures of IoT". In: *International Journal of Advanced Computer Science and Applications* 8.7 (2017), pp. 489–501.

[173] David Airehrour, J. Gutierrez, and Sayan Ray. "A trust-aware RPL routing protocol to detect blackhole and selective forwarding attacks". In: *Australian Journal of Telecommunications and the Digital Economy* 5 (Mar. 2017). DOI: 10.18080/ajtde.v5n1.2.

[174] IETF. *Host Identity Protocol Version 2 (HIPv2)*. 2015. URL: shorturl.at/ipVX1.

[175] Seong Ho Chae et al. "Enhanced Secrecy in Stochastic Wireless Networks: Artificial Noise With Secrecy Protected Zone". In: *IEEE Trans. Information Forensics and Security* 9.10 (2014), pp. 1617–1628.

[176] Yao-Win Peter Hong, Pang-Chang Lan, and C.-C. Jay Kuo. "Enhancing Physical-Layer Secrecy in Multiantenna Wireless Systems: An Overview of Signal Processing Approaches". In: *IEEE Signal Process. Mag.* 30.5 (2013), pp. 29–40.

[177] Qing Li and Wade Trappe. "Light-weight Detection of Spoofing Attacks in Wireless Networks". In: *2006 IEEE International Conference on Mobile Ad Hoc and Sensor Systems*. 2006, pp. 845–851. DOI: 10.1109/MOBHOC.2006.278663.

[178] Kevin Weekly and Kristofer Pister. "Evaluating sinkhole defense techniques in RPL networks". In: *2012 20th IEEE International Conference on Network Protocols (ICNP)*. 2012, pp. 1–6. DOI: 10.1109/ICNP.2012.6459948.

[179] X. "Circumventing sinkholes and wormholes in wireless sensor networks". In: *Conference on Wireless Ad Hoc Networks* (2005).

[180] Indranil Saha and Debapriyay Mukhopadhyay. "Security against Sybil Attack in Wireless Sensor Network through Location Verification". In: *Distributed Computing and Networking*. Ed. by Vijay Garg, Roger Wattenhofer, and Kishore Kothapalli. 2009, pp. 187–192. DOI: 10.1007/978-3-540-92295-7_23.

[181] Pim Otte, Martijn de Vos, and J.A. Pouwelse. "TrustChain: A Sybil-resistant scalable blockchain". In: *Future Generation Computer Systems* (Sept. 2017). DOI: 10.1016/j.future.2017.08.048.

[182] Yinghong Liu and Yuanming Wu. "An Enhanced RSSI-Based Detection Scheme for Sybil Attack in Wireless Sensor Networks". In: *Advances in Information and Communication*. Ed. by Kohei Arai and Rahul Bhatia. Cham: Springer International Publishing, 2020, pp. 87–102. ISBN: 978-3-030-12388-8.

[183] H. Kim. "Protection Against Packet Fragmentation Attacks at 6LoWPAN Adaptation Layer". In: *2008 International Conference on Convergence and Hybrid Information Technology*. 2008, pp. 796–801.

[184] K. Nirmal, B. Janet, and Rajit Kumar. "Analyzing and eliminating phishing threats in IoT, network and other Web applications using iterative intersection". In: *Peer-to-Peer Networking and Applications* (June 2020), pp. 1–13. DOI: 10.1007/s12083-020-00944-z.

[185] Jayant Bokefode et al. "Developing A Secure Cloud Storage System for Storing IoT Data by Applying Role Based Encryption". In: *Procedia Computer Science* 89 (Dec. 2016), pp. 43–50. DOI: 10.1016/j.procs.2016.06.007.

[186] R. Doshi, N. Apthorpe, and N. Feamster. "Machine Learning DDoS Detection for Consumer Internet of Things Devices". In: *2018 IEEE Security and Privacy Workshops (SPW)*. 2018, pp. 29–35.

[187] A. Roohi, M. Adeel, and M. A. Shah. "DDoS in IoT: A Roadmap Towards Security Countermeasures". In: *2019 25th International Conference on Automation and Computing (ICAC)*. 2019, pp. 1–6. DOI: 10.23919/IConAC.2019.8895034.

[188] Y. Afek, A. Bremler-Barr, and S. L. Feibish. "Zero-Day Signature Extraction for High-Volume Attacks". In: *IEEE/ACM Transactions on Networking* 27.2 (2019), pp. 691–706. DOI: 10.1109/TNET.2019.2899124.

[189] Knud Skouby, Reza Tadayoni, and Samuel Tweneboah-Koduah. "Cyber Security Threats to IoT Applications and Service Domains". In: *Wireless Personal Communications* (May 2017). DOI: 10.1007/s11277-017-4434-6.

[190] Mehiar Dabbagh and Ammar Rayes. "Internet of things security and privacy". In: *Internet of Things from hype to reality*. Springer, 2019, pp. 211–238.

[191] Jitendra Patil and Manish Sharma. "Survey of prevention techniques for denial service attacks (DoS) in wireless sensor network". In: *Int. J. Sci. Res.(IJSR), ISSN* (2016), pp. 2319–7064.

[192] Eman Ali Metwally, Noha A Haikal, and Hassan Hussein Soliman. "Detecting Semantic Social Engineering Attack in the Context of Information Security". In: *Digital Transformation Technology*. -: Springer, 2022, pp. 43–65.

[193] Zhiyi Zhang et al. "Sovereign: Self-contained Smart Home with Data-centric Network and Security". In: *IEEE Internet of Things Journal* (2022), pp. 1–1.

[194] Ibrahim Halil Saruhan. "Detecting and Preventing Rogue Deviceson the Network". In: *SANS Institute* (2007).

[195] Mehndi Samra et al. "Detection and Mitigation of Rogue Access Point". In: 1 (Jan. 2015), pp. 195–198.

[196] Firat Kilincer, Fatih Ertam, and Abdulkadir Sengur. "Automated Fake Access Point Attack Detection and Prevention System with IoT Devices". In: *Balkan Journal of Electrical and Computer Engineering* 8 (Jan. 2020), pp. 50–56. DOI: 10.17694/bajece.634104.

[197] Zhanyong Tang et al. “Exploiting Wireless Received Signal Strength Indicators to Detect Evil-Twin Attacks in Smart Homes”. In: *Mob. Inf. Syst.* 2017 (2017), 1248578:1–1248578:14. DOI: 10.1155/2017/1248578.

[198] Georgios Kambourakis, Constantinos Kolias, and Angelos Stavrou. “The Mirai botnet and the IoT Zombie Armies”. In: *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*. USA: IEEE, 2017, pp. 267–272. DOI: 10.1109/MILCOM.2017.8170867.

[199] Artur Marzano et al. “The Evolution of Bashlite and Mirai IoT Botnets”. In: *2018 IEEE Symposium on Computers and Communications (ISCC)*. Brazil: IEEE, June 2018, pp. 00813–00818. DOI: 10.1109/ISCC.2018.8538636.

[200] Bhagyashri Tushir et al. “A Quantitative Study of DDoS and E-DDoS Attacks on WiFi Smart Home Devices”. In: *IEEE Internet of Things Journal* 8.8 (2021), pp. 6282–6292. DOI: 10.1109/JIOT.2020.3026023.

[201] C. Kolias et al. “DDoS in the IoT: Mirai and Other Botnets”. In: *Computer* 50.07 (July 2017), pp. 80–84. ISSN: 1558-0814. DOI: 10.1109/MC.2017.201.

[202] Chibiao Liu and Jinming Qiu. “Performance Study of 802.11w for Preventing DoS Attacks on Wireless Local Area Networks”. In: *Wirel. Pers. Commun.* 95.2 (2017), pp. 1031–1053. DOI: 10.1007/s11277-016-3812-9. URL: https://doi.org/10.1007/s11277-016-3812-9.

[203] Kira Bobrovnikova et al. “Technique for IoT cyberattacks detection based on the energy consumption analysis”. In: *CEUR Workshop Proceedings*. Vol. 2853. 2021.

[204] Valentina Fabi, Giorgia Spigliantini, and Stefano Paolo Corgnati. “Insights on Smart Home Concept and Occupants’ Interaction with Building Controls”. In: *Energy Procedia* 111 (2017). 8th International Conference on Sustainability in Energy and Buildings, SEB-16, 11-13 September 2016, Turin, Italy, pp. 759–769. ISSN: 1876-6102. DOI: https://doi.org/10.1016/j.egypro.2017.03.238. URL: https://www.sciencedirect.com/science/article/pii/S1876610217302680.

[205] Rebecca Ford et al. “Categories and functionality of smart home technology for energy management”. In: *Building and Environment* 123 (2017), pp. 543–554. ISSN: 0360-1323. DOI: https://doi.org/10.1016/j.buildenv.2017.07.020. URL: https://www.sciencedirect.com/science/article/pii/S0360132317303062.

[206] Yang Shi et al. “Energy Audition Based Cyber-Physical Attack Detection System in IoT”. In: *Proceedings of the ACM Turing Celebration ConferenceChina*. ACM TURC ’19. Chengdu, China: Association for Computing Machinery, 2019. ISBN: 9781450371582. DOI: 10.1145/3321408.3321588. URL: https://doi.org/10.1145/3321408.3321588.

[207] Laurina Felius, Fredrik Dessen, and Bozena Hrynyszyn. "Correction to: Retrofitting towards energy-efficient homes in European cold climates: a review". In: *Energy Efficiency* 13 (Jan. 2020). DOI: 10.1007/s12053-019-09838-3.

[208] Johannes Hoffmann, Stephan Neumann, and Thorsten Holz. "Mobile Malware Detection Based on Energy Fingerprints A Dead End". In: Oct. 2013, pp. 348–368.

[209] Kira Bobrovnikova et al. "Technique for IoT cyberattacks detection based on the energy consumption analysis". In: *CEUR Workshop Proceedings*. Vol. 2853. 2021.

[210] Antonio J. Jara, Latif Ladid, and Antonio Skarmeta. "The Internet of Everything through IPv6: an analysis of challenges, solutions and opportunities". In: *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications* 4 (Sept. 2013), pp. 97–118.

[211] Himanshu Sharma and Neeraj Kumar. "Deep learning based physical layer security for terrestrial communications in 5G and beyond networks: A survey". In: *Physical Communication* (2023), p. 102002.

[212] Jungwoo Ryoo et al. "IoE Security Threats and You". In: *2017 International Conference on Software Security and Assurance (ICSSA)*. 2017, pp. 13–19. DOI: 10.1109/ICSSA.2017.28.

[213] Rami Sihwail, Khairuddin Omar, and Khairul Akram Zainol Arifin. "An Effective Memory Analysis for Malware Detection and Classification." In: *Computers, Materials & Continua* 67.2 (2021).

[214] Stefan Vomel and Felix C Freiling. "A survey of main memory acquisition and analysis techniques for the windows operating system". In: *Digital Investigation* 8.1 (2011), pp. 3–22.

[215] Chathuranga Rathnayaka and Aruna Jamdagni. "An efficient approach for advanced malware analysis using memory forensic technique". In: *2017 IEEE Trustcom/BigDataSE/ICESS*. IEEE. 2017, pp. 1145–1150.

[216] Ahmed Zaki and Benjamin Humphrey. "Unveiling the kernel: Rootkit discovery using selective automated kernel memory differencing". In: *Virus Bulletin* (2014), pp. 239–256.

[217] Masoume Aghaeikheirabady, Seyyed Mohammad Reza Farshchi, and Hossein Shirazi. "A new approach to malware detection by comparative analysis of data structures in a memory image". In: *2014 International Congress on Technology, Communication and Knowledge (ICTCK)*. IEEE. 2014, pp. 1–4.

[218] Rayan Mosli et al. "Automated malware detection using artifacts in forensic memory images". In: *2016 IEEE Symposium on Technologies for Homeland Security (HST)*. IEEE. 2016, pp. 1–6.

[219] Rayan Mosli et al. “A behavior-based approach for malware detection”. In: *IFIP International Conference on Digital Forensics*. Springer. 2017, pp. 187–201.

[220] Yiheng Duan et al. “Detective: Automatically identify and analyze malware processes in forensic scenarios via DLLs”. In: *2015 IEEE International Conference on Communications (ICC)*. IEEE. 2015, pp. 5691–5696.

[221] Yusheng Dai et al. “A malware classification method based on memory dump grayscale image”. In: *Digital Investigation* 27 (2018), pp. 30–37.

[222] Rami Sihwail et al. “Malware detection approach based on artifacts in memory image and dynamic analysis”. In: *Applied Sciences* 9.18 (2019), p. 3680.

[223] Pooja Kumari and Ankit Kumar Jain. “A Comprehensive Study of DDoS Attacks over IoT Network and Their Countermeasures”. In: *Computers & Security* (2023), p. 103096.

[224] Cisco Visual Networking Index (VNI). *Cisco Visual Networking Index (VNI)*. -. 2021. URL: shorturl.at/abkSU.

[225] Zaied Shouran, Ahmad Ashari, and Tri Kuntoro. “Internet of Things (IoT) of Smart Home: Privacy and Security”. In: *International Journal of Computer Applications* 182.39 (2019).

[226] Daojing He et al. “Privacy in the internet of things for smart healthcare”. In: *IEEE Communications Magazine* 56.4 (2018), pp. 38–44.

[227] Rachida Hireche, Houssem Mansouri, and Al-Sakib Khan Pathan. “Security and Privacy Management in Internet of Medical Things (IoMT): A Synthesis”. In: *Journal of Cybersecurity and Privacy* 2.3 (2022), pp. 640–661.

[228] Virginia Morgan, Milos Birtus, and Anna Zauskova. “Medical Internet of Things-based Healthcare Systems, Wearable Biometric Sensors, and Personalized Clinical Care in Remotely Monitoring and Caring for Confirmed or Suspected COVID-19 Patients.” In: *American Journal of Medical Research* 8.1 (2021), pp. 81–91.

[229] Jahanzeb Shahid et al. “Data protection and privacy of the internet of healthcare things (IoHTs)”. In: *Applied Sciences* 12.4 (2022), p. 1927.

[230] Tejasvi Alladi et al. “Consumer IoT: Security Vulnerability Case Studies and Solutions”. In: *IEEE Consumer Electronics Magazine* 9.2 (2020), pp. 17–25. DOI: 10.1109/MCE.2019.2953740.

[231] Ibrar Yaqoob et al. “The rise of ransomware and emerging security challenges in the Internet of Things”. In: *Computer Networks* 129 (2017). Special Issue on 5G Wireless Networks for IoT and Body Sensors, pp. 444–458. ISSN: 1389-1286. DOI: shorturl.at/imzM4. URL: shorturl.at/acG26.

[232] Michael Opoku Agyeman, Zainab Al-Waisi, and Igla Hoxha. "Design and Implementation of an IoT-Based Energy Monitoring System for Managing Smart Homes". In: *2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC)*. FMEC, 2019, pp. 253–258. DOI: 10.1109/FMEC.2019.8795363.

[233] Ramana R Avula and Tobias J Oechtering. "Privacy-Enhancing Appliance Filtering For Smart Meters". In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2022, pp. 9042–9046.

[234] Andrea Zanella et al. "Internet of Things for Smart Cities". In: *IEEE Internet of Things Journal* 1.1 (2014), pp. 22–32. DOI: 10.1109/JIOT.2014.2306328.

[235] Luis Puche Rondon et al. "Survey on enterprise Internet-of-Things systems (E-IoT): A security perspective". In: *Ad Hoc Networks* 125 (2022), p. 102728.

[236] Anas Dawod et al. "IoT device integration and payment via an autonomic blockchain-based service for IoT device sharing". In: *Sensors* 22.4 (2022), p. 1344.

[237] Paolo Bellavista et al. "Convergence of MANET and WSN in IoT Urban Scenarios". In: *IEEE Sensors Journal* 13.10 (2013), pp. 3558–3567. DOI: 10.1109/JSEN.2013.2272099.

[238] B Barani Sundaram et al. "Analysis of Machine Learning Data Security in the Internet of Things (IoT) Circumstance". In: *Expert Clouds and Applications*. Springer, 2022, pp. 227–236.

[239] Jinyuan Xu, Baoxing Gu, and Guangzhao Tian. "Review of agricultural IoT technology". In: *Artificial Intelligence in Agriculture* (2022).

[240] Kenneth Li Minn Ang, Jasmine Kah Phooi Seng, and Ericmoore Ngharamike. "Towards crowdsourcing internet of things (crowd-iot): Architectures, security and applications". In: *Future Internet* 14.2 (2022), p. 49.

[241] Minhaj Ahmad Khan and Khaled Salah. "IoT security: Review, blockchain solutions, and open challenges". In: *Future generation computer systems* 82 (2018), pp. 395–411.

[242] Deepak M Birajdar and Sharwari S Solapure. "LEACH: An energy efficient routing protocol using Omnet++ for Wireless Sensor Network". In: *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*. IEEE. 2017, pp. 465–470.

[243] Maurizka Ainur Rahmadhani, Leanna Vidya Yovita, and Ratna Mayasari. "Energy consumption and packet loss analysis of LEACH routing protocol on WSN over DTN". In: *2018 4th International Conference on Wireless and Telematics (ICWT)*. IEEE. 2018, pp. 1–5.

[244] Gaurav Pattewar et al. “Management of IoT Devices Security Using Blockchain—A Review”. In: *Sentimental Analysis and Deep Learning* (2022), pp. 735–743.

[245] Energy Information Administration (EIA). *Cisco Visual Networking Index (VNI)*. International Energy Outlook. 2019. URL: https://www.eia.gov/outlooks/ieo/.

[246] Bhagyashri Tushir et al. “The impact of dos attacks onresource-constrained iot devices: A study on the mirai attack”. In: *arXiv preprint arXiv:2104.09041* (2021).

[247] Surapon Kraijak and Panwit Tuwanut. “A survey on IoT architectures, protocols, applications, security, privacy, real-world implementation and future trends”. In: *11th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2015)*. 2015, pp. 1–6. DOI: 10.1049/cp.2015.0714.

[248] Md Islam et al. “Internet of Things Device Capabilities, Architectures, Protocols, and Smart Applications in Healthcare Domain: A Review”. In: *arXiv preprint arXiv:2204.05921* (2022).

[249] Shariq Murtuza. “Internet of Everything: Application and Various Challenges Analysis a Survey”. In: *2022 1st International Conference on Informatics (ICI)*. 2022, pp. 250–252. DOI: 10.1109/ICI53355.2022.9786891.

[250] Bushra Jamil et al. “Resource Allocation and Task Scheduling in Fog Computing and Internet of Everything Environments: A Taxonomy, Review, and Future Directions”. In: *ACM Computing Surveys (CSUR)* (2022).

[251] Jinsong Zhan, Shaofeng Dong, and Wei Hu. “IoE-supported smart logistics network communication with optimization and security”. In: *Sustainable Energy Technologies and Assessments* 52 (2022), p. 102052.

[252] Andres Robles-Durazno et al. “PLC memory attack detection and response in a clean water supply system”. In: *International Journal of Critical Infrastructure Protection* 26 (2019), p. 100300.

[253] Wei Shi et al. “Intelligent Reflection Enabling Technologies for Integrated and Green Internet-of-Everything Beyond 5G: Communication, Sensing, and Security”. In: *IEEE Wireless Communications* (2022).

[254] Rinki Rani et al. “Towards green computing oriented security: A lightweight postquantum signature for IoE”. In: *Sensors* 21.5 (2021), p. 1883.

[255] Haiyang Zhang et al. “Near-Field Wireless Power Transfer for 6G Internet of Everything Mobile Networks: Opportunities and Challenges”. In: *IEEE Communications Magazine* 60.3 (2022), pp. 12–18.

[256] DDoS attack. *DDoS attack that disrupted Internet was largest of its kind in history, experts say*. DDoS attack. 2016. URL: https://www.theguardian.com/technology/2016/oct/%2026/ddos-attack-dyn-mirai-botnet.

[257] Atul B Kathole et al. "Energy-Aware UAV Based on Blockchain Model Using IoE Application in 6G Network-Driven Cybertwin". In: *Energies* 15.21 (2022), p. 8304.

[258] Lijun Wei et al. "The convergence of ioe and blockchain: Security challenges". In: *IT Professional* 21.5 (2019), pp. 26–32.

[259] Afnan Alotaibi and Murad A Rassam. "Adversarial machine learning attacks against intrusion detection systems: A survey on strategies and defense". In: *Future Internet* 15.2 (2023), p. 62.

[260] Mohammad Abu Alsheikh et al. "Machine learning in wireless sensor networks: Algorithms, strategies, and applications". In: *IEEE Communications Surveys & Tutorials* 16.4 (2014), pp. 1996–2018.

[261] Mehdi Moradi and Mohammad Zulkernine. "A neural network based system for intrusion detection and classification of attacks". In: *Proceedings of the IEEE international conference on advances in intelligent systems-theory and applications*. IEEE Lux-embourg-Kirchberg, Luxembourg. 2004, pp. 15–18.

[262] Alan Bivens et al. "Network-based intrusion detection using neural networks". In: *Intelligent Engineering Systems through Artificial Neural Networks* 12.1 (2002), pp. 579–584.

[263] Adir Krayden et al. "CMOS-MEMS Gas Sensor Dubbed GMOS for SelectiveAnalysis of Gases with Tiny Edge Machine Learning". In: *Engineering Proceedings* 27.1 (2022), p. 81.

[264] Ahmed Hussain et al. "Jamming Detection in IoT Wireless Networks: An Edge-AI Based Approach". In: *Proceedings of the 12th International Conference on the Internet of Things*. 2022, pp. 57–64.

[265] Quansong Qi, Zhiyong Xu, and Pratibha Rani. "Big data analytics challenges to implementing the intelligent Industrial Internet of Things (IIoT) systems in sustainable manufacturing operations". In: *Technological Forecasting and Social Change* 190 (2023), p. 122401.

[266] Ted Szymanski. "An Ultra-Reliable Quantum-Safe Software-Defined Deterministic Internet of Things (IoT) for Data-Centers, Cloud Computing and the Metaverse". In: (2023).

[267] Mazen Juma, Fuad Alattar, and Basim Touqan. "Securing Big Data Integrity for Industrial IoT in Smart Manufacturing Based on the Trusted Consortium Blockchain (TCB)". In: *IoT* 4.1 (2023), pp. 27–55.

[268] J. M. Hamamreh, Muhammad Furqan, and Huseyin Arslan. "Classifications and Applications of Physical Layer Security Techniques for Confidentiality: A Comprehensive Survey". In: *IEEE Communications Surveys and Tutorials* PP (Oct. 2018), pp. 1–1. DOI: 10.1109/COMST.2018.2878035.

[269] Yuhan Jiang and Yulong Zou. "Secrecy Energy Efficiency Maximization for Multi-User Multi-Eavesdropper Cell-Free Massive MIMO Networks". In: *IEEE Transactions on Vehicular Technology* (2023).

[270] Abraham Sanenga et al. "An Overview of Key Technologies in Physical Layer Security". In: *Entropy* 22 (Nov. 2020), p. 1261. DOI: 10.3390/e22111261.

[271] Nora Abdelsalam, Saif Al-Kuwari, and Aiman Erbad. "Physical Layer Security in Satellite Communication: State-of-the-art and Open Problems". In: *arXiv preprint arXiv:2301.03672* (2023).

[272] Yue Wu et al. "Game-theoretic physical layer authentication for spoofing detection in internet of things". In: *Digital Communications and Networks* (2023). ISSN: 2352-8648. DOI: https://doi.org/10.1016/j.dcan.2022.12.016. URL: https://www.sciencedirect.com/science/article/pii/S235286482200284X.

[273] Simone Soderi et al. "Physical layer security based on spread-spectrum watermarking and jamming receiver". In: *Transactions on emerging telecommunications technologies* 28.7 (2017), e3142.

[274] Xingwang Li et al. "Physical Layer Security of Cognitive Ambient Backscatter Communications for Green Internet-of-Things". In: *IEEE Transactions on Green Communications and Networking* 5.3 (2021), pp. 1066–1076. DOI: 10.1109/TGCN.2021.3062060.

[275] Yongpeng Wu et al. "A Survey of Physical Layer Security Techniques for 5G Wireless Networks and Challenges Ahead". In: *IEEE Journal on Selected Areas in Communications* 36.4 (2018), pp. 679–695. DOI: 10.1109/JSAC.2018.2825560.

[276] Fatima Salahdine, Tao Han, and Ning Zhang. "Security in 5G and beyond recent advances and future challenges". In: *Security and Privacy* 6.1 (2023), e271.

[277] Ijaz Ahmad et al. "Security for 5G and Beyond". In: *IEEE Communications Surveys and Tutorials* 21.4 (2019), pp. 3682–3722. DOI: 10.1109/COMST.2019.2916180.

[278] Kanneboina Ashok and S Gopikrishnan. "Statistical Analysis of Remote Health Monitoring Based IoT Security Models & Deployments From a Pragmatic Perspective". In: *IEEE Access* 11 (2023), pp. 2621–2651.

[279] Ruiquan Lin et al. “Deep Reinforcement Learning for Physical Layer Security Enhancement in Energy Harvesting Based Cognitive Radio Networks”. In: *Sensors* 23.2 (2023), p. 807.