Detecting and Defending against Cyber Attacks in a Smart Home Internet of Things Ecosystem

A thesis submitted in partial fulfilment

of the requirement for the degree of Doctor of Philosophy

Eirini Sofia Anthi

February 2022

Cardiff University
School of Computer Science & Informatics

Στη νονα μου

Acknowledgements

I would like to sincerely thank my academic supervisors, Prof Pete Burnap, Dr George Theodorakopoulos, and Prof Omer Rana, for their support, guidance, and their continual encouragement and patience.

I owe a special thanks to Pete, who saw a potential in me that I didn't see in myself and supported me throughout both my personal and academic life. You have opened doors for me that have led to valuable experiences that any PhD student can only dream of. Thank you.

My time as a research student would not have been the same without the support from the School of Computer Science & Informatics, Cardiff University. My colleagues provided feedback, discussions, recommendations, company, and friendship, which made researching my PhD both interesting and enjoyable. In particular, I would like to thank Philipp, Adi, and Matthew.

Furthermore, I would like to thank my friends Amir, Wafi, Zoe, Dora, Matilda, and Shaz for their endless support, love, and encouragement. However, I would like to give my special thanks to Lowri. Thanks for not letting me give up. Dwi ddim yn meddwl y byswn i wedi gallu gorffen y traethawd hir heb dy help di. Diolch am fy ngharu i a choelio ynof i ar yr adegau y doeddwn i ddim yn coelio ynof i fy hun.

Finally, I would like to thank my mum for her unconditional support during my academic studies. I dedicate this thesis to my grandmother.

Abstract

The proliferation in Internet of Things (IoT) devices is demonstrated by their prominence in our daily lives. Although such devices simplify and automate everyday tasks, they also introduce tremendous security flaws. Current security measures are insufficient, making IoT one of the weakest links to breaking into a secure infrastructure which can have serious consequences. Subsequently, this thesis is motivated by the need to develop and further enhance novel mechanisms tailored towards strengthening the overall security infrastructures of IoT ecosystems.

To estimate the degree to which a hub can improve the overall security of the ecosystem, this thesis presents a design and prototype implementation of a novel secure IoT hub, consisting of various built-in security mechanisms that satisfy key security properties (e.g. authentication, confidentiality, access control) applicable to a range of devices. The effectiveness of the hub was evaluated within a smart home IoT network upon which popular cyber attacks were deployed.

To further enhance the security of the IoT environment, the initial experiments towards the development of a three-layered Intrusion Detection System (IDS) is proposed. The IDS aims to: 1) classify IoT devices, 2) identify malicious or benign network packets, and 3) identify the type of attack which has occurred. To support the classification experiments, real network data was collected from a smart home testbed, where a range of cyber attacks from four main attack types were targeted towards the devices.

Lastly, the robustness of the IDS was further evaluated against Adversarial Machine

Learning (AML) attacks. Such attacks may target models by generating adversarial samples which aim to exploit the weaknesses of the pre-trained model, consequently bypassing the detector. This thesis presents a first approach towards automatically generating adversarial malicious DoS IoT network packets. The analysis further explores how adversarial training can enhance the robustness of the IDS.

Contents

List of Publications

This thesis includes work introduced in the following publications:

- **Anthi, E.**, Ahmad, S., Rana, O., Theodorakopoulos, G. and Burnap, P., EclipseIoT: A secure and adaptive hub for the Internet of Things. Computers & Security, 78, pp.477-490. 2018.
- **Anthi, E.**, Williams, L., Słowińska, M., Theodorakopoulos, G. and Burnap, P., A Supervised Intrusion Detection System for Smart Home IoT Devices. IEEE Internet of Things Journal, 6(5), pp.9042-9053. 2019.
- **Anthi, E.**, Javed, A., Rana, O. and Theodorakopoulos, G., Secure data sharing and analysis in cloud-based energy management systems. In Cloud Infrastructures, Services, and IoT Systems for Smart Cities (pp. 228-242). Springer, Cham. 2017. (Best paper award).
- **Anthi, E.**, Williams, L. and Burnap, P., Pulse: an adaptive intrusion detection for the Internet of Things. Living in the Internet of Things: Cybersecurity of the IoT-2018 (pp. 1-4). IET. 2018.
- **Anthi, E.**, Williams, L. Javed, A. and Burnap, P., Hardening Machine Learning Denial of Service (DoS) Defences Against Adversarial Attacks in IoT Smart Home Networks. Computers & Security, page 102352, 2021.

List of Figures

List of Tables

List of Acronyms

6Mapper 6LoWPAN Mapper

ARP Address Resolution Protocol

AES Advanced Encryption Standard

AML Adversarial Machine learning

ABS Alert Based Systems

API Application Programming Interface

ANN Artificial Neural Networks

BSSID Basic Service Set Identifier

BIA Bayesian Inference Approach

BLE Bluetooth Low Energy

CNI Critical National Infrastructure

CSIR Cyber Security Incident Response Guide

DNS Data stream Network

DT Decision Tree

DoS Denial of Service

DDoS Distributed Denial of Service

ES-FCM ELM Fuzzy C-Means

ETX Expected Transmission

XML Extensible Markup Language

FN False Negatives

FP False Positives

FGSM Fast Gradient Sign Method

GAN Generative Adversarial Networks

GEA Graph Embedding and Augmentation

HTTP Hypertext Transfer Protocol

IFTTT If This Then That

ICS Industrial Control Systems

ICT Information Communication Technology

IaaS Infrastructure as a service

ICMP Internet Control Message Protocol

IoT Internet of Things

IP Internet Protocol

ISP Internet Service Provider

IP ACL Internet Protocol Access Listings

IT Information Technology

IDS Intrusion Detection System

JSMA Jacobian based Saliency Map

LED Light Emitting Diode

LAN Local Area Network

LSTM Long Short Term Memory

MITM Man-In-The-Middle

MAC Media Access Control

NCSC National Cyber Security Centre

PCAP Packet Capture

PDML Packet Description Markup Language

Pub-Sub Publish-Subscribe

RNN Random Neural Networks

RPL Routing over Low Power and Lossy Networks

SSH Secure Shell

SDK Software Development Kit

SVM Support Vector Machine

TTL Time to Live

TCP Transmission Control Protocol

TLS Transport Layer Security

TN True Negatives

TP True Positives

UUID Unique User Identification

UDP User Datagram Protocol

WPA Wireless Protected Access

WSN Wireless Sensor Networks

Chapter 1

Introduction

The Internet of Things (IoT) is defined as the system of interconnected electronic devices embedded with software, sensors, actuators, and network connectivity which enable them to connect and exchange data [69]. Smart devices such as wearable devices, home appliances, alarms and camera systems routinely collect personal information and provide various functionalities which automate and support our daily activities and needs. As a result, the popularity of such devices has significantly increased over the past few years. This is due to their affordability, as well as their ubiquitous connectivity which allows them to communicate and exchange information with other technologies, their intelligence, and their decision-making capabilities to invoke actions [121]. This provides seamless user experiences which significantly enhances people's every day lives and is demonstrated by how prominent such devices are today.

The proliferation of smart devices is not only within the domestic environment, but it is also the driving force behind the development of an interconnected knowledge-based world; our economies, societies, machinery of government, and Critical National Infrastructure (CNI) [182]. However, although these concepts support everyday life tasks, their dependency on Information Communication Technology (ICT) and IoT devices introduce severe security risks [64]. Whereas security protocols and best practices for traditional Information Technology (IT) is well-understood and broadly adopted, security for IoT devices is nascent and is rarely sufficient.

Cyber attacks against IoT can lead to disastrous effects including personal information leakage, damage to hardware, disrupting the system's availability, causing system blackouts, and even physically harm individuals [63]. Thus, the scale of the impact of the attacks performed on IoT networks can vary significantly depending on the targeted device. Subsequently, given that IoT devices have a direct impact on our lives, security and privacy considerations must become a high priority [95].

Two of the main reasons that make IoT devices vulnerable to cyber attacks include their limitations in computational power and their heterogeneity. More specifically, it is generally not feasible for IoT devices with restricted computational power, memory, radio bandwidth, and battery resource to execute computationally intensive and latency-sensitive security tasks that generate heavy computation and communication load [208]. As a result, it is not possible to employ complex and robust security measures. In addition, the heterogeneity which surrounds IoT devices in terms of hardware, software, and protocols [62] poses a great challenge towards developing and deploying security mechanisms that can endure with the scale and range of devices [62]. Consequently, it is evident that there is a major gap between security requirements and security capabilities of the IoT devices that are currently available.

1.1 Motivation and Problem Definition

Although IoT is considered as being the next 'Technological Revolution' [97] which is shifting how we as individuals, economic entities, and governmental organisations interact with the physical world, such technologies come with enormous security flaws [209, 106, 218, 117].

Several well established enterprises and organisations [21, 30] have demonstrated that IoT devices are subject to a range of security flaws, including heartbleed, Man-In-The-Middle (MITM), Denial of Service (DoS), data leakage, weak passwords, and more [63, 30]. In addition, IoT devices have recently been employed as part of botnets, such

as *Mirai*, and have launched several of the largest Distributed Denial of Service (DDoS) and spam attacks [123].

As a result, the research in this thesis is motivated by the main matter of principle: given the insecurity of such devices, and given that they are often deeply embedded in networks, IoT may be considered as being the 'weakest link' for breaking into a secure infrastructure. Thus, such devices are attractive targets for cyber attacks. Consequently, there is a significant need for the development of novel security mechanisms that work across many different paradigms and improve not only the defence of IoT against a range of cyber attacks, but also the detection of such attacks, and subsequently their mitigation from IoT networks.

In an attempt to enhance the security of IoT devices, several studies have focused on achieving specific individual security objectives such as authentication, confidentiality, integrity, and access control [219, 125, 132, 161, 213, 138]. However, due to their heterogeneity, applying these security mechanisms in a uniform way to a range of IoT devices may be challenging. IoT hubs are currently among the most popular IoT management models [215]. Several studies have proposed such hubs to tackle the heterogeneity in IoT environments [176, 147, 101, 167, 58, 107]. However, these approaches mainly focus on handling the big data that is generated by smart devices, attempt to overcome constraints of their computational power, their scalability, and not to enhance the overall security of the ecosystem. These approaches will be discussed in more detail in Chapter 3. Therefore, this leads to the first research question:

RQ1 *What security mechanisms can be incorporated within a smart home IoT hub that can be uniformly applied to a range of heterogeneous devices?*

Nevertheless, as secure as an infrastructure may be, it is still in the nature of an adversary to attempt to compromise its security. In this case, it is also important to consider the implementations of the monitoring and detection tools which aim to capture such intrusions. Traditional IT security mechanisms consist of a range of tools, such as

firewalls and Intrusion Detection System (IDS), which assist in monitoring networks. However, due to their heterogeneity and scalability, such mechanisms cannot handle IoT deployments and device and/or vendor constraints [215, 203]. Furthermore, as IoT devices operate deep inside the network, traditional perimeter defences are inadequate as they can help block external attacks, but often fail to prevent them from internal devices or applications [99]. Finally, as the number of IoT devices increases exponentially [106], the number of unknown vulnerabilities and threats also increases, making traditional signature-based IDS systems ineffective.

In an attempt to combat these constraints, current research [60, 141, 145, 168, 80] has focused on using machine learning approaches to develop more adaptable IDSs specifically for IoT ecosystems. However, as discussed further in Chapter 4, such approaches come with a range of limitations. More specifically, such systems are limited to detecting one type of attack at a time, have been evaluated using simulated network data, and only focus on detecting whether network activity is malicious or benign. This may mean that current IDS implementations are limited to unrealistic network behaviours and are inefficient against a range of attacks. To address these limitations, network traffic from a representative IoT smart home network may be collected. This leads to the second research question:

RQ2 *Can supervised machine learning approaches support the automatic detection of a range of cyber attacks based on network packet features collected from a range of IoT devices?*

Additionally, such systems lack focus on device type classification and profiling. Previous work on device classification mostly focuses on distinguishing whether a device is IoT or non-IoT, or identifying the specific vendor of the devices. This is achieved by using statistical traffic features which may be collected and extracted using additional tools and software. Device profiling is an important feature within an IDS for two main reasons. Firstly, it allows the assets within the network to be identified. Given the first

reason, it may be possible to detect anomalies outside of the device's 'normal' behaviour, subsequently allowing countermeasures to be launched towards those devices. In this case, and to align with RQ2, this leads to the third research question:

RQ3 *Can supervised machine learning algorithms successfully classify different IoT devices based on network packet features?*

However, such approaches do not attempt to identify the exact type of attack that has occurred. This is a critical piece of information which may significantly increase the response rate, and therefore the mitigation of the attack by launching appropriate countermeasures. Without this information, identifying the exact type of attack requires significant human effort, specifically in networks with a large number of devices [7]. This can lead to a delayed launch of countermeasures and can have significant consequences. This motivates the fourth research questions:

RQ4 *Given RQ2, can supervised machine learning algorithms further identify the main type of attack which has occurred?*

Subsequently, an IoT tailored IDS implementation which focuses on these three aspects and addresses the aforementioned limitations has the potential to significantly enhance the security of the ecosystem.

It is also important to consider the robustness of such detection systems. Given the popularity of such solutions, adversaries have turned towards Adversarial Machine learning (AML). Such techniques allow the vulnerabilities surrounding the machine learning approach which forms the basis of the IDS to be exploited, subsequently allowing adversaries to bypass the detectors. To the best of our knowledge, thus far, there is no comparable research which investigates the effects of AML in the context of IoT networks. However, there exists several studies which explore the application of AML towards email spam classifiers, malware detectors, and Industrial Control Systems (ICS) [148, 221, 132, 222]. This motivates the fifth research question:

RQ5 *Can AML techniques be used to evaluate the robustness of a supervised IDS for the IoT?*

Finally, it is important to examine methods that may aid in enhancing the robustness of the supervised detector against AML attacks. A popular approach for defending against AML attacks is adversarial training. This involves including a subset of adversarial samples in the original training set and re-training the model. This leads to the sixth research question:

RQ6 *Can adversarial training enhance the robustness of a supervised IDS for the IoT?*

1.2 Research Aims and Objectives

Motivated by the aforementioned research questions discussed in Section 1.1, the overarching goal of this thesis is to explore how security mechanisms tailored for the IoT ecosystem can enhance its security. As previously discussed, heterogeneous IoT devices within a home introduce a new threat vector that may have severe consequences. As such, given the different applications of IoT, the variety of different vendors, and their low computational power, there is a need to create a framework that can accommodate and secure a range of diverse devices. Following a secure framework to deploy IoT within a home, it is equally important to implement mechanisms to monitor the network traffic of the local IoT network to detect attacks effectively. Finally, it is crucial to examine the robustness of such tools by evaluating against attacks that may be obscured, increasing trust and adoption while retaining security. Overcoming heterogeneity and being able to uniformly apply a range of security mechanisms, addressing the gap in effective security monitoring solutions, and gaining an understanding around how the robustness of such security solutions, all contribute to the research outcome; creating a secure IoT environment.

More specifically, within this work, two core novel security mechanisms are proposed; one to defend and one to detect attacks in IoT environments. Both mechanisms are essential to create a more robust and complete security model.

In order to address the overarching goal of this thesis, the following key objectives were identified:

1. Analyse the literature surrounding the state-of-the-art approaches towards defending and detecting attacks in IoT environments to identify the limitations of current approaches.

2. Design a prototype of a secure hub framework to defend against cyber attacks that may threaten a smart home network.

3. Design and implement a typical IoT smart home network to deploy and evaluate the proposed hub using a traditional penetration testing methodology that corresponds to the attacker's objectives as defined in the contribution Chapter.

4. Identify the key requirements of a supervised IDS tailored for IoT environments.

5. Evaluate the feasibility and the performance of the proposed IDS using real network data derived from a typical IoT testbed consisting of a range of devices. This performance will be measured using standard machine learning classification metrics; Precision, Recall, F1-score. The goal is to maximise these measures as higher values correspond to better classification performance.

6. Further evaluate the robustness of the IDS against AML attacks. This will be achieved by developing a methodology to generate adversarial DoS packets and observing the classification performance metrics when such packets are presented to a trained supervised machine learning model.

7. Explore the effectiveness of adversarial training in increasing the robustness of machine learning detectors against AML attacks. This will be achieved by evaluating the performance of adversarial training.

To achieve the research objectives presented herein, the scope of the experiments are designed within the context of smart home IoT environments. In this case, a review of the literature surrounding the definition of a smart home, its properties, the type of devices that are included within such environments, and the types of interactions with such devices was conducted. These are discussed in more detail within the testbed sections, 3.5.1 and 4.2, in each contribution Chapter.

1.3 Research Methodology

Given the technical nature of this work, and its focus on detecting attacks and defending the IoT environment, an empirical methodology based on a quantitative research framework is adopted. As noted by Rasinger [164], quantitative research is deductive; that is, research questions are developed from prior theories, and are then proven (or disproven) during the empirical investigation. Having reviewed relevant and comparable literature (Chapter 2), the research methodology herein combines the core principles of several acknowledged quantitative approaches (e.g.[108, 201]). This includes reviewing relevant literature for appropriate research methods, designing the research methods and strategies to achieve the aforementioned research objectives, designing and implementing testbeds to facilitate experimental work (including identifying the relevant tools required to configure and communicate with IoT devices), collecting relevant data (including identifying the variables to measure), analysing such data, and the interpretation and presentation of results. In addition, rather than conducting a qualitative user-focused study, to inform our design choices for the testbed (see Sections 3.5.1, 4.2, 4.3.1, 3.2), and collect data by interacting with such devices, previous studies concerning smart home network configurations, the devices in such networks, and the number of and interactions with such devices are to be reviewed and adopted where relevant. Figure 1.1 illustrates a graphical representation of the quantitative research methodology followed in this thesis as identified also by [108].

The following steps, and as illustrated in Figure 1.2, describe the exact steps followed in the aforementioned quantitative methodology used to structure the work presented in this thesis which focuses on fulfilling the research objectives identified in Section 1.2.

1. The first part of the research raises the need to comprehend the applicability of a hub architecture towards uniformly securing a range of heterogeneous devices on a smart home network. This is achieved by:

    - Step 1 and Step 2: Reviewing related literature to identify the configurations and design of existing secure IoT hubs, their limitations, the configurations and design of traditional smart home networks, which smart home devices and how many are often connected to such networks, and which cyber attacks are the most impactful threat towards them.
    - Step 3: Designing, configuring, and implementing a smart home testbed.
    - Step 4 and Step 5: Designing, configuring, and implementing a secure hub.
    - Step 6: The validation of the proposed hub is achieved by using an unbiased, industry standard penetration testing methodology.

    The results of the penetration testing evaluation provides the essential justifications which demonstrate the success of the hub's design in increasing the security of heterogeneous devices on a traditional IoT smart home network.

2. The second part of the research raises the need to comprehend the applicability of a supervised IDS tailored towards detecting maliciousness in IoT environments. This is achieved by:

    - Step 1, Step 2, and Step 3: Reviewing related literature to identify the configurations and design of existing IDSs and their limitations.
    - Step 4: Designing, configuring, and implementing a smart home testbed.

- Step 5: Collecting network packet data when the devices on the testbed are in their natural state (including when they were interacted with them) and are under attack. To determine the interactions with the devices, user-focused literature is reviewed to identify the common interactions in a smart home network.
- Step 6: Evaluating the performance of a selection of state-of-the-art supervised classifiers. The validation of such classifiers is achieved using a cross-validation approach, a re-sampling method that uses different portions of the data to test and train a model on different iterations, where the goal is to estimate how accurate a predictive model will perform in practice. In addition, the validation of the best performing classifier is achieved by analysing how the outcomes are re-distributed following classification. This investigation provides insight into whether any frequent misclassifications occur between similar devices and attack types, or conversely, the model can distinctly discriminate between device and attack type behaviours with high accuracy.

The results of this analysis provide the essential justifications which demonstrate the success of a supervised machine learning approach in identifying malicious network packets with high accuracy. Subsequently, this result raises the need to evaluate the robustness of the proposed IDS against AML attacks. This is achieved by:

- Step 6a: Analysing the performance of the IDS when presented with adversarial samples. The validation of such analysis is achieved by comparing the original classification performance of the IDS against its performance when such adversarial samples are present.
- Step 6b: In a similar fashion, analysing the model's performance following adversarial training. The validation of such analysis is achieved by comparing such results against prior performances.

The results of this analysis provides the essential information that the proposed supervised IDS is vulnerable towards AML attacks, as well as the success of adversarial training in enhancing its robustness.

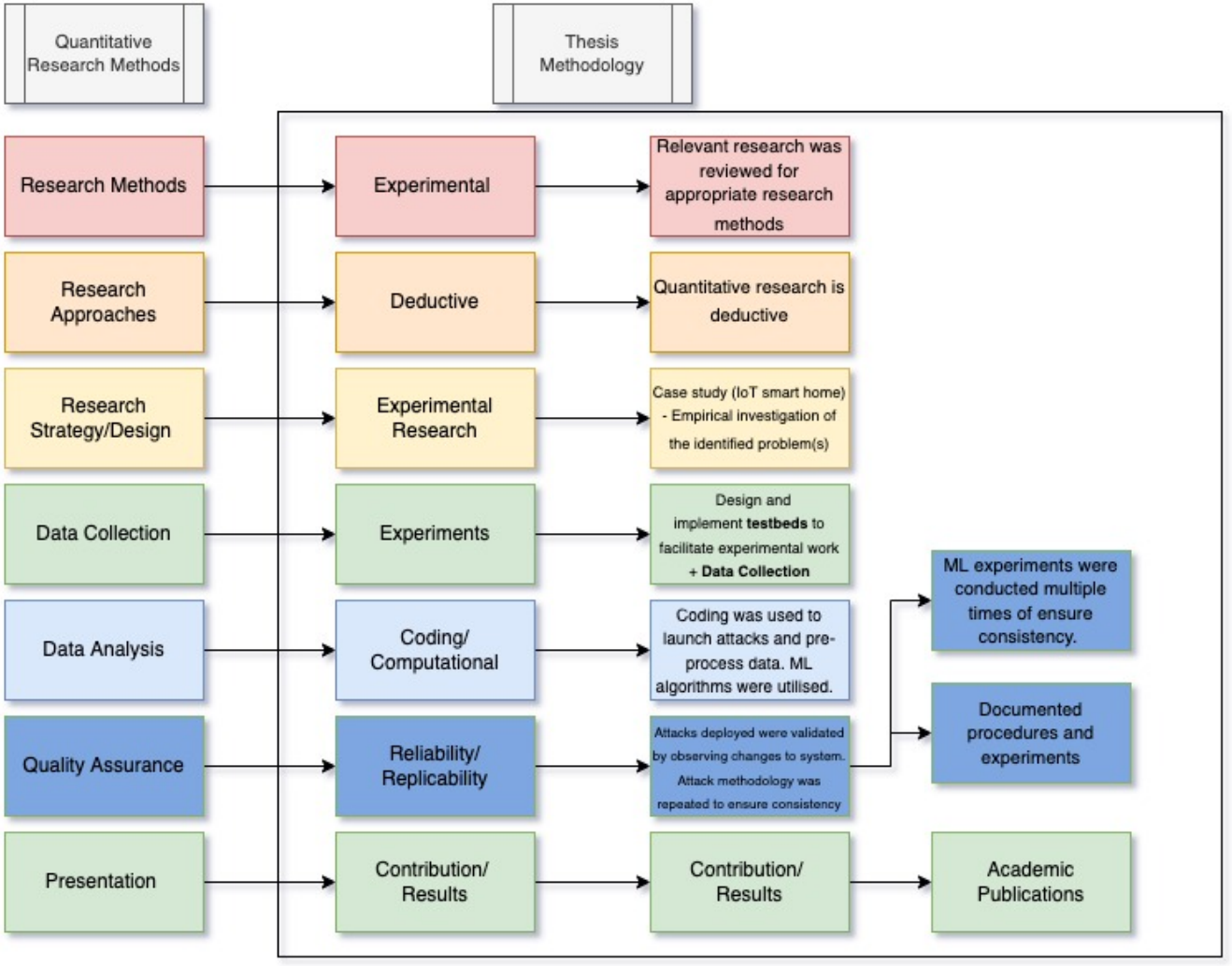

Figure 1.1: A graphical representation of the quantitative research methodology followed in this thesis

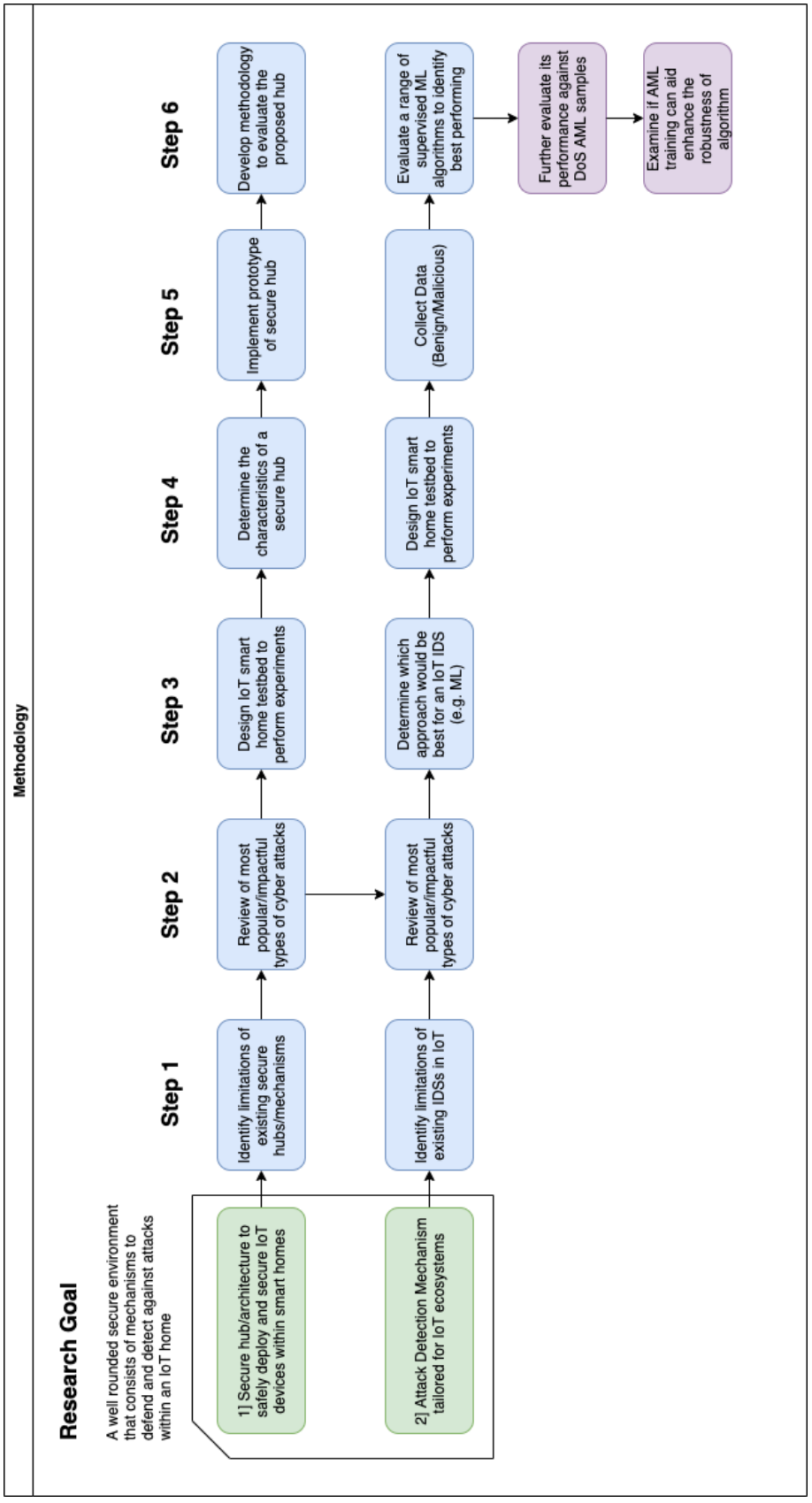

Figure 1.2: A map detailing the exact steps taken within the quantitative methodology followed in this thesis

1.4 Research Contributions

The four main research contributions presented herein are as follows:

C1 As a response to RQ1, this contribution proposes the first design and prototype implementation of a secure and heterogeneity-aware hub for the IoT. The proposed hub can defend against two attacker models which take into consideration some of the most popular attacks that may threaten IoT smart home networks. To defend against these types of adversaries, the hub contains built-in security mechanisms that satisfy the following security properties: secure user authentication, secure access control, confidentiality, device cloaking, and user/attacker behaviour monitoring. More specifically, the hub uses dynamically loadable add-on modules to communicate with various diverse IoT devices, provides policy-based access control and secure authentication, limits the exposure of the local IoT devices through *cloaking*, and offers a *canary-function* based capability to monitor attack behaviours.

C2 To address RQ2, RQ3, and RQ4, this contribution includes an investigation into how supervised machine learning algorithms can be utilised to support a novel three layer IDS tailored towards the IoT. The detection system aims to: 1) classify the IoT devices connected on the network, 2) identify whether network packets are malicious or benign, and 3) given malicious packets in 2), identify the type of attack which has occurred. To the best of our knowledge, thus far, no research has investigated into the development of such three-dimensional IDS in this context.

C3 To address RQ5, this contribution includes an exploration of how AML techniques can be applied to evaluate the robustness of a machine learning based IDS tailored towards the IoT. The proposed approach aims to showcase how manipulating malicious network packet features can force a machine learning model to misclassify malicious packets as benign, consequently bypassing the detector.

To the best of our knowledge, thus far, there is no comparable research which investigates the effects of AML in the context of IoT networks.

C4 To address RQ6, this contribution includes an investigation of how adversarial training may be used to enhance the robustness of a supervised IDS for the IoT.

The contributions presented in this thesis have formed and extended a number of peer-reviewed research papers:

C1 Anthi, E., Ahmad, S., Rana, O., Theodorakopoulos, G. and Burnap, P., 2018. EclipseIoT: A secure and adaptive hub for the Internet of Things. Computers& Security, 78, pp.477-490. [62]

C2 Anthi, E., Williams, L., Słowińska, M., Theodorakopoulos, G. and Burnap, P., 2019. A Supervised Intrusion Detection System for Smart Home IoT Devices. IEEE Internet of Things Journal, 6(5), pp.9042-9053. [66]

C2 Anthi, E., Williams, L. and Burnap, P., 2018. Pulse: an adaptive intrusion detection for the Internet of Things. Living in the Internet of Things: Cybersecurity of the IoT-2018. (pp. 1-4). IET. [64]

C3 & C4 Anthi, E., Williams, L., Javed, A. and Burnap, P., 2021. Hardening Machine Learning Denial of Service (DoS) Defences Against Adversarial Attacks in IoT Smart Home Networks. Computers & Security [65]

The following peer-reviewed publication contributes to the literature review provided in Chapter 2:

Anthi, E., Javed, A., Rana, O. and Theodorakopoulos, G., 2017. Secure data sharing and analysis in cloud-based energy management systems. In Cloud Infrastructures, Services, and IoT Systems for Smart Cities (pp. 228-242). Springer. [63]

1.5 Thesis Structure

The outline for the remainder of this thesis is as follows:

- **Chapter 2** - *Background* - Introduces IoT and the key challenges surrounding cybersecurity in such systems, as well as their comprising topics and the key concepts related to this research.
- **Chapter 3** - *A Secure and Heterogeneity-Aware Hub for IoT Smart Homes* - Introduces the architecture of a novel and secure hub for the IoT environment. This Chapter presents contribution C1.
- **Chapter 4** - *A Supervised Intrusion Detection System for the IoT Environment* - Explores how supervised machine learning algorithms can be utilised to support a novel three layer IDS tailored for the IoT. This Chapter presents contribution C2.
- **Chapter 5** - *Adversarial Attacks on Machine Learning Cybersecurity Defences in IoT* - Investigates how AML can be used to evaluate the robustness of supervised attack detectors, as well as applying AML techniques to enhance their capabilities in detecting cyber attacks. This Chapter presents contribution C3 and C4.
- **Chapter 6** - *Results and Contributions* - Summarises the research contributions and findings presented in this thesis.
- **Chapter 7** - *Conclusions and Future Work* - Concludes the thesis, discusses the real-world implementation of the research outcomes, highlights the limitations surrounding such mechanisms, as well as highlighting proposals for future work.

Chapter 2

Background

2.1 Introduction

Due to the rapid proliferation of smart devices and their prominence in our daily lives, the past decade has seen substantial research revolving around the security of IoT infrastructures. Although such devices may simplify and automate our everyday tasks, they also introduce significant security flaws. Current insufficient security measures employed to defend smart devices make IoT one of the weakest links to breaking into a secure infrastructure, and therefore an attractive target to attackers [64]. As a result, the field of IoT security has become a critical area of research as it coincides with the rapid increase in the motivations, opportunities, and impact of cyber attacks towards IoT devices.

The purpose of this Chapter is twofold. Firstly, the key challenges surrounding cybersecurity in IoT are discussed. This includes the heterogeneity of smart devices and the application of hubs as a unified security mechanism in the ecosystem. This also includes the application of machine learning as an approach towards implementing adaptive and intelligent security mechanisms for detecting malicious behaviour in IoT networks. Secondly, an insight of the wider research revolving around machine learning and cybersecurity is provided, which is used as a basis for reflecting on the research contributions of the work presented in this thesis. The following methodology [26] was followed to identify current, relevant, and comparable state-of-the-art research used to

motivate the work presented in this thesis:

1. Identify a research topic; in this case, literature revolving around or with the following keywords: *secure IoT hubs/architectures, security mechanisms, intrusion detection systems for IoT (signature/anomaly/machine learning based)*, and *bypassing intrusion detection systems*.

2. Search for papers in IEEE Xplore, ACM Digital Library, Google, Google Scholar using the aforementioned keywords.

3. Review secondary sources to gain an overview of the topic, such as *Intrusion detection systems in Internet of Things: A literature review*, *A review of intrusion detection systems using machine and deep learning in internet of things: Challenges, solutions and future directions*, *A review of security concerns in Internet of Things*.

4. Determine relevant preliminary sources and primary research journals (e.g. Computers & Security, IEEE Internet of Things Journal) and extract the references of other relevant research publications.

5. To determine the relevance of the sources the following points were evaluated:

    - the topic - abstracts and conclusions were read to consider the relevance of the work to the research topic investigated herein and whether it covered the topic adequately.
    - the publication date - to ensure that it is sufficiently current for the research topic investigated herein. More specifically, for this work, where possible, literature from 2014 onwards were reviewed.
    - quality of publication - we further investigated the credentials of the authors of the source and the quality of venue to determine their level of expertise and knowledge about the subject.

6. Based on the above, select specific articles to review in more depth.

7. Read articles and prepare bibliographic information and notes on each article with their limitations and their importance.

8. Evaluate the research reports and formulate research questions to expand/or address the limitations of these papers. Subsequently, design an appropriate experimental methodology to address such research questions.

This Chapter is divided into the following main Sections: Section 2.2 defines the aims and objectives of IoT devices, as well as providing context for their applications. Section 2.3 introduces the basic components of the IoT ecosystem and discusses the issues surrounding the heterogeneity of smart devices with respect to cybersecurity. Section 2.4 discusses the cybersecurity vulnerabilities surrounding IoT. Section 2.5 and Section 2.6 discuss both the mitigation and detection of cyber attacks in IoT respectively, including the integration of machine learning in IDS for IoT networks. Section 2.7 discusses the vulnerabilities associated with machine learning algorithms in the context of attack detection. Finally, Section 2.8 summarises the main topics discussed in this Chapter.

2.2 Internet of Things (IoT)

The term IoT was originally coined by Kevin Ashton, who discussed the possibility of linking the then new idea of RFID technology in supply chains with the Internet [69]. Since then, and with the evolution of technology, the definition of IoT has evolved to describing a wider range of applications within different domains, such as transport and health care [194, 106]. Today, IoT is used to describe a things-connected ecosystem of electronic devices which are embedded with software, sensors, actuators, and network connectivity, which enable them to connect and exchange data from their environment without human intervention [162, 69].

IoT devices, such as smart and wearable devices, home appliances, and alarm and camera systems provide various functionalities which automate and support our daily activities and needs. Due to their ubiquitous connectivity which allows them to communicate and exchange information with other technologies, their intelligence, and their decision making capabilities to invoke actions [121], the number of smart devices has increased dramatically. Statista Research Department [42] predicts that by 2025, there will be over 75 billion IoT devices connected around the world, overtaking the number of personal computers and smartphones combined [209]. Figure 2.1 illustrates the increase in the number of connected IoT devices per year between 2015 and 2025.

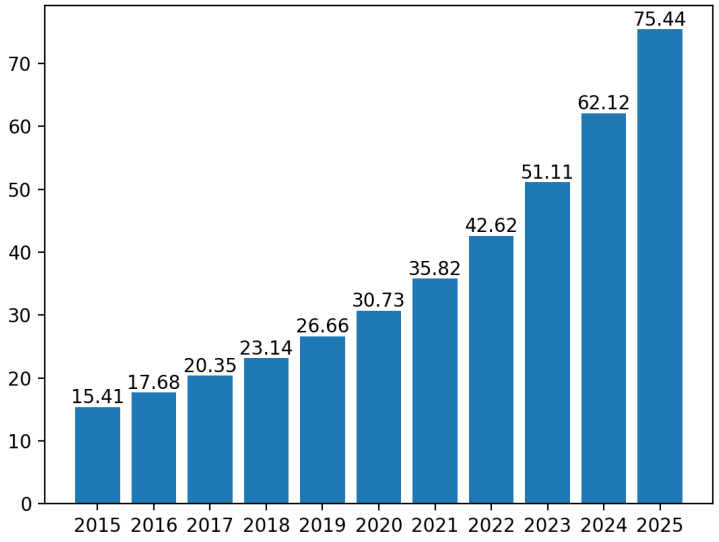

Figure 2.1: Number of connected IoT devices worldwide from 2015 - 2025 as predicted by Statista Research Department [42] (in billions).

2.2.1 Applications of IoT

IoT devices may be embedded in several different domains. Such devices are widely used particularly in the domestic environment. One of the most popular applications of IoT is within smart homes [71]. Smart home appliances include smart thermostats,

lighting, meters, and locks which may be controlled remotely from a smart phone or laptop. Users are able to monitor the usage of such devices, providing more information and allowing the user to make a more informed decision as to when to change such usage. For example, a user may want to schedule their heating system to switch off once it has reached a certain temperature to save energy and money. Other devices may include smart fitness trackers and wearables such as Fitbit which allow users to track their physical movements in order to measure and set personal fitness goals.

However, the use of IoT devices is not limited to smart homes and wearable devices. IoT infrastructures have become the driving force being the development of an interconnected knowledge-based world; our economies, societies, machinery of government, and CNI [182]. These include concepts that may be necessary for a country to function and upon which our daily life depends on, such as smart cities, intelligent transport systems, smart grids, and health care systems [71]. For example, smart meters and energy regulators are embedded into smart grids to measure energy trends in order to efficiently distribute energy across demand [63].

It is therefore evident that IoT is transforming the way in which such domains operate by providing real-time, more accurate, and automated measurements and functionalities [129].

2.3 The IoT Ecosystem

The concept behind IoT infrastructures is as powerful as it is complex. IoT refers to a broad and diverse ecosystem that includes a wide range of devices, applications, protocols, and use cases (see Figure 2.3). As a result, in order to better understand how an IoT ecosystem operates, such infrastructures are separated into seven key layers [88]. As seen in Figure 2.2, each layer consists of a variety of different devices, technologies, and operations.

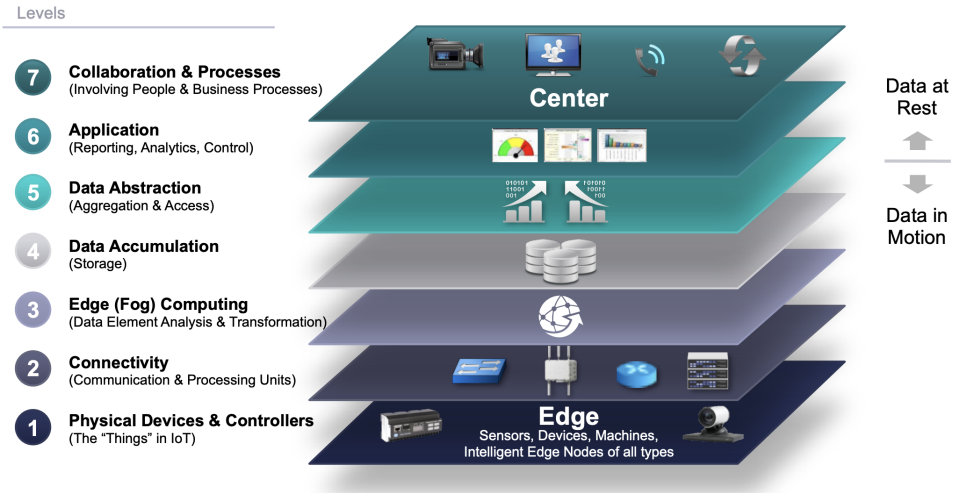

Figure 2.2: Internet of Things reference model [24]

In more detail:

1. **Physical devices and controllers (Perception Layer)** - This is the lowest layer in the IoT ecosystem architecture. It consists of a range of different sensors and devices that are responsible for gathering information from the environment.

2. **Connectivity (Communication Layer)** - This is the core layer of in the IoT system and is responsible for connecting the sensors, network devices, and servers. Various types of technologies are used in this layer, such as Wi-Fi, WAN, Local Area Network (LAN), 4G, 5G, LoRA, etc. [76].

3. **Edge computing (Cloud Edge or Gateway)** - This layer interfaces the data and control planes to the higher layers of the cloud or software layers. Protocol conversion, routing to higher layer software functions, and "fast path" logic for low latency decision making are implemented in this layer [212].

4. **Data accumulation (Middleware Layer)** - This layer is responsible for storing, analysing, and processing the data received from the sensors and devices. [206]

5. **Data abstraction** - In this layer, the data collected from the sensors is organised and pre-processed in order to be able to used for further processing.

6. **Application** - This layer sits at the top of the architecture and is responsible for delivering application-specific services to the user. Moreover, it defines the various domains in which IoT can be deployed, as discussed in Section 2.2.1 [177, 76].

7. **Collaboration and processing** - In this layer, the processed data in the lower layers is integrated into business applications. The main challenge here is to effectively leverage the value of IoT and its services and to translate it into economic growth, efficient decision making, and business optimisation.

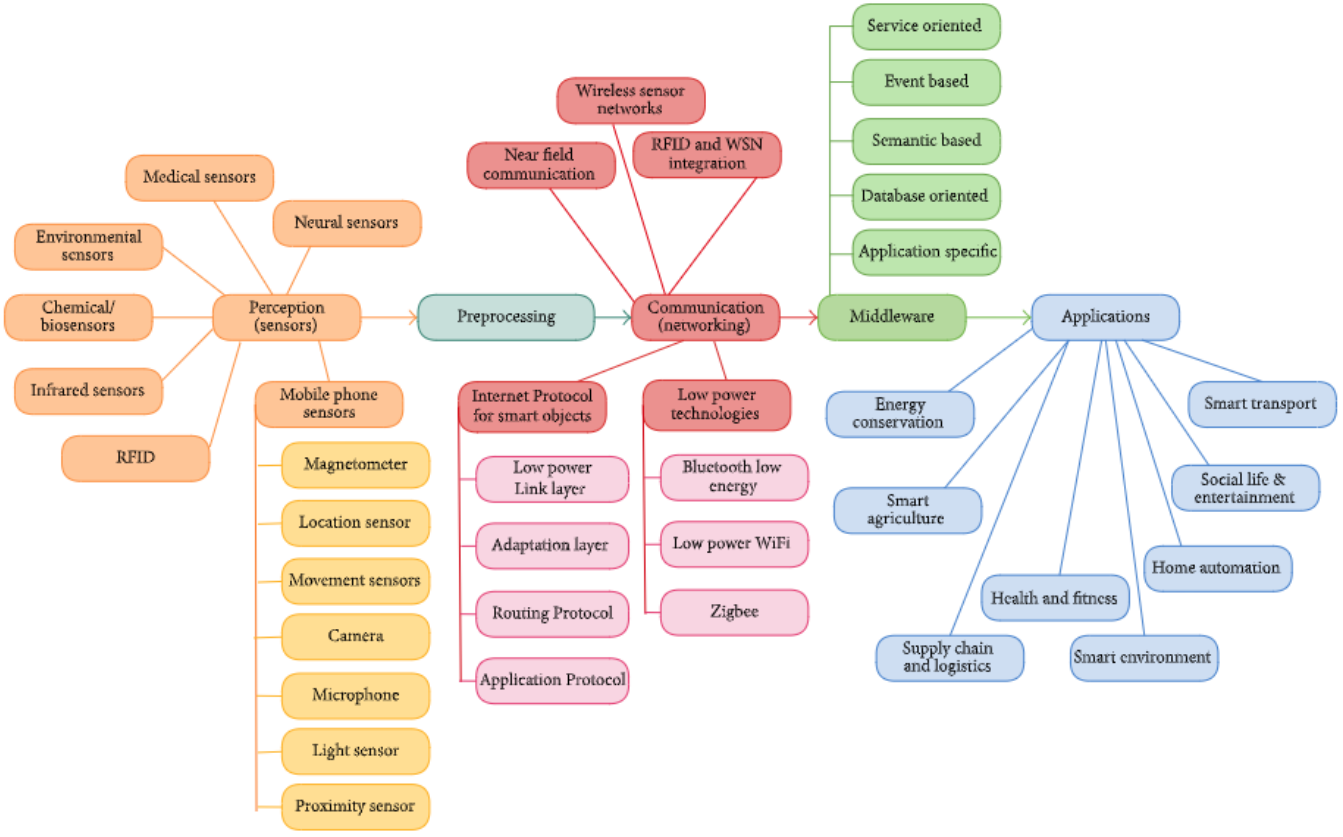

Figure 2.3: Taxonomy of IoT technologies in each architectural layer [177]

2.4 Cybersecurity in IoT

Following the discussion in Section 2.2.1, it is evident that IoT is one of the fastest growing technologies which has huge impact in various domains [48]. However, although IoT comes with several advantages, one of the main issues surrounding such devices are their tremendous security flaws.

A prominent study in the field is by Hewlett Packard Enterprise [15], who, in 2016, demonstrated that 70% of the most popular IoT devices on the market are vulnerable to a range of security threats. In particular, they discovered that each device had a recklessly high number of security vulnerabilities, each suffering from, on average, 25 issues. Other, later surveys, found similar limitations, such as the OWASP top 10 IoT vulnerabilities [30] and a study by Synopsys [43] and Abouzakhar et al. [49] who revealed a lack of security in medical devices. More recently, IoT devices have been employed as part of botnets and have launched several of the largest DDoS attacks [22]. A recent study by the IBM X-Force showed a 500% increase in overall IoT attacks which was largely driven by a new botnet variant; Mozi [159].

Since then, IoT devices have been developed to include some security improvements, such as transport encryption. However, statistics show that as the number of IoT devices increase, the number of cybersecurity attacks against them also increase and the majority of these devices are still insecure. More specifically, in 2019 Avast software [17] reported that a German-manufactured childrenâs smart watch was recalled by the EU when its controller app was found to have significant vulnerabilities. The device could potentially have been compromised to allow hackers to track the childâs movements in real time, or spoof the GPS location data to deceive parents. Furthermore, another study by Kumar et al. [126] discovered that many IoT devices had ports and protocols left open, even if they are not exclusively used by the device. In addition, they reported that many devices were found to have the File Transfer Protocol (FTP) and Telnet services open. In more detail, 17.4% of devices had weak FTP credentials while 2.1% had weak Telnet credentials. An even more recent study conducted in 2020

by Mangino et al. [140], revealed that a large number of IoT devices found online were susceptible to the aforementioned vulnerabilities. AAs a result, as threats towards IoT progressively increase in their complexity and severity, it is evident that the state of their security remains weak; thus, requiring further attention.

There are three main reasons as to why IoT devices are vulnerable to cyber attacks:

1. The heterogeneity of devices significantly expands the attack surface of IoT ecosystems. As such, implementing security mechanisms that can be universally applied to IoT consists a complex challenge.

2. The restrictions in the computational power of IoT devices severely affect their security. It is generally not feasible for IoT devices with restricted computational power, memory, radio bandwidth, and battery resource to execute computationally intensive and latency-sensitive security tasks that generate heavy computation and communication load [63, 218]. As a result, it is not possible to employ light-weight, complex, and robust security measures.

3. It is difficult to update the software or apply patches on such devices [183].

Given that they suffer from security vulnerabilities and that they are often deeply embedded in networks, IoT devices are considered as being one of the weakest links for breaking into a secure infrastructure.

2.4.1 Cyber Attacks in IoT Ecosystems

Multiple studies (e.g. [158, 179, 120, 146, 61]) have demonstrated that IoT devices are vulnerable to a wide range of cyber attacks. Figure 2.4 captures a range of different types of attacks that can occur in IoT networks. These include physical, network-based, firmware, and encryption attacks.

Physical and Close Proximity Attacks

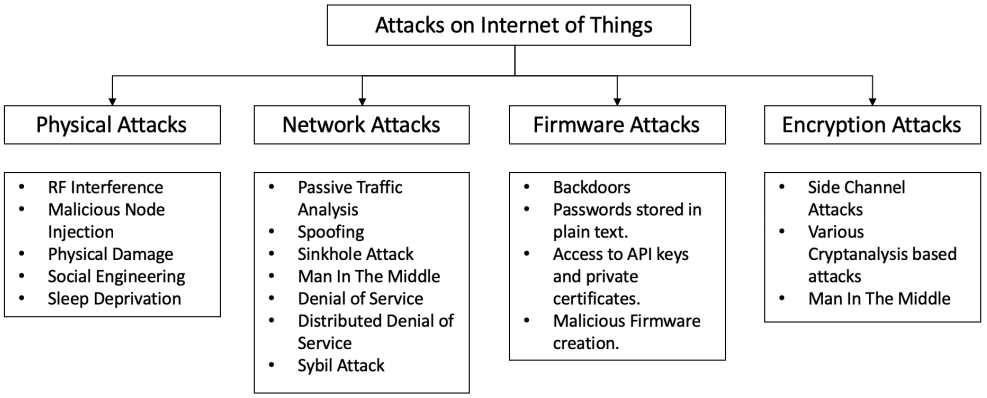

Figure 2.4: Types of cyber attacks in which IoT devices are vulnerable against [61].

Physical attacks focus on damaging the hardware components of a device by targeting their ports, memory, power source, etc. Such attacks can disrupt the device's functionality and can alter its behaviour. In order for an adversary to deploy these attacks, they are required to have physical access to the devices. The following are examples of physical attacks in which IoT devices are vulnerable against:

- Radio Frequency Interference/Jamming: This mostly refers to DoS attacks that can be caused by creating and transmitting noise signals over radio frequency.
- Physical Penetration: The adversary can potentially access an exposed device interface (e.g. JTAG), and thus can readily access memory, sensitive key material, passwords, configuration data, and a variety of other sensitive parameters [172].
- Physical Damage: The attacker physically damages the device in order to impact the availability of service.
- Malicious Node Injection: An attacker deploys a rogue device on the network that can potentially be used to monitor the network traffic and intercept the communications. This attack can also be considered as a network based attack.

Social Engineering Attacks

- Social Engineering: Rather than the device itself, these psychological attacks are directed towards the user of an IoT device. The aim is to gather the user's trust towards sharing sensitive information [111].

Network-Based Attacks

Network-based attacks may be deployed on IoT ecosystems via the network from anywhere in the world. An attacker does not need to have physical access to the devices to launch attacks such as:

- Passive Traffic Analysis: The adversary passively monitors the network in order to try and intercept sensitive user data. Information can be gathered by monitoring the network's traffic. An attacker can also interfere with traffic flow, known as MITM [144].

- Spoofing: During such attacks, adversaries impersonate legitimate devices to make malicious entities seem as if they are legitimate ones so that attacks may be launched [218].

- Sinkhole: A compromised device attempts to attract network traffic by advertising fake routing updates. One of the impacts of such attacks is that it can be used to launch other attacks [149].

- DoS/DDoS: DoS and DDoS are the most common kinds of attacks that occur in IoT networks. Such attacks reduce, interrupt, or completely eliminate the network's communications and range from relatively simple jamming to more sophisticated attacks. They can be launched remotely and are difficult to detect before the network or service becomes unavailable [117].

- Sleep Deprivation: In IoT networks, the devices have the ability to enter power-saving modes in order to preserve their lifetime and reduce their power consumption. This attack focuses on forbidding these devices from going into power-

saving mode by continually sending traffic to them and therefore exhausting their battery resources [63].

- MITM attack: is an attack in which an attacker makes an independent connection with the victims B and C, making them believe that they are communicating with each other. The attacker intercepts the message coming from B to C, and vice versa, and re-routes them [139].

Firmware Attacks

Firmware attacks are associated with the software which is installed on the IoT device. More specifically:

- Backdoors: The attacker modifies the firmware of the device by inserting code which allows them to gain remote access to the device when it is connected to the network.
- Unencrypted Information: Attackers reverse engineer the compiled firmware and investigate it in order to find encrypted passwords, API keys, and public-key certificates. This information allows them to intercept the communication of the devices and gain access to sensitive user data.
- Malicious Firmware: Attackers may not only modify the device's firmware in order to create backdoors, but also to launch other attacks such as DDoS.

Encryption Attacks

Encryption attacks target any form of encryption mechanism that is employed on an IoT network [223], such as:

- Side Channel Attack: Attackers deploy this attack to bypass the encryption in IoT devices. This is possible by eavesdropping on the device's side channel emissions and waiting for an encryption key to be used to access the device.

- Cryptanalysis-Based Attacks: These attacks assume the possession of ciphertext or plaintext. Their purpose is to find the encryption key used by breaking the encryption scheme of the system. Examples of cryptanalysis attacks on IoT systems include Known-plaintext attack, Chosen-plaintext attack, Chosen Ciphertext attack, and Ciphertext-only attack.

- Ciphertext Only Attack: In this methid, the attacker can access the ciphertext and determine the corresponding plaintext.

- Known Plaintext Attack: In this method, the attacker knows the plaintext and a few parts of the ciphertext. The adversary's aim is to decrypt the remaining part of the ciphertext utilizing this information.

2.4.2 Consequences of Cyber Attacks on IoT

The consequences associated with the attacks discussed in Section 2.4.1 may vary, from digital loss of data to physical harm. For example, a seemingly harmless deauthentication attack can cause no significant damage. But if it is deployed on a device with critical significance, such as the steering wheel of a smart car, it can pose a threat to human life. Other high severity security attacks against, for example, CNI concepts, can disrupt the system's availability and energy resources, causing system blackouts and other indiscriminate and long-lasting damage. The effects of these security issues may cause major interference to the operation of services. For example, public transportation networks can be targeted to cause chaos during peak travel periods and attacks to power grids can result in wasting huge amounts of energy and a possible blackout of the system [63, 6]. In addition, such attacks may lead to financial implications. A recent survey by Irdeto Global Connected Industries [1] showed that IoT-related cyber attacks may have high financial impact. More specifically, 700 organisations in the transport, manufacturing, and healthcare domains have suffered financial losses that exceed $300,000 each due to IoT-related cyber attacks.

As a result, given that the most important aspect of IoT devices is their inter-connectivity, and ultimately, their connection to the Internet, network based attacks may impact the majority of devices. Along with the combination of the severity of the consequences that such cyber attacks pose towards IoT and their heterogeneity, this thesis is motivated by the research surrounding network based attacks in IoT; in particular, securing heterogeneous smart devices and detecting such type of attacks.

2.5 Mitigating IoT Cyber Attacks

There are several studies which concern the security of IoT. Such work focus on mitigating specific security elements, such as:

- **Authentication** - which aids IoT to distinguish legitimate source nodes and address identity-based attacks such as spoofing [207] (e.g.[219, 125, 160, 211]).
- **Access control** - which prevents unauthorised users from accessing IoT resources [59] (e.g. [213, 138, 113, 136]).
- **Confidentiality and Integrity** - which ensures that the data is only readable by the intended recipient and has not been altered (e.g. [132, 161, 89, 188]).

However, given the heterogeneity surrounding IoT, such security mechanisms may not be applicable to all of the devices connected to the network. Additionally, such security solutions may also generate a heavy computation and communication load specifically in outdoor IoT devices, such as cheap sensors with lightweight security protections [207].

As discussed in Section 2.3, IoT ecosystems do not consist of a single technology. Rather, it is an accumulation of various different technologies that work together as part of a robust architecture [177]. This emphasises that IoT is a heavily heterogeneous environment.

More specifically, in the context of IoT, heterogeneity refers to the implementation of a wide diversity of hardware, software, protocols, platforms, policies, etc. [153]. For instance, not all IoT devices are sold by the same vendor and are built for the same purpose. Depending on their applications and use case, IoT devices include several different features to severe different usages [62, 124]. Subsequently, IoT devices may be categorised into different types by taking into consideration parameters such as their size (e.g. small or large), their portability (e.g. portable or fixed), their power source (e.g. battery or external source), and their processing capabilities (e.g. commercial or embedded) [48].

In addition, depending on the task they were designed to do, IoT devices use different communication protocols to connect to the Internet, gateways, or with each other [62, 177]. Few of the most common communication technologies used among IoT devices include IEEE 802.15.4, low power WiFi, 6LoWPAN, RFID, Sigfox, LoraWAN, Bluetooth Low Energy (BLE), and Zigbee [177]. Each of these protocols have different attributes that make them suitable for different types of devices. For example, if a device has limited battery storage, it may be best to use BLE as it does not consume a lot of energy in comparison to other protocols [177].

It is therefore evident that due to the range of different technologies used within IoT infrastructures and the restriction of computational power of such devices, one of the biggest challenges posed in this infrastructure is the application of common security mechanisms [104]. As a result, it is critical to develop unified security mechanisms for heterogeneous devices in the IoT ecosystem.

One of the most current models to manage heterogeneous devices in a uniformed way is IoT hubs [215]. IoT hubs allow the control of multiple devices from one central point, which often allows additional security controls. Given this feature, the first research question which is explored in Chapter 3 is:

> ***RQ1:*** *What security mechanisms can be incorporated within a smart home IoT hub that can be uniformly applied to a range of heterogeneous devices?*

Section 2.5.1 provides an overview of current hub implementations in the IoT ecosystem.

2.5.1 IoT Hubs

The fundamental idea of IoT hubs is to allow devices and sensors with low computational power to focus solely on the task they are designed to do (e.g. measuring the temperature), without having to consider more complex functions (e.g. network connectivity, data processing) [214]. These functions are handled by the core of the hub, the gateway, which has significantly more computational power and storage in comparison to the devices and sensors themselves [215]. IoT hubs also have the capability of connecting a wide range of diverse smart devices and sensors (e.g. thermostats, smart locks, light sensors) by using wireless technologies.

There exist several frameworks which focus mainly on addressing heterogeneity and big data within IoT. Such frameworks include [147, 176, 101, 167, 58, 107]. However, such implementations have limitations which include limited inter-operation capabilities between devices from different vendors and protocol heterogeneity. Thus, to address the requirements needed by the communication interface, programming abstraction, and community-driven device support, Mozzami et al. [147] implemented SPOT, Extensible Markup Language (XML) based hub. SPOT uses device drivers to support additional devices. However, this technique can be extremely time-consuming and not user friendly. Additionally, this hub only includes OAuth authentication as a security mechanism, and the hub's overall security performance is not evaluated.

Saxena et al. [176] propose an implementation of an IoT gateway which is expected to operate over 5G networks, with the main focus being on resolving the challenge of resource constrained wireless networks. However, this approach does not address core security concerns, and similar to Mozzami et al. [147], was not evaluated against cyber attacks. Furthermore, Gloria et al. [101] designed an IoT gateway that addresses IoT

heterogeneity by supporting all the available wired and wireless communication protocols. However, this approach focuses only on low-level embedded devices and does not support commercial smart home devices. Additionally, this solution does not focus on security, and the authentication mechanism that is employed is primarily for platform configuration purposes. Moreover, to tackle heterogeneity and to control the large data volume generated by IoT, Razafimandimby et al. [167] proposed a Bayesian Inference Approach (BIA). This model is based on a hierarchical architecture which consists of simple nodes, smart gateways and data centers. It employs probabilistic techniques in order to avoid sending *useless* data to the network. Although this approach is successful in reducing resource consumption, it does not address the security of the IoT ecosystem. Alsheri & Sandhu [58] propose an architecture where virtual objects use publish and subscribe (PubSub) topics to deliver services to users through a security policy. Although this approach is successful in eliminating scalability, heterogeneity, and provides a secure access control mechanism, the system does not provide any other security mechanisms and has not been evaluated against a range of attacks.

Commercial implementations such as *Samsung Smart Things*, *Apple HomeKit*, and *Google Smart Home* provide similar solutions to Mozzami et al. [147], and are capable of supporting limited devices from various third party vendors. In addition, they may also be able to provide device cloaking and isolation if they are used in combination with a separate sub-network. However, this is subject to device and hub compatibility as the incorporation of IoT devices within a new sub-network may cause the functionalities and interactions with the included IoT devices to be lost (e.g. the Philips Hue lamps must be on the same sub-network as their interacting devices. Changing lighting settings with a smartphone is therefore not possible from the main network). In this case, such hubs by default do not provide this mechanism. It is also important to highlight, that often, these hubs are not able to inter-operate with each other.

Other technologies from vendors such as IFTTT, Nest Thermostat, and AllJoyn [18, 16, 29, 27] do not provide third-party extensibility or security mechanisms. As a result,

they cannot provide a unified interface, which further demonstrates that heterogeneity within IoT products and services is a current problem. Guoqiang et al. [107] propose a centralised framework which employs a gateway which allows different communication protocols to be plugged in to support various networks. Nevertheless, this approach requires advanced configuration techniques and changes in the physical hardware, which can be costly and is not easily extensible. Finally, this framework does not include a security policy or other mechanisms to improve the security of IoT devices.

To summarise these approaches, Table 2.1 describes existing frameworks, along with the the features they provide. A definition of these is as follows:

- **Authentication** - The hub provides a mechanism to verify the identity of a user (e.g. via the creation of an account)
- **Confidentiality** - The hub provides mechanisms to protect the data against unintentional, unlawful, or unauthorized access (e.g. via TLS or AES).
- **Access Control** - The hub provides a mechanism that regulates who and when they can interact the devices (e.g. via access control polices).
- **Device Cloaking** - The hub should camouflage the IoT devices, in order to increase the difficulty for an attacker to locate them.
- **Heterogeneity Aware** - The hub supports devices from different vendors.
- **Monitoring Attacker Behaviours** - The hub includes mechanisms that aid in monitoring user/attacker behaviours (e.g. using IoT canary functions)

It is evident that previous propositions, which aim to tackle the heterogeneity of IoT, do not specialise in improving the security of the ecosystem. Instead, the focus is on handling the big data that is generated by smart devices, attempting to overcome constraints of their computational power and their scalability. While security is a key concern within these existing approaches, the supported security requirements are limited and

only partially addressed as they may provide a maximum of two security mechanisms, leaving the hubs exposed to attackers.

Implementations	Authentication	Confidentiality	Access Control	Device Cloaking (by default)	Heterogeneity Aware	Monitoring Attacker Behaviours
Mozzami et al. [147]	✓	✗	✗	✗	✓	✗
Saxena et al. [176]	✗	✓	✗	✗	✓	✗
Gloria et al. [101]	✓	✗	✗	✗	✓	✗
Razafimandimby et al. [167]	✗	✗	✗	✗	✓	✗
Alshehri & Sandhu [58]	✗	✗	✓	✗	✓	✗
Guoqiang [107]	✗	✗	✗	✗	✓	✗
Samsung Smart Things [36]	✓	✓	✗	✗	✓	✗
Apple HomeKit [23]	✓	✓	✗	✗	✗	✗
Nest Hub [27]	✓	✓	✗	✗	✓	✗
Proposed Hub	✓	✓	✓	✓	✓	✓

Table 2.1: Existing implementations of hub frameworks in IoT ecosystems

2.6 Detecting Cyber Attacks in IoT

Incorporating security mechanisms into the initial design process of IoT devices or securing multiple heterogeneous devices using a hub, is critical when mitigating cyber attacks. However, in an attempt to bypass the system defenses, adversaries continuously develop new attack techniques to compromise the devices. As discussed in Section 2.4.1, given that the main characteristic of IoT devices is their connectivity, attackers may employ network-based attacks more often to compromise such devices as physical access is not required to launch these attacks. As a result, to enhance the security of IoT networks, several tools which focus on detecting attacks at the network-level have been developed (discussed in Section 2.6.2).

A traditional IT security ecosystem consists of a variety of tools which aid early attack detection and, subsequently, the mitigation of the deployment of a range of cyber attacks. Examples of such tools include static perimeter network defences (e.g. firewalls), ubiquitous use of end-point defences (e.g. anti-virus), and software patches from vendors.

One of the most effective and valuable security mechanisms for detecting malicious network behaviour is an Intrusion Detection System (IDS) [131]. Specifically, in the context of IoT, developing mechanisms that detect attacks at the network-level may also offer some significant advantages such as [187]:

- Network-level mechanisms can be implemented across a range of IoT devices, rather than device-level security which may be specific to a particular device/vendor.
- Network-level security tools can be enhanced on a continuous basis, unlike device-level security mechanisms which are hard to update and patch.
- Network-level security tools add an extra layer of protection that can augment any device-level security implemented by the manufacturer.

As a result, it is important to focus on developing more effective IDSs tailored for the IoT ecosystems.

2.6.1 Intrusion Detection Systems (IDS)

An IDS may generally be described as a software application or hardware appliance which monitors network traffic and alerts the system administrator if suspicious activity is found [131]. The design of an IDS often includes a data collection module which collects data from the network, an analysis module which processes the collected data in order to detect and identify malicious network behaviour, and finally an attack reporting mechanism which notifies the network administrators of such behaviours.

More specifically, IDSs may differ in their implementations of the following components:

Data Sources

- Host-Based: The IDS is deployed on the host device itself and often requires software to operate.
- Network-Based: The IDS is deployed on the network itself and analyses its traffic.

Detection Methods

- Signature-Based: The IDS observes network traffic to find the patterns and signatures of known attacks and abnormalities. This approach is considered as being unsophisticated as it does not learn the nature of the environment, and is thus ineffective towards unseen attack signatures.
- Anomaly-Based: Also known as 'behaviour-based detection', the IDS models the natural behaviour of the network and raises an alarm when it significantly

deviates from a calculated threshold from this state. Thus, such systems are assumed to be able to detect unseen attacks [184], though with the risk of generating a large amount of false positive alarms.

- Hybrid-Based: In an attempt to overcome their disadvantages, this IDS model is formed by using a combination of both signature-based and anomaly-based detection methods.

Times of Detection

- Online Networks: The IDS is deployed to detect attacks on a live network.
- Offline Networks: The IDS is deployed on data retrieved from a network which is no longer online.

Architectures

- Centralised: The IDS is deployed in a centralised manner when it is installed on a single host. A single IDS can monitor a wide subnet; however, it may be faced with a bottle neck when monitoring the traffic of a busy network.
- Distributed: The IDS is deployed in a distributed manner when it is installed on multiple nodes in the network. Multiple deployments of an IDS may be more suited for handling busy networks; though, it is more difficult to manage these multiple instances [157].

There currently exist various IDS solutions which have mainly been designed and implemented to satisfy the requirements of conventional IT networks. For example, popular IDSs, such as SNORT and Bro, are particularly effective on traditional IP-only networks as they are static and use signature-based techniques [215, 171].

In the context of IoT, early IDS frameworks were developed for Wireless Sensor Networks (WSN) (e.g. [216, 217, 54, 119]). However, due to the proliferation of IoT

devices and the increase in their heterogeneity, such IDS solutions, as well as IDS solutions for traditional IT infrastructures, are ineffective within current IoT environments as they are unable to adapt and scale to the device's range of possible normal behaviours, their heterogeneous platforms and protocols, and array of cyber attack threats [146, 215].

2.6.2 IDSs in IoT

A significant amount of existing examples of IDSs in IoT have based their detection methods on signature-based, anomaly-based, and hybrid-based implementations. However, the recent advancements in technology and the rapid increase in the amount of data produced from such technologies has opened a greater range of attacks making these techniques less effective in securing IoT devices [184].

To overcome this limitation, research has focused on developing machine learning based IDSs. Nevertheless, the majority of such systems are designed to detect one type of attack at a time, for example, DoS attacks such as Hello Flood (e.g.[190, 91, 193]), routing attacks (e.g. [166, 180, 158, 152]), and IoT botnet activity (e.g. [141, 145, 144]). Therefore, due to the significant increase of various types of cyber attacks in IoT, it is necessary to implement IDSs which focus on a broader spectrum of attacks. In addition, several of these systems have been evaluated using simulated network data. In this case, current IDSs have not been designed to detect cyber attacks in real IoT networks. Although their performances may be promising, it is necessary to take into consideration that simulated data may significantly deviate from real network data, as it may not be a true representation of how IoT devices are truly used.

Table 2.2 summarises existing state-of-the-art IDS solutions tailored towards the IoT ecosystem. In particular, they are categorised according to their detection methods, the security threats they detect, and their validation strategy (i.e. whether they are evaluated on simulated or real network traffic). As a result, it is evident that previous IDS

proposals dedicated for the IoT ecosystem are still at the early stages of development. Several approaches have used data from network simulations or have evaluated the system on a small array of IoT devices, which may significantly decline from a realistic environment. Additionally, such approaches focus on detecting whether specific cyber-attacks have occurred (i.e. whether packets are malicious or benign), but do not attempt to identify the type of attack. This is an important feature of an IDS, as specific countermeasures can be employed for specific attack types.

Work	Security Threat	Detection Method	Validation Strategy	Devices	Attack Classification
Stephen & Arockiam [190]	Sinkhole	Anomaly-based	Simulation	✗	✗
Raza et al. [166]	Sinkhole & selective forwarding	Hybrid	Simulation	✗	✗
Shreenivas et al. [180]	Routing attacks against RPL protocol	Hybrid	Simulation	✗	✗
Pongle & Chavan [158]	Wormhole	Anomaly-based	Simulation	✗	✗
Jun & Chi [118]	✗	Specification-based	✗	✗	✗
Summerville et al. [193]	Worm propagation, SQL code injection, & directory traversal	Anomaly-based	Empirical	2	✗
Midi et al. [146]	ICMP flood, replication, smurf	Hybrid	Empirical	2	✗
Thanigaivelan et al. [196]	✗	Anomaly-based	✗	✗	✗
Oh et al. [152]	Routing Attacks	Signature-based	Empirical	1	✗
Ioulianou et al. [116]	Hello Flood	Signature-based	Simulation	✗	✗
Shukla [181]	Wormhole	Machine Learning	Simulation	✗	✗
Doshi et al. [91]	DDoS	Machine Learning	Empirical	2	✗
Amouri et al. [60]	Identifies Malicious Nodes	Machine Learning	Simulation	✗	✗
McDermott et al.[141]	Botnets	Machine Learning	Simulation	✗	✗
Meidan et al. [144]	Botnets	Machine Learning	Empirical	9	✗
Restuccia et al. [168]	✗	Machine Learning	✗	✗	✗
Brun et al. [80]	UDP Flood, TCP SYN, Sleep Deprivation Attack, Barrage Attack, and Broadcast Attack	Deep Learning	Empirical	3	✗
Diro and Chilamkurti [90]	normal, DoS, Probe, R2L.U2R	Deep Learning	KDDCUP99	✗	✗
Rathore and Park [165]	✗	Semi-supervised learning	NSL-KDD	✗	✗
Canedo and Skjellum [82]	✗	Machine Learning	Simulation	✗	✗
Saeed et al. [174]	Malicious nodes	Deep Learning	Empirical (microsesors)	✗	✗

Table 2.2: Current work which focus machine learning based IDSs for IoT

2.6.2.1 Signature/Event/Rule Based IDSs

Stephen and Arockiam [190] suggest a lightweight, hybrid, and centralised approach towards the detection of Hello Flood and Sybil attacks in IoT networks, which use the Routing over Low Power and Lossy Networks (RPL) as a routing protocol. Their system is based on an algorithm that uses detection metrics, such as the number of packets received and transmitted, to validate the Intrusion Ratio (IR) by the IDS agent. Raza et al. [166] implemented a real-time IDS for the IoT called SVELTE. This system consists of a 6LoWPAN Mapper (6Mapper), a intrusion detection module, and a mini firewall. It analyses the mapped data to identify any intrusions in the network. Its performance in detecting various attacks seems promising. However, it has only been tested to detect spoofed or altered information, sinkhole, and selective-forwarding attacks. Shreenivas et al. [180] extended SVELTE by adding another intrusion detection module that uses an Expected Transmission (ETX) metric to identify malicious activity on the network. They also proposed a geographic hint to detect malicious nodes that conduct attacks against ETX-based networks. Their results demonstrated that the overall true positive rate increases when they combine the EXT and rank-based mechanisms.

Pongle and Chavan [158] propose a centralised and distributed architecture for a hybrid IDS, which they implemented based on simulated scenarios and networks. It focuses on detecting routing attacks such as the wormhole attack. Jun and Chi [118] presented an event-processing-based IDS for the IoT. This system is specification-based and it uses Complex Event Processing techniques for attack detection. This system collects data from IoT devices, extracts various events, and performs security event detection by attempting to match events with rules stored in a Rule Pattern Repository. Although it is more efficient than traditional IDS, it is CPU intensive. Summerville, Zach, and Chen [193] developed an IDS for IoT based on a deep packet analysis approach which employs a bit-pattern technique. The network payloads are treated as a sequence of bytes called bit-pattern, and the feature selection operates as an overlapping tuple of bytes called n-grams. When the corresponding bits matches all positions, a match

between the bit-pattern and n-grams occurs [175]. The system is evaluated by deploying four attacks and demonstrates a very low false positive rate.

Midi et al. [146] proposed Kalis, a knowledge-driven, adaptive, and lightweight IDS. It collects knowledge about features and entities of the monitored network and leverages it to dynamically configure the most effective set of detection techniques. It can be extended for new protocol standards, whilst at the same time providing a knowledge sharing mechanism that enables collaborative incident detection [175]. Results showed that the system had a high accuracy in detecting mainly DoS and routing attacks. Furthermore, Thanigaivelan et al. [196] proposed a hybrid IDS for IoT. In this system, each node on the network monitors its neighbor. If abnormal behavior is detected, the monitoring node will block the packets from the abnormally behaving node at the data link layer and reports to its parent node. Oh et al. [152], implemented a distributed lightweight IDS for IoT, which is based on an algorithm that matches packet payloads and attack signatures. They evaluate the IDS by deploying conventional attacks and by using attack signatures from traditional IDSs such as SNORT. The results demonstrated that this system's performance is promising. Finally, Ioulianou et al. [116] proposed a hybrid lightweight signature-based IDS, in an attempt to mitigate two variations of DoS attacks: "Hello" flood and version number modification. However, although their results look promising, their system is tested in a simulated environment using Cooja.

2.6.2.2 Machine Learning Based IDSs

The integration of machine learning in IDS systems has shown promising results and tackles the majority of the aforementioned limitations of existing systems. This is because such approaches allow more flexibility in the detection of cyber attacks, as they are able to automatically derive and learn patterns from network traffic and subsequently identify whether unseen incoming packets are malicious or not [154].

In machine learning, classification is the problem of identifying to which from a set of categories (i.e. malicious or benign) an unseen instance (i.e. network traffic) be-

longs to, based on a training dataset. Several machine learning approaches have been explored. The following Sections introduce the two state-of-the-art approaches used to support IoT network IDS applications, unsupervised, and supervised machine learning.

2.6.2.3 Unsupervised Machine Learning Approaches

Unsupervised machine learning differs significantly from supervised approaches. The key difference is that such approaches do not require a label dataset. The models use statistical inferences to learn structured patterns from unlabelled data. In general, unseen predictors which exhibit a distinct similarity in the features they contain are grouped together, subsequently classifying similar data points as the same target value. This common example of unsupervised learning is known as clustering. Another unsupervised learning technique is known as association. Such models allow associations among data objects within large datasets to be established. This unsupervised technique focuses on discovering relationships between variables to establish their target values.

Researchers have explored how unsupervised learning techniques may be applied in network intrusion detection. Early examples of applying unsupervised learning in IoT tailored IDSs include [174, 90, 52, 195].

In more detail, Meidan et al. [144] and McDermott et al. [141] both focus on the detection of botnets in the IoT ecosystem and employ deep learning techniques, such as autoencoders and Long Short Term Memory (LSTM)s, to achieve this. The results in both cases are promising as they can successfully detect the botnets; however, these methods have not been deployed to detect a range of attacks and have been evaluated in a simulated environment. Restuccia et al. [168] review the security threats in IoT networks and discuss a potential security solution which employs a combination of machine learning algorithms and polymorphic software and hardware. However, no description of the experimental setup, implementation, and evaluation of the proposed system is provided. Brun et al. [80] designed a deep learning based approach using

dense Random Neural Networks (RNN)s for the detection of network attacks. Although this approach often successfully detects attacks, the system was evaluated on a testbed consisting of only three devices and simplistic cyber attacks were employed. Additionally, the packet features were associated to specific attacks; for example, to identify DoS attacks, the frequency of packets over a specific period of time was observed, limiting the attack space.

Furthermore, Diro and Chilamkurti [90] designed a distributed attack detection mechanism for social IoT, which employs shallow neural networks. Nevertheless, this approach was evaluated using the KDDCUP99 dataset. Rathore and Park [165] introduced a fog-based attack detection framework that relies on the fog computing paradigm and a newly proposed ELM-based Semi-supervised Fuzzy C-Means (ESFCM) method. However, their system was evaluate using the NSL-KDD dataset. Saeed et al. [174] present an intrusion detection and prevention mechanism which employs RNNs. However, this work focuses on detecting out-of-bound memory accesses.

Although unsupervised approaches may seem as an attractive option due to the fact that network data does not need to be labelled, such approaches come with a few challenges. Firstly, when using unsupervised learning, it is not clear how many classes will be identified in the training set. Furthermore, it is often difficult to interpret how the model makes its decisions and how it is learning the patterns from the data. Finally, it may also be challenging to evaluate its performance without providing ground truth labels.

2.6.2.4 Supervised Machine Learning Approaches

In order to predict whether network traffic is benign or malicious, supervised learning methods depend on the existence of annotated training datasets. Training data usually consists of two parts; predictors and target values. Predictors are the data points (i.e. network packets) that need to be classified. These are often represented as feature vec-

tors; an n-dimensional array of numerical values that represent the presence of features in the network data. Target values, often referred to as class labels, are the known categories that a predictor belongs to. Supervised approaches are mostly used for classification problems as it is possible to clearly define the number of classes contained within a specific problem. As a result, the classifier can learn a decision boundary to distinguish different classes accurately. These features make this approach particularly attractive in the context of intrusion detection, particularly when distinguishing between benign and malicious packets, as well as the known devices on the network.

Several studies have developed supervised machine learning approaches to support IoT tailored IDSs. Amouri, Alaparthy, and Morgera [60] developed an IDS for IoT networks which employs linear regression. The IDS attempts to profile the benign behaviour of the nodes and identify anomalies in the network traffic. The results demonstrate that the system is able to successfully distinguish benign and malicious nodes. However, the IDS's performance is evaluated within a simulated network and not a real testbed. Therefore, further evaluation is required to test the efficiency of their system against a larger array of attacks and devices. Doshi et al. [91], investigated the effectiveness of a range of machine learning algorithms (e.g. decision tree, random forest, neural network) in IoT networks to detect DDoS attacks. They show that by utilising features relevant to IoT-specific network behaviors (e.g. limited number of endpoints and regular time intervals between packets), they can achieve high accuracy of DDoS detection in IoT network traffic with a variety of machine learning algorithms. Nevertheless, the experiments focus only on detecting DDoS attacks. Additionally, Shukla [181] proposed an IDS that uses a combination of machine learning algorithms such as K-means and decision tree, to detect wormhole attacks on 6LoWPAN IoT networks. The results from this work are promising; however, the evaluation of the proposed IDS was based on a simulation and the effectiveness of the IDS has not been tested against other attacks. Finally, Canedo and Skjellum [82] investigated how Artificial Neural Networks (ANN) can be used in a gateway in order to detect anomalies in the data sent from the edge devices. Nevertheless, their approach was evaluated using mostly

embedded devices/sensors such as Arduino Uno. Additionally, they focus on detecting modified data received from these sensors.

Finally, there are a few commercially available solutions that employ machine learning algorithms to detect cyber attacks provided by companies such as Darktrace [8] and Veracode [44]. However, there is no transparency of the methodology and algorithms employed by these companies and therefore it is not possible to directly compare this work with these products. Finally, they focus mainly on identifying malicious activity and do not attempt to classify the attack that is occurring on the network.

Nevertheless, these approaches are still at the early stages of development. More specifically, as shown in Table 2.2, the majority of these studies have used data from network simulations or have evaluated the system on a small array of IoT devices, which may significantly deviate from a realistic environment. Furthermore, the proposed systems focus on detecting one or two attacks; in particular, routing attacks and DoS. This motivates the second research question:

> ***RQ2:*** *Can supervised machine learning approaches support the automatic detection of a range of cyber attacks based on network packet features collected from a range of IoT devices?*

Additionally, as shown in Table 2.2, previous work lack focus in device profiling. This is important in networks consisting of various IoT devices as it allows the assets within the network to be identified, and subsequently it may be possible to detect anomalies outside of the device's 'normal' behaviour. Furthermore, understanding the behaviour of each device can assist in detecting malfunctions or attacks initiating from the IoT devices themselves. Although other work has considered IoT device classification (e.g. [145, 185, 72], such approaches utilise traffic flow features and other statistical attributes to identify the devices, such as device activity cycles and signalling patterns. Such metrics may be extracted using additional tools and software, which in the context of this work which utilises network traffic, would increase the time and effort to perform

feature engineering. Furthermore, device profiling has a received less attention in the context of IDSs and mostly focus on distinguishing between whether a device is IoT or non-IoT, or identifying the vendor of the devices. In this case, and to align with RQ2 which utilises packet features to distinguish between malicious and benign packets, RQ3 aims to answer the question of whether the same packet features used in RQ2 may also be used to successfully classify the devices connected on the network. This motivates the third research question:

> ***RQ3:*** *Can supervised machine learning algorithms successfully classify different IoT devices based on network packet features?*

To the best of our knowledge, to date, there is no published research that has shown the ability to identify attack types in IoT networks. Bolozoni et al. [78] propose a machine learning approach to classify the different types of cyber attacks detected by Alert Based Systems (ABS) in traditional IT networks. To achieve this, byte sequences were extracted from alert payloads triggered by a certain attack. Sequences were compared to previous alert data. Although this technique is effective in these networks, such approach relies on the alerts produced by the ABS, which may not be effective in IoT environments due to the large number of devices. Additionally, as the detection method uses payload values to detect attacks, attacks which IoT systems are vulnerable to and which do not alert the payload (e.g. DoS) are not detected. Subba et al. [192] implemented a model that uses feed forward and the back propagation algorithms to detect and classify cyber attacks. However, to evaluate their system, they used the NSL-KDD dataset which is collected from a traditional IT network. As a result, there is no evidence that this approach would be as effective if deployed in a heterogeneous IoT environment, which consists of many more protocols, devices, and network behaviours.

Table 2.2 highlights that current IDSs to not attempt to identify the type of attack that has occurred. According to the Cyber Security Incident Response Guide (CSIR) [7] and National Cyber Security Centre (NCSC), one of the toughest challenges for organisations is to identify the type of cyber attack which is occurring on the network without

having to perform an in depth investigation, which can be a very time consuming process. This may be a particularly challenging task in IoT networks due to the amount of connected devices and the larger attack vector. As a result, without this information, significant human effort is needed to respond to alerts, determine the severity of an attack, and launch countermeasures. This motivates the fourth research question:

> ***RQ4:*** *Given RQ2, can supervised machine learning algorithms further identify the main type of attack which has occurred?*

2.7 Adversarial Machine Learning

Section 2.6.2.2 provides an overview of machine learning and how such technologies can be used to automatically detect network intrusion. However, the introduction of such IDSs has also created an additional attack vector; the learning models may also be subject to cyber attacks.

The act of deploying attacks towards machine learning based intrusion systems is known as Adversarial Machine Learning (AML). The aim of AML is to automatically introduce slight perturbations to unseen data points in order to confuse the pre-trained model. The model's effectiveness may be reduced as it is presented with unseen data points that it cannot associate target values to, subsequently increasing the number of misclassifications and bypassing the detector.

Such techniques have been studied extensively in other research areas. Figure 2.5 illustrates an AML example in the area of computer vision, where calculated noise is introduced to the pixels in an image causing the trained model to misclassify it as a different object [103, 150].

Due to the advancement in machine learning, there has been a substantial increase in IDSs which use such techniques for IoT networks. Table 2.2 presents a summary of such systems and the machine learning methods they employ. Nevertheless, there has

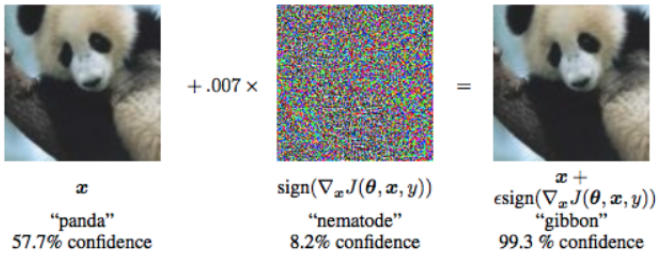

Figure 2.5: Deploying adversarial machine learning in images [102]

been significantly less focus on AML in this context. In the field of cyber security, such research has mainly focused on email spam classifiers [148], malware detection [105, 114, 74], and recently against network IDSs for traditional networks [169] and Industrial Control Systems (ICS) [222, 212, 93, 67].

In more detail, both Nelson et al. [148] and Zhou et al. [221] demonstrated that an adversary can exploit and successfully bypass the machine learning methods employed in spam filters by modifying a small percentage of the original training data. Moreover, Grosse et al. [105] evaluate the robustness of a neural network trained on the DREBIN Android malware dataset. They report that it is possible to confuse the model by perturbing a small amount of the features in the training set. Such an attack is considered as being a grey-box attack, as in order to be successful, the adversary must have access or knowledge of the dataset and the features it includes. Furthermore, Hu and Tan [114] proposed a more advanced adversarial technique which uses the concept of Generative Adversarial Networks (GAN) to successfully attack malware classifiers without requiring any knowledge of the data and the system. This is known as a black box attack.

In the context of IoT, there exist only a handful of investigations into AML attacks; the majority of which focus on machine learning detection methods for malware. Particularly, Abusaina et al. [50] investigated a range of off-the-self methods to craft adversarial IoT software and a Graph Embedding and Augmentation (GEA) method. The results show that all adversarial samples were successful in bypassing the detector.

Moreover, Liu et al. [110] developed a framework which employs genetic algorithms in order to generate adversarial samples for IoT Android applications. The framework demonstrated to have a success rate of nearly 100%. Furthermore, there exist a few studies that focus on detecting and defending against adversarial samples in IoT. For instance, Baracaldo et al. [74] use contextual information about the origin and the transformation of data points in the training set to identify perturbed data in a sensors' measurement dataset.

Recent work has focused on AML against traditional network IDSs and ICS. More specifically, Rigaki et al. [169] use the KDD'99 dataset to generate adversarial samples and demonstrate the effectiveness of AML against supervised algorithms. Moreover, Zizzo et al. [222] showcase a simple AML attack against an LSTM classifier which was applied on an ICS dataset. Yaghoubi and Fainekos [210] propose a gradient based search approach which was evaluated on a Simulink model of a steam condenser. However, this approach is efficient only against a handful of systems that may specifically employ RNN with smooth activation functions. Finally, Erba et al. [93] demonstrate two types of real-time evasion attacks, using RNN models and an autoencoder to generate adversarial samples.

Consequently, the existence of such techniques means that networks which incorporate machine learning based IDSs may be at risk of being vulnerable to such attacks. This leads to the fifth research question:

RQ5 *Can AML techniques be used to evaluate the robustness of a supervised IDS for the IoT?*

Finally, there is a need to develop methods to defend against AML attacks and build more robust machine learning models. Currently, a few methods towards defending AML attacks have been proposed in the literature. Two of the most popular techniques include adversarial training and adversarial sample detection. The former has been explored in the field of visual computing, where Goodfellow et al. [103] demonstrated

that re-training the classifier on a dataset containing both the original and adversarial samples significantly improves its efficiency against adversarial samples. The latter technique involves the implementation of mechanisms that are capable of detecting the presence of such samples using direct classification, neural network uncertainty, or input processing [222]. This motivates the sixth research question:

RQ6 *Can adversarial training enhance the robustness of a supervised IDS for the IoT?*

Evidently, it is understandable that machine learning based IDSs in the context of IoT require further evaluation against AML attacks.

2.8 Summary

This Chapter has explored the subject of cybersecurity in IoT. To gain a full understanding of the topic, the aims and objectives of IoT devices were discussed, as well as providing context for their applications. Additionally, the architectures of IoT environments were outlined. In addition to their advantages, it was identified that one of the disadvantages of IoT is their vulnerability to cyber attacks and their security flaws. Subsequently, the array of attacks that IoT ecosystems are most vulnerable to and the consequences of such attacks were described.

One of the key characteristics of IoT devices is their heterogeneity. In this case, this Chapter has focused on methods for both detecting cyber attacks in heterogeneous devices and the mitigation of such attacks. In particular, IoT hubs were discussed as a tool for controlling the security of multiple devices from one central point. Additionally, IDSs were identified as being one of the most effective security mechanisms for detecting malicious network behaviour. The different implementations of such systems were presented. Particular attention was paid towards machine learning based systems, including both supervised and unsupervised approaches. Finally, an overview

of how the learning models which support such IDSs may be subject to AML attacks was provided.

Finally in this Chapter, the relevant work surrounding the aforementioned IoT subjects was also discussed and a range of research gaps were identified. This led to the development of six research questions which are addressed in the respective contribution Chapters:

Contribution 1: Chapter 3 contributes towards addressing RQ1, where a novel design of a secure and heterogeneity-aware IoT hub prototype and network architecture is proposed. Although other hubs exist, they mainly aim to tackle the heterogeneity and big data in IoT and do not focus on security. Such approaches may include one or two standalone security mechanism(s) such as authentication or access control, leaving other security vulnerabilities exposed. In particular, this thesis aims to develop a secure and heterogeneity-aware hub that is able to defend against two attacker models which take into consideration some of the most popular attacks that may threaten IoT smart home networks. To achieve this, the hub contains the following security mechanisms: policy-based access control and secure user authentication, transport encryption, limits exposure of local IoT devices through *cloaking*, and offers a *canary-function* based capability to monitor attack behaviours. Finally, the hub uses dynamically loadable add-on modules to communicate with various diverse IoT devices, tackling heterogeneity. The hub's architecture and implementation are discussed, along with its use within a smart home testbed consisting of commercially available devices. The effectiveness of the hub was further evaluated by simulating various attacks such as spoofing, MITM, and DoS. The results demonstrated that direct attacks upon the hub were successfully mitigated due to the security techniques used.

Contribution 2: Chapter 4 contributes towards addressing RQ2, RQ3, and RQ4. Although there has been substantial research revolving around machine learning based IDSs in IoT, such approaches come with a range of limitations. Firstly, the majority of these systems have been evaluated on simulated data or on a very small number of

devices, which may significantly deviate from a realistic environment. Secondly, they lack focus on device classification and profiling, which aids in IoT asset identification and detecting anomalies outside of the device's 'normal' behaviour. Thirdly, they focus on detecting one or two attacks and fourthly, they do not attempt to identify the type of attack that has occurred. Without this information, significant human effort is needed to respond to alerts, determine the severity of an attack, and launch countermeasures. This may have severe consequences ranging from data and financial loss to physical harm. To address the aforementioned limitations, this Chapter introduces a novel design of a supervised three layer IDS which: 1) classifies the type and profile the normal behaviour of each IoT device connected to the network, 2) identifies malicious packets on the network for a range of cyber attacks, and 3) given 2) classify the type of the attack that has been deployed. To investigate the effectiveness of the system, the Chapter presents the initial classification experiments of an IoT tailored IDS which utilises network data from a smart home IoT testbed. Furthermore, the system is evaluated against 12 attacks from four main categories. The performance of the system's three core functions resulted in an F1-score of over 90%, demonstrating the efficiency of the proposed approach.

Contribution 3: Chapter 5 contributes towards addressing RQ5. Although there has been substantial research in machine learning based IDSs for IoT, there has been significantly less focus on AML. The use of machine learning in such systems has introduced an additional attack vector; the trained models may also be subject to attacks. This may be achieved by introducing calculated perturbations to malicious data points. Such attacks are critical as they can be exploited by adversaries in order to bypass the supervised based detector. Subsequently, this Chapter provides the first analysis on how AML can be applied in the context of an IoT smart home network. The analysis focuses on manipulating DoS packets, one of the most critical attacks against IoT devices. The results demonstrated that by modifying a selected set of features, the classifier's performance decreased by a maximum of 31.7 percentage points.

Contribution 4: Following the positive findings from Contribution 3, Chapter 5 also contributes towards addressing RQ6. Specifically, it explores how adversarial training may contribute towards increasing the robustness of a supervised IDS for IoT against AML attacks. By including adversarial samples in the training set and re-training the model, the results show that the classifier's performance increases, showcasing the effectiveness of this approach.

Chapter 3

A Secure and Heterogeneity-Aware Hub for IoT Smart Homes

3.1 Introduction

One of the core characteristics of the IoT ecosystem is the heterogeneity of devices. This attribute significantly expands the attack surface of IoT networks. As a result, implementing universal IoT security mechanisms is a complex challenge. One of the most current models to manage heterogeneous devices in a uniform way is IoT hubs [215]. Hubs allow the control of multiple devices from one central point, which often allows additional security controls.

Although there exist several IoT hubs (e.g. [147, 176, 101]), such approaches have primarily focused on tackling device heterogeneity and may partially include one or two standalone security mechanisms such as authentication or access control, leaving other security vulnerabilities exposed. Furthermore, commercial hub implementations (e.g. [36, 23, 41]), have limited inter-operation capabilities between devices from different vendors, and often prioritise connection with devices from their own vendor and pay less attention to the security of the IoT network. As a result, such hubs, and subsequently the IoT devices they control, are still vulnerable to adversaries.

To combat the aforementioned limitations, the work presented in this Chapter focuses on a novel and heterogeneity-aware hub design and implementation for the IoT which

is able to defend a smart home ecosystem against two attacker models. Such attacker models are based on existing literature (e.g. [151, 96, 98]), and take into consideration some of the most popular network based attacks one would expect to target a smart home environment [56, 31]. Given that previous work is yet to be able to defend against such attacks, this Chapter presents the first design of a secure and heterogeneity-aware hub that is capable of securing against such attacker models. More specifically, the proposed hub was designed to contain five built-in security mechanisms that provide secure authentication, policy based access control, capability to monitor attack behaviours using IoT canary functions, device cloaking and isolation, and confidentiality.

This aims to address the first research question introduced in this thesis:

> ***RQ1:*** *What security mechanisms can be incorporated within a smart home IoT hub that can be uniformly applied to a range of heterogeneous devices?*

In answering this question, the following research contribution is made:

> ***C1:*** *A novel design and prototype implementation of a secure and heterogeneity-aware hub for the IoT, which significantly increases the security of a smart home network.*

More specifically, this Chapter is divided into the following main sections: Section 3.2 identifies the vulnerabilities of a traditional smart home network configuration and presents the two attacker models considered. Section 3.3 presents an overview of the proposed hub architecture and its security properties. Section 3.4 presents the prototype implementation of the hub. Section 3.5 discusses the security evaluation of the hub using a penetration testing methodology. Finally, Section 3.7 summarises the findings and highlights the research contributions of this Chapter. Parts of this Chapter have been published in [62].

3.2 Traditional Smart Home Topology

One of the most popular approaches to IoT networking in a smart home is a star topology, where all the smart devices communicate to a central access point [87, 133, 191]. More specifically, in a typical smart home, traditional computers and smart devices gain Internet access by connecting to the same access point. Figure 3.1 illustrates an example of a conventional network topology of such ecosystem. This particular example of a network topology consists of one central router provided by the Internet Service Provider (ISP), conventional IT devices (e.g. laptops), and IoT devices which are connected to the access point by a power line adaptor across Wi-Fi or Ethernet whilst the user is able to control them via the individual mobile/web applications they provide. It is worth highlighting that smart home devices often support multiple users at the same time. As a result, more than one user can control and change the settings of smart devices in the network.

Vendor specific devices and sensors, such as the Philips Hue lamps and the Samsung Smart Things motion sensor, are connected to their control hubs via Zigbee or BLE. It is worth noting that the IoT devices considered herein communicate through a wired or WiFi network connection, and that media such as Bluetooth, ZigBee, Z-Wave, and 6LowPAN are out of the scope of this thesis.

Nevertheless, although this is a traditional network configuration, the smart devices are exposed to anyone that may have access to the local network, subsequently increasing their security risks. As a result, the devices may be vulnerable to various wireless attacks such as passive sniffing, MITM attacks, DoS, and more [66].

3.2.1 Threat Model

There are many reasons why an adversary may wish to compromise IoT devices within a smart home. Firstly, IoT devices such as smart cameras may be used for physical security purposes. Therefore, attackers may use active attacks (e.g. DoS) to cause a

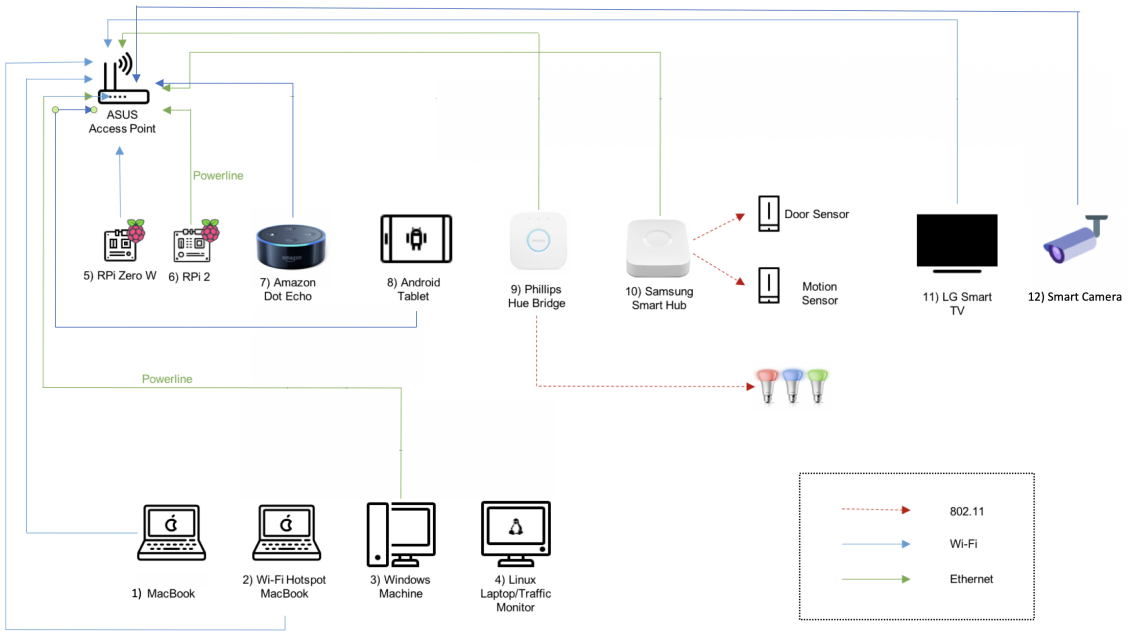

Figure 3.1: Traditional smart home IoT network illustrating multiple means of connection.

camera to black out, allowing the coast to be clear to physically access a home without creating digital forensic evidence [151]. Devices such as smart robot vacuum cleaners, which have some mobility around the house, can potentially provide adversaries with information about the home's layout. This information can be used to further plan activities and movements [20]. Furthermore, other devices such as wearables may have access to sensitive user information such as usernames, passwords, and even credit card details which may be compromised. Moreover, an adversary can hijack, and subsequently control an IoT device. Such attacks are considered difficult to detect as the adversary does not change the basic functionality of the device in obvious ways. More specifically, an adversary may compromise a smart lock and may choose to unlock it in an unusual time frame (e.g. in the middle of the night). Finally, an adversary may passively monitor the network traffic of a smart home in order to perform privacy based attacks. Research has demonstrated that a smart home network observer can infer privacy sensitive in-home activities by analysing Internet traffic from smart homes containing commercially available IoT devices even when the devices use encryption

[68].

3.2.1.1 Attacker Models

It is evident that IoT infrastructures may be subject to most types of network attacks as discussed in Chapter 2. By exploiting these attacks, adversaries can intercept, interfere, or change the behavior of the IoT devices in various ways. Some of these approaches may require physical access to the devices - which can significantly increase the difficulty of launching the attacks; but other attacks, which are often the most severe, can be deployed over the Internet from near proximity or a remote location [130]. Such attacks can target specific IoT devices directly i.e. by taking advantage of search engines that index vulnerable IoT devices in the world, such as Shodan [39] or ZoomEy [47]. Alternatively, adversaries can compromise a traditional device and then target the IoT devices from this point. In the former case, it is not guaranteed that the IoT devices will be indexed by these search engines. Therefore, if an adversary has a specific target in mind, they may not be able to find any relevant vulnerable devices online. It is also worth noting that given that the testbed was deployed within an actual home, adding an IoT device on these search engines to conduct experiments was considered too risky. Additionally, due to the limited capabilities and processing power of the devices, compromising one specific IoT device would significantly limit the attacker's range/scope of the attack as they might not be able to pivot to other vulnerable IoT devices within the network. However, in the latter case, an adversary will be able to cause maximum damage by deploying a range of attacks against any devices within the smart home. As a result, given the high impact of such attacks, the experiments presented in this Chapter focuses on replicating the possible attacking behaviour deployed by the aforementioned adversaries.

Subsequently, based on the IoT smart home network topology discussed in Section 3.2, and the possible network threats faced by a smart home [31, 56], the experiments herein consider two attacker models which utilise two possible and realistic methods of gain-

ing access to the smart work network. Such attacker models can cause the most damage by enabling the adversary to deploy a range of attacks targeting multiple devices. These models are based on those presented in other relevant work [220, 151, 96, 98, 109] and include the most popular network-based types of attacks that IoT devices are vulnerable to. Subsequently, the focus of this Chapter is to develop a hub framework which is able to defend IoT devices from attacks as part of the aforementioned attacker models [220, 31, 56].

Attacker Model #1

In the first model, it is assumed that the attacker does not have physical access to the IoT devices, but has successfully retrieved the password for the central access point within the smart home network. This type of attacker may be physically located within the wireless range of the targeted user's smart home network. An attacker with control over the wireless router can access devices over the local network and can deploy several different attacks [198, 199, 200]. Such an attacker may have a pre-existing relationship with the victim and was given administrative access to the router/network [151] when they were present in the home. The purpose of this attacker is to actively deploy attacks upon the IoT devices and network in order to obtain more information about the devices, to make them unavailable to the users, and to intercept sensitive data. More specifically:

The attacker has the following capabilities:

- Scan the network.
- Passively eavesdrop on the wireless communications.
- Deploy active attacks such as MAC/ARP Spoofing, DoS, and MITM.

The attacker may have the following objectives:

- to collect information about the connected devices (i.e. what devices are connected, what ports are open).

- control the IoT devices.
- intercept sensitive user data.
- make the devices unavailable to the intended user.

Attacker Model #2

In the second attacker model, it is considered that an attacker has remotely compromised a device (e.g. laptop) and has gained access to the main smart home network by utilising e.g. a phishing attack. For the purposes of the experiments herein, this adversary is assumed to have the same privileges as a legitimate user. The objective of this adversary is to gain unauthorised access to the IoT devices and their functions. More specifically, they attempt to remain undetected by not interrupting the device functionality or by actively deploying attacks. Their goal is to use the legitimate device functionalities to cause further damage (e.g. unlocking the smart lock in the middle of the night).

Subsequently, as highlighted in Table 2.1, although existing implementations may provide one or two security mechanisms, the aforementioned attackers would still be able to deploy direct attacks on the hubs to make them unavailable to the user (i.e. DoS), scan the network to identify the connected devices, perform MITM attack to intercept sensitive data, and access the devices.

3.3 Hub Architecture Overview

To combat the attacker models discussed in the previous Section, this Chapter proposes an architecture of a novel secure and heterogeneity-aware hub. More specifically, Figure 3.2 visualises the main hub architecture, which consists of three main components; a gateway, a policy server, and a sub-network. The IoT devices, together with the gateway and policy server, are placed within a separate isolated sub-network. The rationale

behind this is so that the sub-network camouflages the aforementioned components and the IoT devices so that they are no longer visible to the users that join the main network. At the same time, devices on the sub-network are no longer able to connect or access other devices on the main network. This prevents the spread of attacks and isolates possibly problematic devices that cannot be immediately taken offline.

Within the sub-network, the gateway of the proposed hub is the core of the network and the only externally visible device to the user. Subsequently, the gateway is the main point of communication to the other IoT devices. The gateway is able to communicate securely with a range of commercial and embedded devices. Furthermore, users are able to interact with the gateway over a secure communications channel which is established after a three-step authentication process is completed. The communication channels between the gateway and the user and gateway and IoT devices are encrypted and protect the data in transit from eavesdropping.

Once a user has authenticated, they can begin to interact with a device which is connected to the network by sending their request to the gateway. Prior to this request being fulfilled, it is redirected by the gateway to the policy server. The policy server then queries the policy database in order to determine whether the request satisfies the predefined attributes associated with the user making the request. Depending on the result, the policy server grants or rejects the request. If the request is denied, the policy server also triggers a specific action i.e. notifies the administrator. Finally, to monitor user access to the devices, IoT tailored canary functions are incorporated to the available modules in the gateway in order to monitor unauthorised access to the IoT devices.

Table 3.1 summarises the security properties that are incorporated within the proposed hub and can defend against the aforementioned attacker models. Section 3.4.3 discusses how the hub is able to communicate with various diverse devices. This is achieved by using custom add-on modules and device Application Programming Interface (API)s.

Attacker Objective	Hub Security Mechanism
Collect information about the connected devices (i.e. what devices are connected, what ports are open?)	The hub camouflages the IoT devices, in order to increase the difficulty for an attacker to locate them. Section 3.4.5 describes a sub-network configuration in which the hub and IoT devices are camouflaged and no longer visible to users in the main network. **(Device Cloaking)**
Unauthorised control/access to the IoT devices	Section 3.4.2, introduces a three-step authentication process contained within the hub. **(User Authentication)** To further enhance the security of the system, Section 3.4.4.1 presents the implementation of a policy server which allows secure access control. **(Access Control)**
Intercept sensitive user data	The communications between the user and the hub should be encrypted. This is achieved by configuring the hub to use Transport Layer Security (TLS) and Advanced Encryption Standard (AES). **(Confidentiality)**
Make the devices unavailable to the intended user	The hub limits the exposure of the IoT devices (Device Cloaking), increasing the difficulty of an attacker to deploy direct attacks against these devices (e.g. DoS), to them unavailable to the user.
If an adversary has the same privileges as a legitimate user, they access the IoT devices and their functions.	Section 3.4.4.2, describes how the user/adversary behaviour can be monitored/detected by using IoT canary functions. **(Behaviour Monitoring)**

Table 3.1: Hub security properties which aid in securing smart home IoT devices

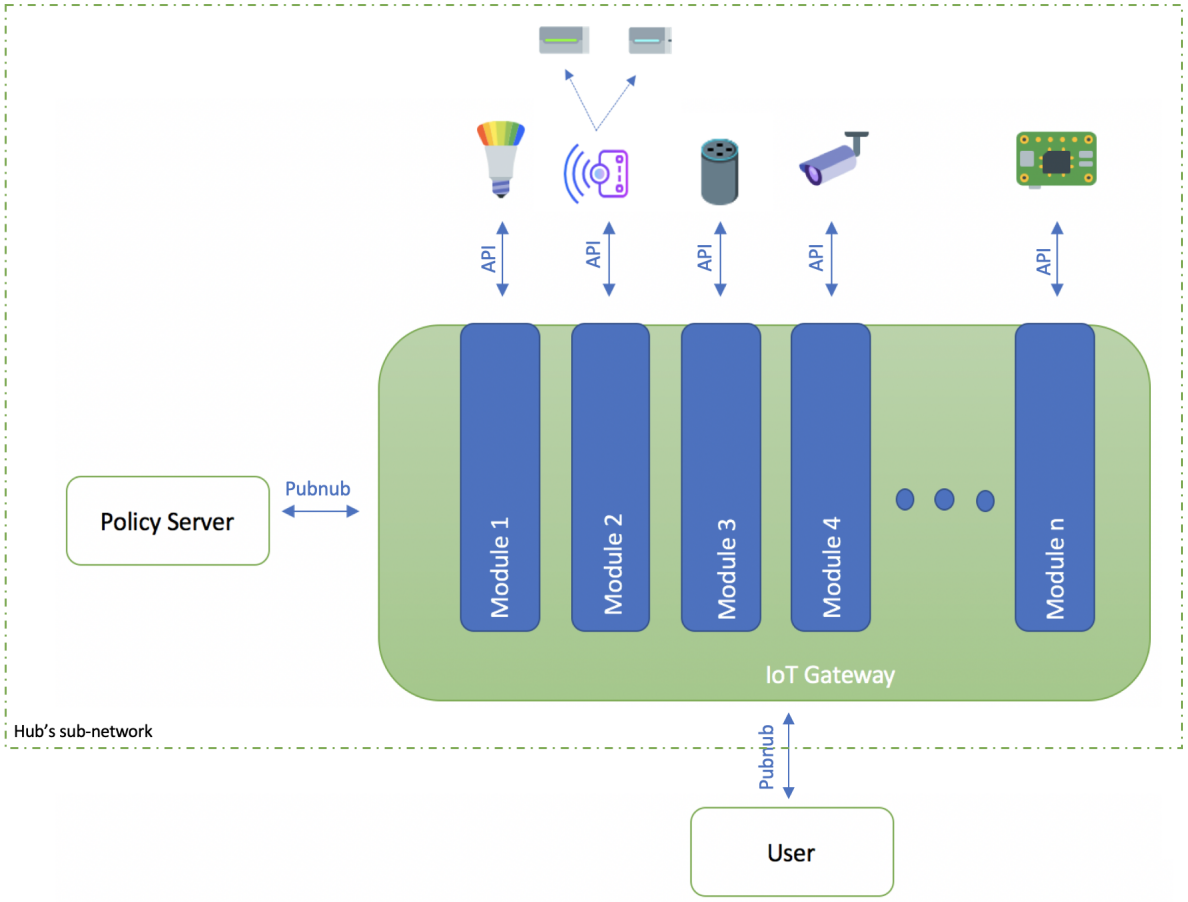

Figure 3.2: Proposed hub consisting of a gateway, a policy server, and a sub-network.

3.4 Hub Prototype

To investigate the feasibility of the hub infrastructure discussed in Section 3.3, a prototype of the proposed gateway scheme was implemented, and its application within a smart home IoT testbed was evaluated. Sections 3.4.1 and 3.4.4.1 discuss in more detail the configurations and the technologies used in both the gateway and policy server respectively. Finally Section 3.4.5 discusses the integration of the hub within a sub-network.

3.4.1 Gateway

The following Sections discuss the gateway components and their functionalities in more detail.

3.4.1.1 Cloud Service Provider

In the case of many IoT-driven enterprises, the choice is often between establishing a costly, vulnerable, and non-scalable on-site server infrastructure, or choosing an IoT cloud-based solution that allows for avoiding unnecessary costs while presenting no major trade-offs on platform functionality [156, 77, 3, 28].

Several works surrounding IoT (e.g. [189, 53, 81, 92, 94]) review the benefits of cloud-based services, noting that some of their greatest advantages include their:

- Increased Data and Processing Power: Cloud services provide enormous storage facilities and processing power which support the analysis of large and complex datasets generated by IoT devices.
- Scalability: As the need for new computing resources and services arises, the large routing architecture of the Cloud's model allows IoT-based applications to scale up to larger sizes based on the new requirements without having to invest

heavily in the new resources added [92]. Conversely, in on-premise IoT network infrastructures, scaling up requires purchasing more hardware, investing more time and increasing configuration efforts to make it run efficiently.

- Accessibility and Collaboration: Cloud allows IoT data to be accessed from anywhere and anytime as long as the IoT's have Internet connection [81] and shared among different IoT applications and a group of users. Such accessibility is especially important when concerning IoT projects which involve real-time monitoring and management of connected devices.

- Dynamic Provisioning of Services: An advanced IoT cloud provider offers the tools to provision, manage, and update the IoT devices and process and communicate with the acquired data remotely and in real-time, whilst adapting to the rapidly changing business need and policies in a cost-effective manner [127].

- Multi-Tenancy and Resource Optimisation: Allows instances of IoT applications to share the same service infrastructure of Cloud. It also allows integrating several services (Infrastructure, platform, and software) from different service providers available on Clouds and to meet the tailor-made demand of the user. This reduces the cost of operation and gains in the service's quality.

- Cheaper Upfront and Long Term Costs: There is often a large initial upfront investment and increased risk when implementing an in-house IoT system. In addition, there is the issue of ongoing hardware maintenance and IT personnel costs. From the Cloud perspective, these challenges are addressed as there are significantly reduced up-front costs, there is often a flexible pricing scheme based on the actual usage, and the Cloud providers are responsible for updating and maintaining the hardware and the overall infrastructure.

However, as cloud servers are typically more targeted by cybercriminals, one of the biggest factors which may influence migrating from an on-premise IoT infrastructure to the Cloud is the inherent security risk to critical data. Companies have to trust that

providers have safeguards in place against vulnerabilities. In the event of data leaks, businesses will be at the mercy of providers. However, it is important to note that most reliable providers operate under strict security regulations. In addition, any problem on the Cloud may suspend access to data, whilst on-site solutions still support mission-critical systems [94].

In the hub infrastructure presented herein, given the aforementioned discussion, the communication links between the gateway and the user and the gateway and the policy server are supported by a cloud service provider. For this work, the suitability of a range of popular cloud service providers which allow users and IoT devices to interact with each other was examined. Some of these providers included Dweet.io [11], KURA [12], and Pubnub [35].

Some cloud service providers are subject to privacy and security concerns. For example, it is anticipated that Dweet.io may not be suitable for enterprise-level security unless a specific licence is purchased. Moreover, although KURA provides some security mechanisms such as TLS, it does not allow provisioning the security through the cloud or the configuration of the security post-deployment.

Subsequently, after evaluating the features of the aforementioned technologies, it was determined that Pubnub was the most suitable cloud provider. Pubnub is an Infrastructure as a service (IaaS) which allows users to control and connect various heterogeneous IoT devices. Their API provides device monitoring and security provisioning capabilities to connect devices and to create multiple IoT gateways. They also supply well-documented references for a range of programming languages and operating systems.

More specifically, Pubnub's free tier was used for this prototype. This access provides a service for up to 100 devices and 1 million messages per month. With the employment of Pubnub, the hub provides a middleware solution which uses a Publish-Subscribe (Pub-Sub) architecture with security support. More specifically, a publisher (i.e. any source of data) pushes messages out to interested subscribers (i.e. receivers

of data) using live-feed data streams known as channels or topics. The subscribers of a specific publisher channel are immediately notified when new messages have been published on that channel, and the message data (or payload) is received together with the notification [34]. This technique allows users who interact with IoT devices to authenticate with the gateway using a three-step process, which ultimately creates a secure channel. The channel is analogous to a topic, in which the gateway and the user both provide permission to publish and subscribe. Requests by the users and responses from the gateway are securely passed through PubNub's DSN. On top of these aforementioned features, due to the secure nature of Pubnub, by using this service to interact with their smart home devices instead of the native mobile applications for each device, users can also be sure that their data is not shared with any third party providers for marketing purposes. For instance, the SmartThings app collects and shares personal user information with providers such as Google Analytics [33].

Table 3.2 presents the main channels that were implemented for the secure IoT hub.

Channel Name	**Description**
policy_server	The gateway forwards the users' request to this channel, where they are being processed/validated by the policy server.
gateway_auth	When a user joins this channel, their presence is detected via PubNub's presence() function and the 3-step authentication process begins.
admin_channel	This channel is generated once the gateway is initialised. It is used for administrative tasks such as creating add-on modules remotely via PasteBin.
uuid_channel(s)	Every user has a unique channel associated with their UUID.

Table 3.2: Main PubNub communication channels

3.4.2 Authentication Mechanism

Prior to sending instructions to the hub's gateway, users are required to complete a three-step authentication process. Once a user has authenticated, a secure communication channel using TLS is created. In addition to transport encryption, it is also possible to employ AES256 algorithm to achieve application level encryption. PubNub provides the main functionality for both mechanisms. In more detail, the authentication process is as follows:

1. Firstly, the user is required to join the *Gateway Authentication (gateway_auth)* channel by subscribing to it. Once the client is subscribed, the gateway is able to detect the presence of the user. Then, the gateway creates a channel where the channel's name corresponds to the user's Unique User Identification (UUID), which is a unique string of up to 64-characters that is used to identify a client (end user, device or server) that connects to the PubNub platform. If a UUID is not set by the user, PubNub Software Development Kit (SDK) will generate a random one.

2. Once the UUID channel is created, the gateway computes the hash of the user's UUID number and forwards it to the user over the *Gateway Authentication* channel. The user is notified that their channel has been established. When the user receives the hash from the gateway, unless it is already computed, the hash of their own UUID must be created and compared with that which is received.

3. If the two hashes correspond, the user is able to join this specific UUID channel, establishing a one-to-one communication with the gateway.

4. At this point, the gateway must ensure that only one user is on the channel and that the correct user is on the correct channel by checking the user's UUID. If these two conditions are met, the gateway will send an authentication key to the client over the UUID channel. This particular configuration also allows for

the AES256 key to be sent over the same channel, allowing the user to send encrypted requests.

5. Lastly, the secured communication is formed by using a secure/locked channel, which utilises the TLS protocol.

Enhanced Confidentiality using AES

As mentioned in the previous Section, PubNub's channels employ TLS/SSL in order to achieve point-to-point encryption. With TLS/SSL enabled, the data is encrypted as it traverses through the network/Internet, but is decrypted, processed, and re-encrypted as it passes through the PubNub network. To ensure the highest levels of message integrity and end-to-end data security, it is possible to use TLS/SSL in combination with AES256 in order to achieve application-level encryption of the payload. As a result, even if an adversary manages to perform a MITM attack and strip the TLS/SSL, the data will still be encrypted [13]. In this work, two versions of the prototype were implemented: TLS-only and TLS in combination with AES256.

3.4.3 Module-Based Adaptation

The hub's primary function for handling device heterogeneity is achieved by the inclusion of add-on modules within the gateway. These represent the interfaces of each smart device connected on the network and are responsible for various device features, such as adjusting the brightness of a smart lamp. The add-on modules are implemented in Python and support a range of IoT devices. The modules interact with the devices using their API. If there was no API associated with the device's vendor, an API developed by a third-party was used. All the APIs used in this implementation employed TLS, thus the communication between the gateway and IoT device was also secure. However, due to compatibility issues, the add-on module for these devices had limited functionality. Nevertheless, this demonstrates that add-on modules for a range of commercial and embedded devices can be written and installed in the hub's gateway,

making it possible to communicate with various heterogeneous devices. Table A2 in the Appendix describes the modules that were implemented for each device and the functions they contained.

3.4.4 Access Control

As discussed in Section 3.2.1.1, the second attacker model may attempt to gain unauthorised access to the IoT devices. To prevent this threat, the hub incorporates two mechanisms for access control: a policy server and IoT tailored canary functions.

3.4.4.1 Security Policy Server

The security policy server operates separately to the gateway. Each request sent by a user to interact with the smart devices within the sub-network is forwarded to the gateway. Before being fulfilled, the requests are sent from the gateway to the policy server. The main role of the policy server is to ensure that each request complies with the policies that are defined on the server. These requests are verified by the policy server, which queries the database for various security attributes i.e. the time of the request. Once the result is determined, if it complies with the policies, the gateway executes the IoT function that the user has requested. Otherwise, the user is informed that their request was rejected. The policy server logs all the access attempts. An administrator can set access policies that take into consideration attributes such as time of the request and previous failed attempts. For the purposes of this prototype, three examples of policies were implemented. These included:

- The time of the request is within the acceptable access time-frame for the particular module.
- The user has not been rejected access three or more times in the same day [79].

- The user's latest access was not rejected under a minute prior to the current access request [79].

The current implementation of the policy server also allows for a versatile remote configuration, which enables administrators to remotely call functions in order to create new policies or modify existing ones.

3.4.4.2 IoT Canary Functions

To further enhance the security of the proposed hub, and particularly the access control, IoT tailored canary functions were implemented. These functions are complementary to the policy server and offer an extra layer of defence. They are used mostly for detecting unusual user/adversary behaviour whilst the policy server aims to prevent unauthorised access to the devices. These functions are based on the canary files used within traditional IT systems. A canary file is a forged file which is typically placed amongst genuine ones in order to support the early detection of unauthorised data access, copying, or modification [204]. Its name originates from the use of canaries as sentinel species within coal mining environments to warn workers of the build up of gasses such as carbon monoxide underground [112]. In this context, if an unauthorised user attempts to interact or gain access to these functions, a specific action is triggered.

In the proposed implementation, bogus add-on modules and functions that represent sensitive operations of IoT devices were created and deployed. These are displayed to users as regular module functions. In order to lure attackers, the canary functions represented bogus sensitive and important device operations. For example, a module named *Smart Lock* is assumed to operate a smart lock, and it contains a function called *get pin*.

In the proposed implementation, various severity grades that correspond to different actions were implemented. More specifically, the severity of the canary function is classified according to one of the following three categories:

- Grade A - Severe: A function classified as grade A would prevent unauthorized access by causing the system to shut down and notify an administrator by email.
- Grade B - Average severity: A grade B canary would notify the administrator by email and blacklist the user from further access.
- Grade C - Least severe: A grade C function would only notify an administrator by email and no further action would be taken.

As a result, if a user has requested to access any of the functions that are enlisted as being bogus, the policy server will query the database to inspect the severity of invoking the specific canary function and will trigger a possible action.

3.4.5 Network Configuration of the Hub

A second access point (router) was used to create a sub-network to which IoT devices, irrespective of their manufacturer, can connect and communicate securely. Within the sub-network, the gateway of the proposed hub is the core of the network and the only externally visible device. Moreover, the gateway is also the main point of communication to the other IoT devices that sit within the sub-network. The sub-network configuration provides additional security, as users who join the main/local network can no longer view the connected smart devices as they are camouflaged. This is due to the 'hidden' setting of the sub-network's router. It is worth highlighting that security through obscurity is not considered to be among the best practises when used as the principal means of security. However, in this use case, it is used as an additional feature to support a more robust and secure architecture.

Furthermore, devices on the sub-network can no longer connect or access other devices on the main network due to the IP Access Control Lists (ACL) configurations on the sub-network's router. From a security perspective, this configuration reduces the risk of attacks that are targeted towards vulnerable devices (e.g. default password attacks),

as well as aligning IoT networks with typical home and corporate networks. This mitigates the attacks targeted towards IoT devices. Finally, the sub-network protects the main network by isolating vulnerable and infected IoT devices. Isolating the IoT devices and components in a separate network significantly increases the difficulty of an adversary to locate them, as pivoting across the two networks would require considerable effort from the attacker.

Figure 3.3 illustrates the proposed network and hub configuration. More specifically, it illustrates the smart home's main access point (top-left of the Figure) connected to two conventional laptops. A user interacts with the devices in the sub-network through PubNub and the hub's gateway. A second access point (bottom-centre of the Figure) is used to create a sub-network (illustrated by the green dashed line) which contains all the components of the proposed hub, i.e. the hub's gateway, the policy server, the database, and the connected IoT devices (e.g. Samsung Hub connected to a smart plug and a motion sensor and Philips Bridge Hub connected to a smart bulb).

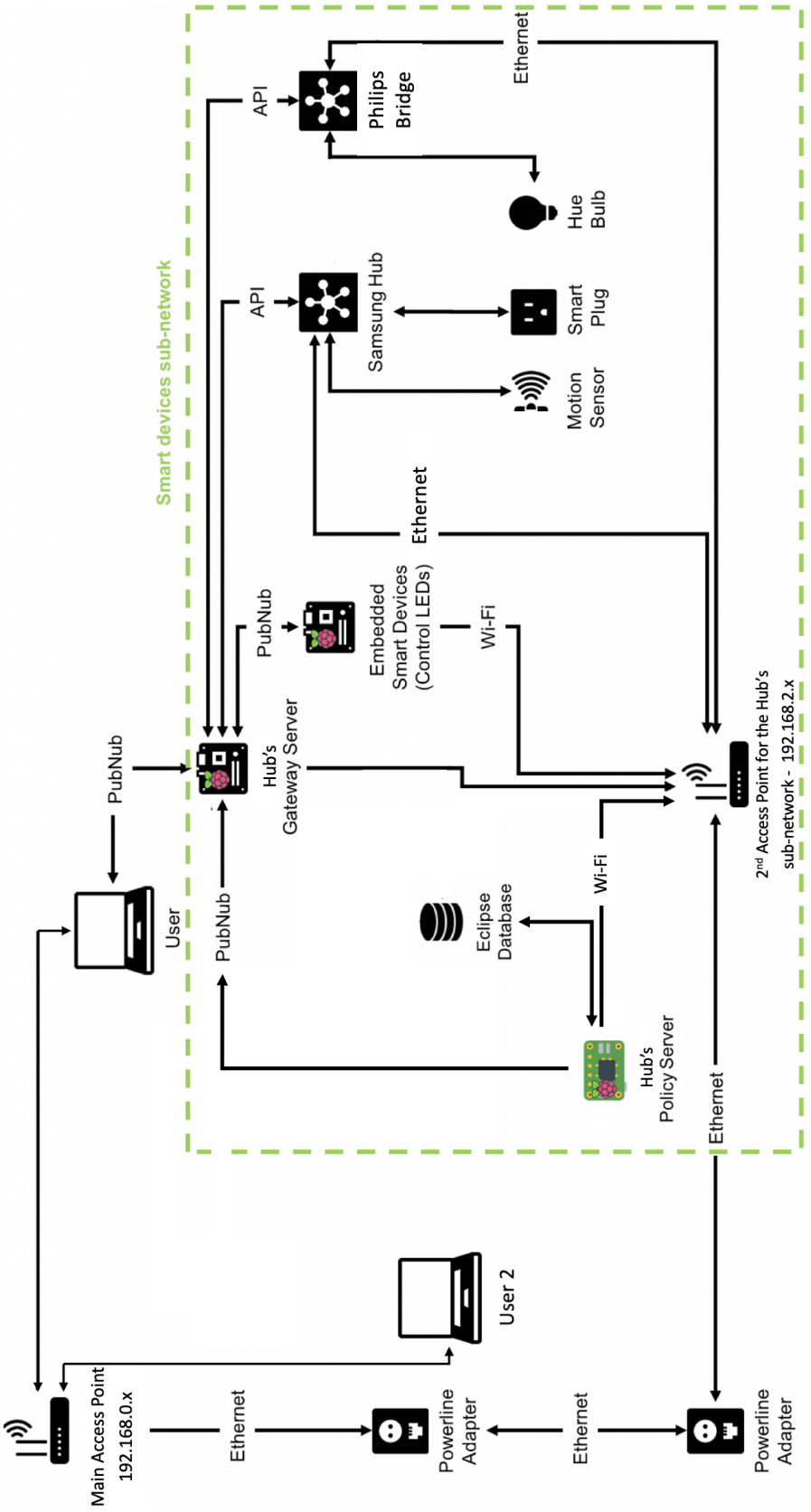

Figure 3.3: A network configuration which contains a specialised sub-network used to isolate the smart devices

It is also important to note that the inclusion of the gateway within the IoT network enhances heterogeneity and interoperability. Without the gateway, the incorporation of IoT devices within a new sub-network may cause the functionalities and interactions with the included IoT devices to be lost. For instance, the Philips Hue lamps must be on the same sub-network as their interacting devices. Changing lighting settings with a smartphone is therefore not possible from the main network. However, the inclusion of a gateway resolves this issue as it receives its Internet connectivity through Pubnub's cloud platform. Consequently, a user can access the gateway using a secure channel from the main network and can control the smart devices from anywhere using the Internet.

Table 3.3 summarises the key components of the proposed hub, their functionalities, and the security aspects they enhance.

Component	Functionality	Advantage
Gateway	Authentication	Enhanced Access Control
	User Requests	State Change, secure communication
	IoT tailored Canary Files	Early unauthorised access detection
	Module creation remotely	Post-deployment configuration
Policy Server	Accepts/Rejects requests	Enhanced request control mechanism
	Add/Edit/Delete policy remotely	Post-deployment configuration
	Create canary functions remotely	Post-deployment configuration
	Manage canary triggers	Alerting system administrators
Sub-network	Isolates/Camouflages devices and hub	Enhances overall security

Table 3.3: Hub components and their functionalities

3.5 Evaluation

In order to evaluate the security of the proposed hub architecture, several attacks were performed on a traditional IoT smart home testbed, both when the hub was and was not deployed. The robustness of the proposed hub architecture was evaluated using

a traditional penetration testing methodology. The following Sections describe the testbed configurations and the evaluation methodology.

3.5.1 Smart Home Testbed

Cisco's VNI [4] report that between 2017-2019, the average household in North America, Western Europe, and Central and Eastern Europe had on average 8, 5.4, and 2.5 smart devices respectively. A smart home is the symbiosis of different devices, i.e., sensors, connections, and applications that build a dynamic heterogeneous architecture with the aim of efficiently managing home devices, and providing advanced services to users [100]. To support the experiments described in this Chapter and to evaluate the proposed hub, an IoT testbed deployed in a home environment was configured in a star topology (where all the smart devices communicate to a central access point) as this is one of the most popular IoT network setups [87, 133, 191]. The testbed comprised of a range of IoT devices emulating a smart home environment, was designed and implemented. In more detail, the testbed consisted of existing and readily available commercial IoT devices of different types/applications: lighting, entertainment, safety cameras, sensors, and embedded devices [128]. Such devices included a TP-Link NC200 IP camera, the Samsung SmartHub with a connected motion and door sensors, an LG Television, the Philips Hue starter kit, and a Raspberry Pi 3 was used to emulate a user device which controlled Light Emitting Diode (LED) lights. The specific devices were chosen based on their popularity, availability, affordability and other relevant work (e.g. [185, 134, 57, 186]).The gateway and policy server were installed onto an additional Raspberry Pi 3 and a Raspberry Pi Zero Wireless system respectively. The rationale for this, was mostly the affordability and popularity of the Raspberry Pi devices. This testbed was deployed in a home environment.

3.5.2 Evaluation Methodology

In order to assess the effectiveness of the proposed prototype, a penetration testing methodology was followed. The penetration testing process involves analysing a target system in order to discover possible vulnerabilities that may be a result of incorrect system configurations, vulnerable hardware or software, and lack of security mechanisms. This analysis is typically carried out from the perspective of a potential attacker and can include the exploitation of vulnerabilities [142].

For the purposes of this evaluation, and to design an appropriate, valid, and non-biased methodology to evaluate the proposed hub, the core steps involved in a typical penetration testing approach were followed [5, 2]. These included 1) defining the scope of the penetration testing, 2) defining the attack vector and establishing the tools to be used, 3) information gathering, 4) vulnerability exploitation, and 5) reporting of the results. Due to the knowledge surrounding the network infrastructure schematics during the evaluation, the penetration testing strategy was considered as being white-box testing. More specifically, this process involved:

1. **Scope of penetration testing** - The scope of the penetration testing was defined. The aim was to target the IoT devices in the smart home network. The goal of the test was to evaluate if an adversary could deploy a range of network-based attacks that correspond to the attacker objectives as defined in Section 3.2.1.1. Subsequently, the attack vector included network scanning, passive eavesdropping, spoofing, DoS, and MITM.

2. **Attack vector and tools** - During this step, the appropriate tools that would be used for the assessment were chosen. These were selected based on industry standards, ease of use, available documentation, and cost. As such, the following tools were selected:

    - *Nmap* and *Zenmap* were used for Reconnaissance.

- *Airodump-ng* was used to identifying near proximity access points.
- *Deauth*, *aireplay-ng*, *Ping flood*, and *hping3* were used to deploy DoS attack.
- The *arpspoof* was used for spoofing.
- The *tcpdump* was used for passive network sniffing.
- The *Wi-Fi Pineapple* device/tool was used for the Evil Twins attack.

3. **Information gathering**: For the purposes of this experiment, it was assumed that an attacker has compromised the central access point and has access to the main network. Therefore, passive information gathering approaches and Open Source Intelligence Techniques were not used. The focus of this step was to use the relevant tools in order to collect information about the connected devices on the network including secondary access points, connected devices, MAC addresses, etc.

4. **Vulnerability exploitation**: Once vulnerabilities or vulnerable devices were identified, there was an attempt to exploit them or gain control of the device.

5. **Reporting**: Documenting and reporting the results of the tests, discussing future defence mechanisms, and reverting the network to its previous state in the event that information and settings were altered.

3.5.3 Attack Deployment

Subsequently, and based on the aforementioned methodology, the traditional IoT smart home network when both the hub was and was not deployed were evaluated. The penetration testing methodology which was followed to evaluate the hub is described below and is separated according to the attacker objectives considered previously:

Attacker Objective 1: Collect information about the connected devices on network (i.e. near proximity Access Points, what devices are connected to each, MAC

addresses, etc.

The first step, which is also the first attacker objective, involved discovering all the devices and possible sub-networks connected to the main network. This included identifying connected devices and their MAC addresses and Basic Service Set Identifiers (BSSIDs) of near proximity Access Points. To achieve this, the network was scanned using the airodump-ng tool. In the traditional IoT setup, the main/central Access Point was identified. By focusing on this central Access Point, all the connected devices, both traditional IT and IoT, became immediately visible.

On the other hand, when the proposed hub was deployed, the tool identified three Access Points in range. These included the main access point that the attacker was already connected to, an Access Point that did not advertise its SSID - which was the proposed hub's sub-network Access Point, and a third near proximity Access Point. At this point, and as the adversary only had access to the main network's Access Point, only the connected traditional IT devices such as such as laptops and desktops were revealed and subsequently accessible to them.

Following this, in order for the adversary to identify what other devices are within the home, they needed to iterate over the other two Access Points to investigate what devices are connected to each one. The Access Point with the hidden SSID was shown to have significantly a strongest signal as reported by the *PWR* attribute provided by airodump-ng. It is important to highlight that, by limiting the range of this Access Point, it could have been more challenging to detect it. Therefore, it was assumed that this Access Point was the one most relevant to the targeted smart home. As such, the next step involved attempting to identify the SSID of this Access Point. To achieve this, the aireplay-ng tool was used to forcefully disconnect a client from the target Access Point. The SSID was then broadcasted and captured when the client attempted to reconnect. Having retrieved the SSID of the hidden Access Point, its reachability was tested by sending an Internet Control Message Protocol (ICMP) echo packet. There was no response from the Access Point, as this was disabled in its configuration.

Finally, following the success of identifying the SSID and MAC address of the sub-networkâs router, the next step was to attempt to capture the WPA handshake in order to gain access to the sub-network. Subsequently, 4 deauthentication packets were sent to the Access Point. Once the clients were disconnected and reconnected, the handshake was captured and the hash of the password was taken. However, at this point brute-forcing the passwords was unsuccessful.

Attacker Objective 2: Gain Information about the network topology and devices

To gain further information about the network topology and the devices that are connected, Zenmap and Nmap were used. In the traditional IoT network, it was possible to reveal the topology and information about all the devices connected on the network, such as IP addresses, names, and open ports. It was noted that few of the IoT devices had between three and six ports open. On the contrary, when the proposed hub was deployed, it was only possible to gain information about the traditional IT devices that were connected on the main network. In both cases, these demonstrated to also have SSH and HTTP ports open. However, it was not possible to view and gather additional information about the smart devices on the sub-network.

Attacker Objective 3: Make the devices unavailable to the intended user

Given that all the devices on the traditional IoT network were exposed and their IP/MAC addresses were successfully identified in the previous steps, it was possible to perform Denial of Service (DoS) attacks to any smart device by using the deauth, Ping flood, and hping3 tools. On the contrary, when the proposed hub was deployed, the IoT devices were hidden/isolated within the sub-network, increasing the difficulty in identifying them and performing DoS attacks upon them.

DoS attacks often leverage ARP spoofing to link multiple IP addresses with a single targetâs MAC address. As a result, traffic that is intended for many different IP addresses will be redirected to the target‘s MAC address, overloading the target with traffic [10]. Given that in the traditional IoT smart home network it was possible to

successfully gain information regarding all of the devices, ARP spoofing was successfully performed in multiple smart devices, using the arpspoof tool. However, when the proposed hub was deployed, it was not possible to gain enough information regarding the connected devices, leading to an unsuccessful attack.

Attacker Objective 4: Intercept sensitive user data

Following the previous steps, the next step was to attempt to intercept sensitive data by passively sniffing the network. In the traditional IoT network, all IoT devices apart from one, employed the TLS protocol and, therefore, intercepting any traffic in plain text was not possible. When concerning the one device that did not employ TLS, it was possible to successfully intercept the credentials necessary to log in onto the devices's web interface. For the purposes of this attack, we assumed that the adversary had successfully managed to brute-force the password of the sub-network's Access Point. In this case passive sniffing was also performed in the secondary network. However, as the hub employs both the TLS and AES algorithm, it was not possible to intercept any traffic in plain text.

As part of intercepting sensitive user data an attempt to perform a Man-In-The-Middle attack in the form of Evil Twins against the sub-network's router was also attempted. However, this attack was unsuccessful due to the WPA configuration of the wireless Access Point.

Attacker Objective 5: Control the IoT devices

As part of the evaluation, the scenario where a seemingly legitimate user attempts to 'unusually' control the devices was also examined. In the traditional IoT smart home setting, there is no mechanism that aids in identifying unusual user behaviour (i.e. a user attempts to unlock the door in the middle of the night). As such, an adversary may be interacting with the smart devices undetected. However, when the proposed hub was deployed and due to the time frame defined policies and canary functions, such adversary is possible to be detected. In this case, when a user attempted to interact

with a device outside the specified time frame, the policy server rejected the request. Additionally, when a user attempted to run one of the canary functions, the system sent a notification to the admin and blacklisted the specific user.

A summary of these attacks as well as their results in both network typologies can be found in Table 3.4.

Attack	**Attack Method**	**Target**	**IoT**	**Proposed hub**
Device Detection	airodump-ng/Nmap	Network Identification	✓	✗
MAC Spoofing	airodump-ng	System Blackout	✓	✗
ARP Spoofing	arpspoof	System Blackout	✓	✗
Passive Sniffing	Wireshark	Data Leakage	✓	✗
MITM	Evils Twin	Data Leakage	✓	✗
DoS	deauth, ping flood, hping3	System Blackout	✓	✗

Table 3.4: Attacks used to evaluate the security of both the traditional network and the network once the hub is deployed. The ✓and ✗markers denote that the attack was successful and unsuccessful respectively.

Overall, the experiments demonstrate that the proposed hub implementation significantly enhances the overall security of the IoT ecosystem as it was able to defend against most of the attacks that are traditionally deployed on a conventional IoT network. However, it is important to highlight that further evaluation should be considered to deploy more complex and sophisticated attacks and to consider other attacker models. Different adversaries may be able to attack the same system by utilising different methods. As such, although the penetration testing methodology used was designed to consider popular attacks within such networks and considers adversaries that may cause significant damage, it is limited to the researcher's mindset and experience.

Additionally, it is important to note that the security of the proposed hub can be further enhanced or reduced depending on the exact configuration settings and hardware that is used to deploy this framework. For instance, if the signal range of the sub-network's Access Point was configured to be more more limited, it would have been more chal-

lenging to detect it. Another example is, if a weaker or more common password was used for the sub-network's AP, it would be possible for the attacker to brute-force it and gain access to that network. Equally, if a a cutting-edge secure router consisting with advanced security and privacy features is used, then the overall hub's security will dramatically increase. Therefore, an in-depth assessment of the most appropriate configurations and components used as part of the proposed hub is needed before deploying the hub in a real IoT network.

3.5.4 Performance Evaluation

To measure the performance of the hub, the execution time of the round-trip of the user's request to the gateway, policy server, and the response back was measured. As the hub has two variants of implementations, one that employs only the TLS protocol and another that employs the TLS in combination with AES algorithm, both cases were assessed. Due to the centralised and synchronous architecture of the hub, more devices on the sub-network may mean that there are more requests made to the gateway and policy server. As a result, the overhead may increase, resulting in possible delays or blackouts.

As mentioned in Section 3.4.3, the implemented add-on modules contain several functions that serve different operations for each IoT device. Therefore, as each function has its own execution time, they were measured separately. Each function was executed 100 times and the average time was calculated in seconds (s). For example, Figure 3.4 illustrates the execution time of the *lg_tv.py* module which operates an LG smart TV and contains the largest amount of functions in the network.

This Figure demonstrates that the performance of the proposed hub varies according to the configurations that the user has selected. Specifically, the delay is greater when more security mechanisms are employed, in this case, when TLS is used in combination with AES in the application layer. However, it is up to the user to decide which

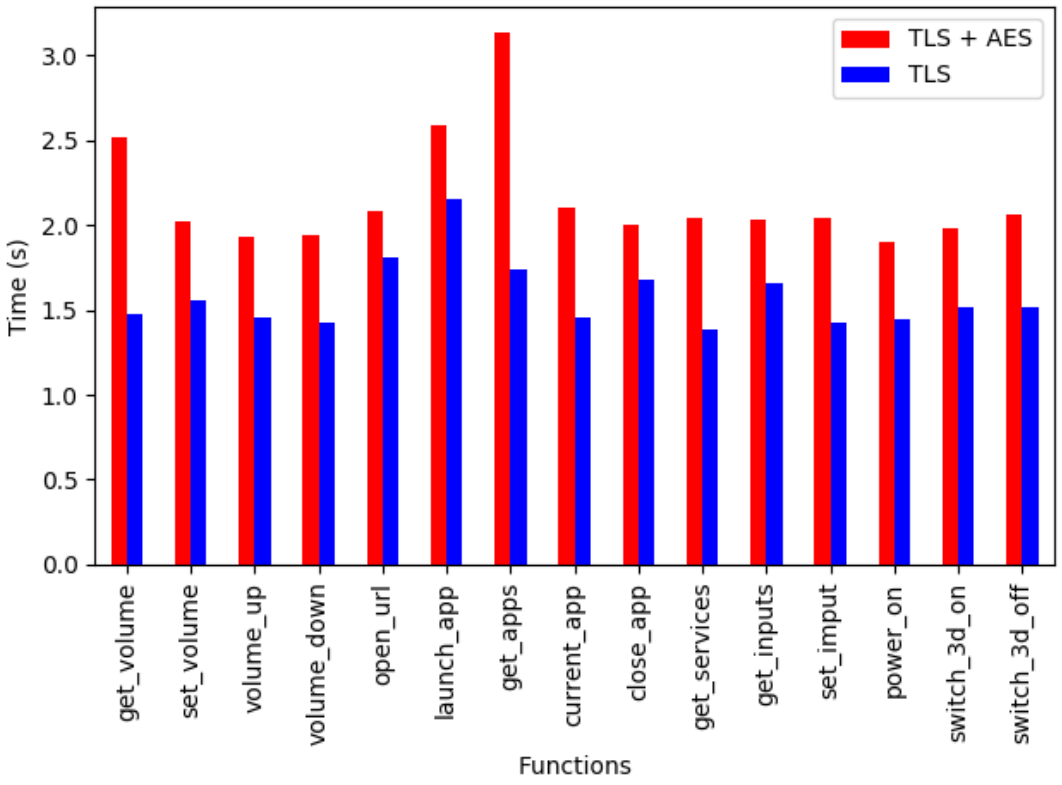

Figure 3.4: Execution times of the functions contained within the module that controls the LG smart TV.

of these mechanisms should be supported given the additional overhead. Nevertheless, the performance of the proposed hub could be improved. Currently when a request is being processed by the gateway, few other functions cannot operate at the same time and therefore there is a delay. Additionally, the performance of the system may also be affected in a larger network. Both of these issues may be addressed by making the framework operate asynchronously.

3.6 Limitations

One of the main limitations surrounding this work is that the hub relies on a third-party provider (i.e. PubNub) to support the core security functionalities within the

framework. If such third-parties halted their services, the operations on the framework would be affected. In addition, the framework always requires online access to manage local security, which by default may introduce new security risks as it maintains an always open socket connection to every device.

Secondly, although APIs provide an accessible and user-friendly interface to access, add, and control the smart devices, they are often subject to offering the control of limited device functionality and may have limits for the number of requests that they can receive. In addition, not all devices have an official API that can be used to control them, and therefore, third party APIs may be used. In this case, it is essential to assess the security and the legitimacy of these APIs, as they may also introduce new security and privacy risks to the hub if they are not secure (e.g. use request rate limiting, encrypt data via TLS, etc.).

The time-consuming and skill overhead associated with the need to manually develop module scripts and policies for each new device on the network is also considered as a limitation of this work. Aligning with this limitation is therefore the limited number of devices used to evaluate the performance of the hub.

Finally, the limited financial resources restricted the number of devices that could be purchased to be included in the testbed. As such, it is worth noting that the work presented in this Chapter was designed with a limited number of devices in mind, and if the Hub was to be expanded, the proposed solution which makes use of PubNub's free tier would need to be updated to use the paid version as it supports more devices. Moreover, the use of PasteBin's free tier to remotely create new modules may also introduce security and privacy risks if the developer includes sensitive information, such as passwords or usernames, as the free tier only allows public modules which may expose such information. The paid version of PasteBin may be used as it would significantly increase the security of the proposed solution due to the inclusion of features that are not included in the free tier, such as private bins and encryption capabilities.

3.7 Summary

The aim of this Chapter was to examine the feasibility of using a hub architecture to address the heterogeneity and enhance the security in IoT smart home environments. To tackle the limitations of existing hub implementations, which focus mainly on big data or scalability and may partially satisfy a maximum of two security mechanisms, this Chapter introduced the first design of a secure and heterogeneity-aware hub which fulfils five essential security properties that can defend against two attacker models that may threaten IoT smart home networks. These properties are: user authentication, access control, user/attacker behaviour monitoring, device cloaking, and confidentiality.

More specifically, the proposed hub implementation consists of three main components: a gateway, a policy server, and a sub-network. The main role of the gateway is to communicate with heterogeneous IoT devices regardless of their vendors via add-on modules, whilst allowing the users to access their devices over a secure communications channel. Simultaneously, the policy server maintains accountability for user access. Both these components are integrated within a sub-network which is configured to camouflage the IoT devices. This adds an extra layer of security as it provides segmentation between the IoT devices and traditional IT devices, whilst increasing the difficulty for an attacker to locate them. Furthermore, the hub provides secure communication channels between the users and the gateway. Finally, IoT tailored canary functions were included in the add-on modules to further enhance the overall security of the system by monitoring user behaviours.

To evaluate the effectiveness of the proposed hub architecture, a prototype of the hub was implemented and incorporated within a smart home IoT testbed containing a range of commercially available devices. A penetration testing methodology was applied to evaluate the robustness of the hub implementation and the results of when the hub was and was not deployed were compared. The results showed that the proposed hub significantly improved the security of the heterogeneous IoT network, as it was able to mitigate against most attacks which affect conventional smart home IoT networks.

Implementing secure IoT hubs is critical in order to defend heterogeneous IoT devices from cyber attacks. However, it is nowhere near a complete security model as adversaries will continuously develop new attack techniques to compromise such systems. As a result, to enhance the security of IoT networks even further, tools which focus on detecting malicious network behaviour must be present. In this case, Chapter 4 investigates the feasibility of employing supervised machine learning to support an IDS tailored for the IoT.

Chapter 4

A Supervised Intrusion Detection System for the IoT Environment

4.1 Introduction

IDS systems have emerged as successful attack detection and identification methods in IoT networks. Due to the proliferation of IoT devices, their heterogeneity, and the amount of data which is produced from such technologies, IDSs face a greater array of attacks to mitigate against. To overcome these limitations, machine learning techniques have been integrated to support IDSs in IoT networks.

Although there has been substantial research on machine learning based IDSs in IoT (e.g. [60, 91, 80, 144]), such approaches come with a range of limitations as identified in Table 2.2 in Chapter 2. These include the use of simulated network data and the evaluation of such systems on a small number of devices and cyber attacks. Additionally, existing literature lack focus on device profiling and classification. This is an important feature as it allows IoT assets to be identified and subsequently may aid in detect anomalies outside of the device's 'normal' behaviour. Furthermore, such systems do not investigate the automated identification of the type of attack which has occurred, which is critical for early attack response. According to the CSIR and NCSC [7], one of the main challenges faced by organisations is to identify the type of cyber attack which is occurring on the network without having to perform an in depth in-

vestigation. As a result, this may cause a delay in the attack detection and mitigation, which may subsequently lead to consequences ranging from data and financial loss to physical harm. Therefore, the ability to automatically detect the exact attack type may significantly reduce the human effort associated with responding to alerts, determining the severity of an attack, and launching effective countermeasures.

This Chapter introduces the initial classification experiments which form the basis of a novel supervised three layer IDS. A dataset containing both benign and malicious traffic was generated from a representative smart home IoT testbed. This dataset supported classification experiments involving a range of supervised classifiers. The experiments aided in answering the following research questions:

RQ2 *Can supervised machine learning approaches support the automatic detection of a range of cyber attacks based on network packet features collected from a range of IoT devices?*

RQ3 *Can supervised machine learning algorithms successfully classify different IoT devices based on network packet features?*

RQ4 *Given RQ2, can supervised machine learning algorithms further identify the main type of attack which has occurred?*

In answering these questions the following research contribution was made:

C2 An investigation into how supervised machine learning algorithms can be utilised to support a novel three layer IDS tailored towards the IoT.

The remainder of this Chapter is divided into the following main sections: Section 4.2 presents the implementation of a representative example of a smart home testbed. Section 4.3 discusses the generation and collection of IoT network traffic including the design, implementation, and deployment of cyber attacks. Sections 4.4, 4.5, and 4.6

discuss the preparation of the collected network traffic data for classification experiments. Sections 4.7 evaluates and reports the performance of the supervised classification experiments using the collected network data. Finally, Section 4.9 summarises the findings and highlights the research contributions of this Chapter. Parts of this Chapter have been published in [66] and [64].

4.2 IoT Testbed

Due to accessibility and location reasons, a new IoT testbed based on the same criteria listed in the previous Chapter was implemented. In particular, as defined in Chapter 3, a smart home is the symbiosis of different devices, i.e., sensors, connections, and applications that build a dynamic heterogeneous architecture with the aim of efficiently managing home devices, and providing advanced services to users [100]. To support the experiments described in this Chapter, and in contrast to existing studies revolving around IDSs in IoT, a testbed - housed at the University's campus - emulating a typical 'smart environment' was designed and implemented. For the same reasons discussed in Chapter 3, the devices were connected in a star topology. This configuration consisted of existing and readily available commercial IoT devices of different types/applications; lighting, entertainment, safety cameras, sensors, and appliance control [128]. Such devices included the Belkin NetCam camera, TP-Link NC200 Camera, TP-Link Smart Plug, Amazon Echo Dot, Lifx Lamp, and a Samsung Smart Things Hub and a British Gas Hive each connected to two sensors: a motion sensor and a window/door sensor. The specific devices were chosen based on their popularity, availability, affordability, and their use in other relevant and comparable work (e.g. [185, 134, 57, 186]). Table 4.1 summarises these devices, including their type and their connectivity protocol.

In addition to these devices, a laptop was connected to the IoT network in order to deploy various network based attacks. This designated machine was a Lenovo ThinkPad

IoT Device	Type	Protocol(s)	Firmware Version
Amazon Echo Dot (2nd Gen)	Multimedia	Wi-Fi	661664720
Belkin NetCam HD+ (F7D7606v1)	Multimedia	Wi-Fi	WW_2.00.7217.PVT
TP-Lik NC200	Multimedia	Wi-Fi	NC200_v2.1.7
Hive Hub	Sensors	Ethernet, ZigBee	1.0.0-7002-N105.3
Samsung Smart Things Hub (v3)	Sensors	Ethernet, BLE	0.24.22
TP-Link SmartPlug (HS100)	Sensors	Wi-Fi	1.0.8
Apple TV (5th Gen)	Multimedia	Wi-Fi	tvOS 10.1.1
Lifx Smart Lamp (3rd Gen)	Lamp	Wi-Fi, ZigBee	v2.9

Table 4.1: IoT devices included in the smart home testbed

configured to run the Kali Linux operating system [25]. At the same time, a network traffic capture tool was set to run on the access point in order to continually record the network traffic and save the log files in the syslog server. Figure 4.2 illustrates the network architecture of the IoT testbed.

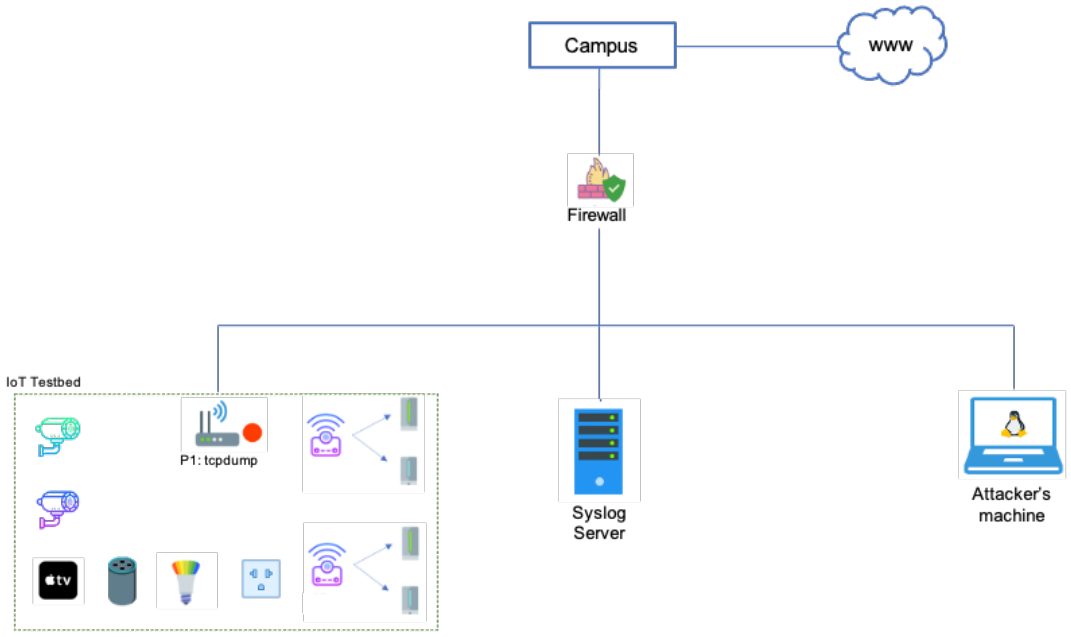

Figure 4.1: Overview of the architecture of the IoT testbed

4.3 Data Collection

In order to collect network traffic, the testbed was designed to capture all the packets on the network. All the inbound and outbound traffic from the smart devices were captured using the tcpdump [46] tool, which was running continually on the Access Point indicated with red circular marker in Figure 4.2. The data collection process was automated using Cron jobs and bash scripts. Network packets were continuously captured and saved to the Syslog Server in a Packet Capture (PCAP) format. Files were generated in one-minute intervals and were accessed remotely via SSH to connect to the Syslog Server. To conform to comparable research (e.g. [185]), three weeks worth of benign data was collected. For two weeks, attacks were systematically launched to generate and collect malicious data from the testbed. Overall, an estimated total of over six million benign and three million malicious packets were collected.

4.3.1 Benign Network Data

In order to generate a dataset consisting of benign network traffic, the IoT devices on the testbed were regularly interacted with. The interactions were decided based on the intended use of the device, its feasibility within the lab, and other relevant research.

- **Smart Things Hub**: The door and motion sensors were attached to the office door which was regularly used.
- **Belkin NetCam & TP-Lik NC200**: Both cameras were constantly on and positioned in the office.
- **TP-Link SmartPlug & Lifx Smart Lamp**: The smart plug was used to power the smart lamp and was scheduled to turn off in the morning and on in the afternoon. When on, the lamp was interacted with to change its colour [180].

- **Apple TV**: The Apple TV was used to occasionally stream videos from YouTube. However, due to the environment of the testbed and the use case of the Apple TV, it was impractical to interact with the device as much as the others in the testbed. Subsequently, the amount of network traffic generated from this device was significantly less than the other devices on the testbed, and thus was omitted from the classification experiments herein.

- **Hive Hub**: The hub was also connected to a door and motion sensor. The door sensors was attached to a desk draw and the motion sensor faced the main entrance of the office.

- **Amazon Echo Dot**: The Amazon Alexa was asked to complete some of its most popular types of interactions. Because the number of possible interactions with this device is not binary (e.g. on/off), a user-based study [135], that explored the the most popular interactions with it, was taken into consideration to inform our decisions. As such, these included: checking the weather, finding facts, listening to news, tell a joke, play music, set time, and check the time. However, due to the limited selection of devices, and the fact that the smart plug was scheduled to control the lamp using its own application, the Echo Dot was not connected to the other devices on the testbed. In this case, the Alexa was not asked to control their functionalities, such as turning the lamp on and off. This may be considered as a limitation within the study as this command is amongst Alexa's most popular use cases.

4.3.2 Malicious Network Data

Acquiring malicious network activity required the design and deployment of a range of cyber attacks. Although most IoT devices support a range of communication protocols, such as IEEE 802.15.4, Bluetooth, ZigBee, Z-Wave, LoRaWAN, and Cellular (GPRS/2G/3G/4G), the experiments in this thesis focus on the WiFi and Ethernet pro-

tocols. This is due to the following main two reasons: 1) the majority of the IoT devices in the testbed have Wi-Fi connectivity, and 2) such attacks can be launched remotely over the Internet making them particularly hazardous.

The details of the attacks that were deployed on the testbed are described in the following Sections. These attacks can be categorised into four main attack types: Reconnaissance, DoS, MITM/Spoofing, and multi-stage (iot-toolkit). They were selected for the experiments herein as they constitute some of the most popular attacks that may threaten an IoT ecosystem [65, 64, 66]. The specific configurations of these attacks can be found in Table A3 in the Appendix. To deploy the attacks the tools were chosen based on the same criteria as discussed in Chapter 5 which include industry standards, ease of use, available documentation, and cost.

In addition to the design, implementation, and launching of such attacks, logs were generated to record the date, time, type, and variation of the attack that was deployed. This was to support the labelling of malicious data discussed in Section 4.4. Additionally, the logs of all the outputs generated during the attacks, including the output returned by the tools used, were created for debugging purposes.

Reconnaissance Attack

After gaining access to a network, adversaries are able to perform reconnaissance in order to explore and learn more about the available targets. In the experiments herein, a range of attacks which fall under the reconnaissance attack type, and are widely used by both adversaries and cybersecurity professionals, were deployed using Nmap [137]. A detailed glossary reporting the exact scans deployed can be found in Table A3 in the Appendix.

Denial of Service (DoS) Attack

Adversaries launch DoS attacks in order to make machines or network resources unavailable to their intended users by temporarily or indefinitely disrupting the services of a host connected to the Internet. The most common type of DoS attack is Flooding,

during which a large number of communication requests are sent to the target. To deploy the DoS attacks described here, the hping3 tool [14] was used. A detailed glossary reporting the exact DoS attacks deployed can be found in Table A3 in the Appendix.

Man-In-The-Middle (MITM) Attack

Adversaries launch MITM attacks in order to intercept, replay, and possibly alter the communications between two legitimate parties. MITM attacks can be classified as being either active or passive. During a passive MITM attack, the adversary captures the transmitted data and passes it on to the original recipient. In an active attack, the adversary not only captures the data, but also modifies it. In the experiments herein, network traffic was collected for both passive and active MITM attacks.

In order to launch the passive MITM attack on the IoT testbed, both the Arpspoof and Ettercap tools were tested. Arpspoof is a command line tool which allows the attacker to intercept packets on a switched LAN by redirecting the traffic between two devices by forging ARP reply packets. This specific attack exploits the lack of authentication mechanism when updating the ARP cache. This results in new IP-to-MAC mappings which overwrite the previous values in the ARP cache. As it is distributed as part of the Kali Linux installation, the Arpspoof tool was mainly used for automating the launch of the MITM attack herein. To deploy the ARP poisoning attack, the following steps were taken:

1. The packet forwarding was configured on the attacker's laptop so that it would not drop the packets that were destined to other devices. The port forwarding makes the machine act like a router, as it will then receive every packet and forward it to another destination.

2. For this attack, the adversary is placing themselves between the Access Point and one of the IoT devices.

3. The machine running the Arpspoof was setup to spoof the ARP cache so that packets originating from the IoT devices to the router would pass through the

attacker's machine and vice versa.

The active MITM attack against the IoT devices herein involved injecting a spoofed ICMP packet crafted using Scapy [37], to force the host and router to make an ARP request. As a response, an illegitimate ARP replay is generated. By forging an ARP reply, an attacker may easily change the IP and MAC association contained in a host ARP cache which effectively routes the victim's traffic to the attacker's machine. Although this is not a traditional MITM attack, it plays an important role in data theft. ICMP is one of the most widely used protocols in the networking field. It operates in the network layer and is mainly used for diagnostic purposes. Unlike other protocols, any IP network device has the capability to send, receive, or process ICMP messages.

Multi-stage Attack using iot-toolkit

The iot-toolkit [40] was developed for SI6 Networks by Fernando Gont. The tool consists of a set of security assessment and trouble-shooting tools. More particularly, it can be used to assess the resilience of specific TP-Link smart devices by scanning for vulnerable devices, gather information about their specifications, and toggle them to change their operating status (e.g. ON/OFF). Given that such devices were included in the testbed, this tool was used to evaluate their security.

The toolkit includes a switch/toggle functionality which aims to switch the devices on and off. The implementation of this attack relies on the injection of JSON packets with altered payloads. Due to the physical nature of the plug, this type of attack has the capability of causing substantial damage to the device that it supports. Thus, the data generated from launching this attack was captured for the experiments herein.

4.3.2.1 Attack Scenarios

In order to generate a more representative and broad sample of malicious network activity, it was important to introduce some randomness to the deployment of the attacks discussed in Section 4.3. In this case, the automated launching of attacks at

random intervals were implemented using bash scripts. Randomisation was achieved by implementing a timer which launched the attacks at random, for a random period of time ranging between five seconds and 20 minutes. The idle time between the launch of each attack was also randomised using the same principle. The speed of the toggle attack, i.e. the amount of malicious packets sent to the device, launched using the iot-toolkit was also randomised.

As a result, in addition to launching of the aforementioned stand alone attacks, four automated attack scenarios were implemented and deployed. These scenarios were as follows:

1. Network Scanning

   In this scenario, either one type of quick scan or one type of quick scan followed by one type of intense scan was launched. The intense scan was launched with the probability of 0.5. The rationale for this scenario was that an attacker will often perform an attack with a quick scan to determine the available hosts and then decide whether to proceed to a more complex attack.

2. Network Scanning and DoS

   This scenario also incorporates a quick type of reconnaissance; however the attacker also performs one or more of the aforementioned DoS attacks on the target network. A maximum of six DoS attacks were performed one after another. The random duration of the attack, as well as the idle times between each, were randomised. The attacks were targeted towards each IoT device connected to the testbed at random using their known IP addresses.

3. Network Scanning and MITM

   This scenario also incorporates a quick type of reconnaissance followed by a MITM attack performed using ARP spoofing. This attack uses passive monitoring only or ICMP packet injection with a chosen probability of 0.5. The launching of these attacks, their idle times, as well as the number of injected packets were selected randomly.

4. Complete Attack with iot-toolkit

   In this scenario, the TP-Link smart plug on the testbed is targeted using the iot-toolkit. More specifically, the three attacks distributed as part of the iot-toolkit were launched. Again, the launching of these attacks along with the aforementioned consideration were selected randomly.

4.4 Data Labeling

Supervised machine learning relies on annotated training datasets, where the target value to which a predictor belongs is known. There exists various pre-annotated network traffic datasets; these are mostly collected from conventional IT systems.

Such datasets often contain packet-based or flow-based information about the network. While flow-based data contains meta information about network connections, packet-based data contains finer grain information about the transmitted packets. Flow-based network data may be captured using specialised software (e.g. NetFlow [86], IPFIX [85], sFlow [19], and OpenFlow [143]). Capturing network traffic on a packet-level can be achieved by using a network tap or by mirroring ports of the network devices.

The information contained within the network datasets dictate the features that can be derived to support machine learning experiments. In the context of packet-based data, features including packet headers, network and transport protocols such as TCP, UDP, ICMP, and IP, and metadata such as sequence number, TCP flags, and checksum values can be found. Whereas flow-based network data contains volume-based meta information such as transmitted bytes per second, packets per second, flow-size, and number of transmitted packets and bytes [85].

For supervised machine learning experiments, other than the features which represent the network data, such data points must also be associated with a target value. In the context of cyber attacks, predictors are often labelled as whether they are malicious or benign. There exists few studies, such as [78] and [192], which have focused on

classifying the type of attacks which have occurred on a network, such as DoS and DDoS.

There are two main approaches to labelling network traffic. Several datasets have been manually labelled using human annotators (e.g. [55]). Others have employed an automated approach by associating the deployment time of the attack with the packets (e.g. [170]).

In order to create a gold standard for the supervised classification experiments in this Chapter, the dataset collected in Section 4.3 was labelled according to following three points:

1. Given that the IP addresses of the devices on the testbed would change when they were under specific attacks (e.g. DoS), outbound packets from each device were labelled using their known MAC addresses. Packets were therefore labelled as one of the following devices: *Amazon Echo Dot, Belkin Net, TP-Lik NC200, Hive Hub, Samsung Smart Things Hub, TP-Link SmartPlug*, and *Lifx Smart Lamp*.

2. As attacks were launched systematically, *malicious* outbound packets from the dedicated attacking laptop were identified and labelled using its MAC address and the time frame of when the attacks were launched. In order to avoid the mis-labelling of packets, the services and applications (e.g. e-mail and web browsers) on the attacker's laptop were deactivated. Packets that were collected from the aforementioned devices outside of these time frames were labelled as *benign*.

3. Given the aforementioned point, malicious packets were labelled with the type of attack that was launched using the same approach. Malicious packets were therefore labelled as one of the following attack categories: *DoS, MITM, Scanning,* or *iot-toolkit*.

4.5 Sample Size Reduction and Class Balancing

Datasets containing a significantly large number of packets, such as those produced here, require high computational power and processing time when they are used to support machine learning experiments. Given the large volume of data collected over the three week period, and therefore the local computational power required to process such data, to determine a representative sample of benign activity, data collected from the first seven days of collection was selected. This included weekdays where there was substantial activity and a Saturday and Sunday where there was much less.

A possible limitation of the approach towards the sampling of this data is the possible change in activity at different time periods. For example, a device may incur a firmware update during the second week of data collection. In this case, packet sizes may be larger, and are subsequently not represented in the final dataset used to train and test the models. As a consequence, the final model may be unable to accurately decide whether such packets are malicious or benign.

As discussed in Section 4.3, in order to generate a representative dataset of malicious network activity, the configurations of the attacks were initialised and launched at random. In order to extract a sample of malicious data, the first 110,395 packets were selected.

Figures 4.2 - 4.4 show the distribution of packets across all the classes for each sampled dataset. For device type classification, the sample size contained a total of 994,341 packets. For attack detection, the sample size contained a total of 220,785 packets. For attack type classification, the sample size contained a total of 110,395 packets.

Weka [45], a popular suite of machine learning software, was used to support both the data sampling and classification experiments in this Chapter. An uneven balance of class labels across each experiment has the potential to negatively affect classification performance. Given this, the ‘spread sub-sampling’ and ‘class balancing’ filters available as part of Weka were applied to generate a random sub-sample of packets, and to

subsequently balance the distribution of classes within those samples.

For device type classification, the final sample size used for the experiments herein contained 70,000 packets, with 10,000 packets per device. For attack detection, the final sample size contained 80,000 packets, 40,000 and 40,000 of benign and malicious packets respectively. Finally, for attack type classification, the final sample size contained 40,000 packets, with 10,000 packets for each attack category.

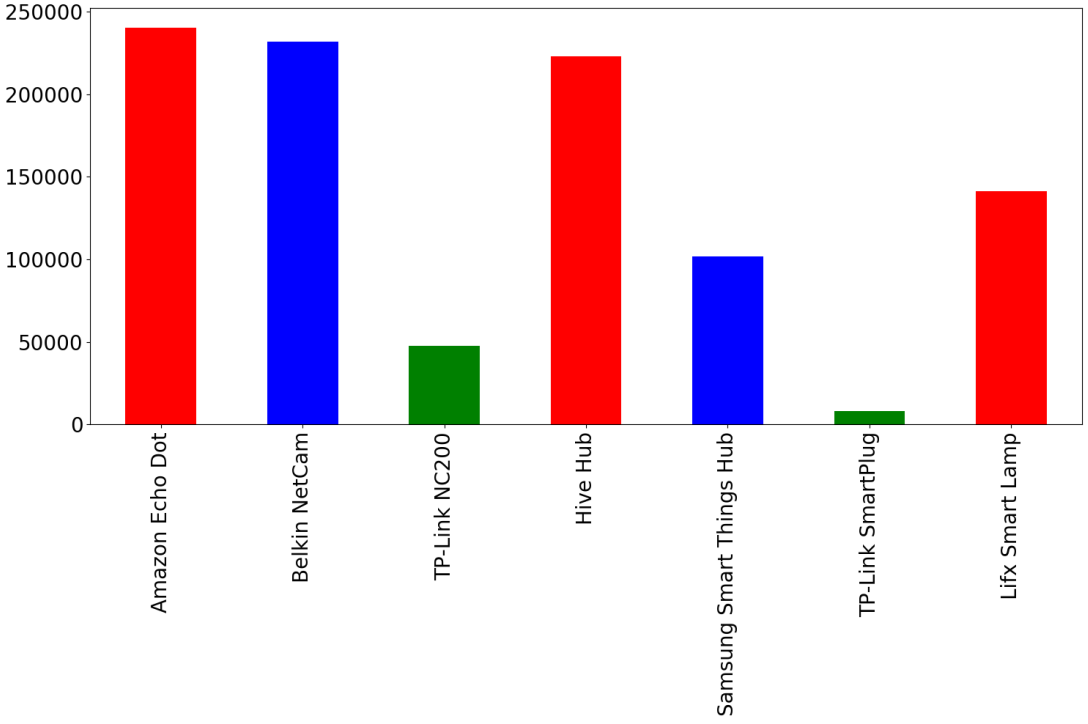

Figure 4.2: Distribution of packets across IoT devices

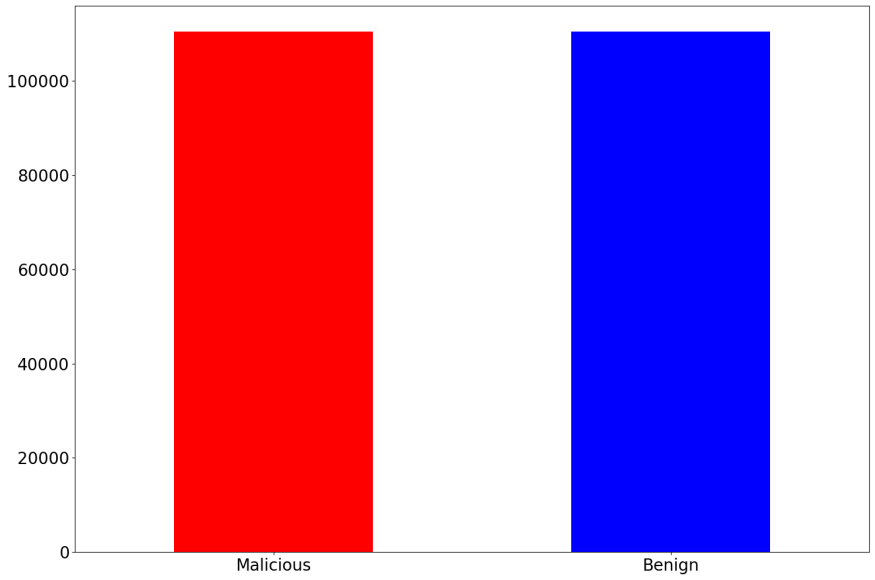

Figure 4.3: Distribution of packets across attack detection

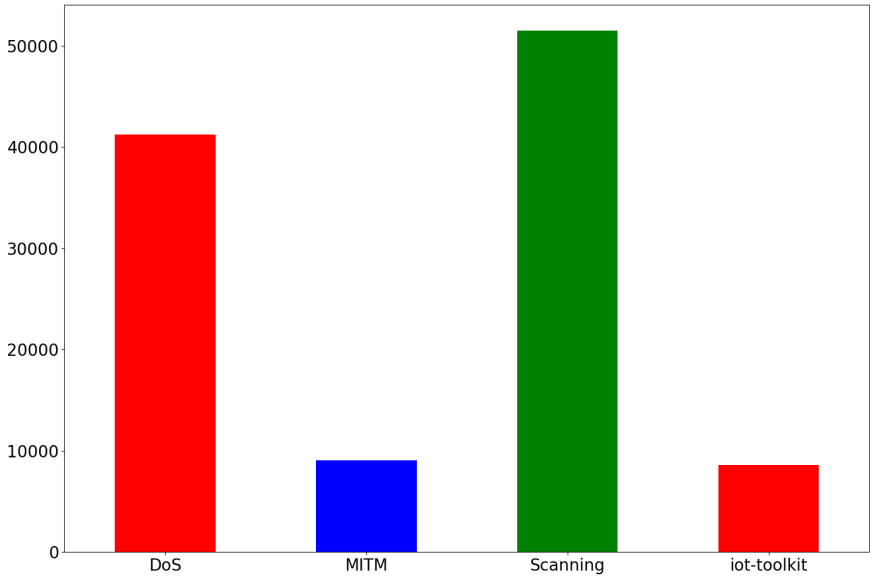

Figure 4.4: Distribution of packets across attack types

4.6 Feature Selection

The aim of the initial experiments herein is to investigate the feasibility of classifying device types and whether malicious behaviour has occurred by analysing individual network packets. The reasoning behind this approach is, as single packets are the smallest piece of network information, they are quicker to process, and subsequently improve the speed of identifying malicious activity. It is important to note that analysing individual packets may be effective when attacks/malicious behaviours do not consist of a sequence of actions to be deployed (e.g. reconnaissance or DoS). By deploying these attacks, adversaries aim to achieve maximum damage as soon as they gain access to the system [83]. However, this approach may be ineffective in detecting more sophisticated attacks where the adversary attempts to modify the system discretely over a period of time, by deploying attacks that involve sequencing of activities (e.g. installing a backdoor). This is because the supervised model does not consider the relationships between each packet or the sequencing of time.

Subsequently, to extract the necessary packet features, the raw PCAP files containing the network packets were first converted to a Packet Description Markup Language (PDML) [32] format. The reason for this conversion is because Wireshark [46], a popular open-source software for analysing packets, only allows access to 11 packet attributes (packet ID, time, source, destination, protocol, length, frame is ignored, info, delta time, frequency, and DSCP/CoS). Conversely, PDML conforms to the XML standard and contains details for Layers 2-7 of the OSI model [73]. As a result, it allows access to several packet attributes which can be used as features.

A network packet consists of a series of layers (Physical, Data Link, Network, Transport, and Application), with each layer being a child of the previous layer, built from the lowest layer up [38]. Each layer has its own header composing of a range of different fields providing protocol specific information (e.g. IP flags) and general packet information (e.g. packet length). For the classification experiments in this Chapter, all of the fields which compose each of the aforementioned layers were extracted resulting

in a total of 133 features. In addition to the aforementioned attributes, other fields were also included such as frame information [46].

From this feature set, features which represented identifying properties, such as source IP address, timestamp, packet ID, etc., were removed. This was to ensure that the model was not dependent on specific network configurations and that the features of the network behaviour were captured, rather than the network actors and devices. Finally, as the network traffic was encrypted, the payload information from the Application Layer was not considered as a feature. Subsequently, Table A1 in the Appendix reports the final feature set which contained 121 features. In order to be compatible with Weka, the PDML files were further converted to Attribute-Relation File Format (ARFF) format using a Python script.

At this stage, it is important to highlight that all features, apart from IP version, flags, and ports, were defined as having numerical values. The aforementioned features were defined as categorical as they can only have a specific range of values (e.g. flags are binary and can only be 0 or 1).

After projecting the packets onto their labels (discussed in Section 4.4), the packet data was represented as a feature vector. For all three experiments, each packet was represented as a double: (network packet features), where each value represents each of the 121 packet features, and (class label) which represents the label assigned to the packet. For example, a *malicious DoS* packet sent from the attacker's machine would be represented as the following two feature vectors, reporting the values for the first ten features (len, caplen, frame.encap_type, frame.offset_shift, frame.len, frame.marked, frame.ignored, eth.lg, eth.ig):

$$\underbrace{(64, 64, 1, 0, 50, 50, 0, 0, 0, 0, ...}_{\text{Network Packet Features}} \underbrace{Malicious)}_{\text{Class Label}}$$

$$\underbrace{(50, 50, 1, 0, 50, 50, 0, 0, 0, 0, ...}_{\text{Network Packet Features}} \underbrace{DoS)}_{\text{Class Label}}$$

4.7 Network Traffic Classification

Once the network traffic data was annotated, the dataset was split into training and testing data. A supervised classification algorithm was then used to predict the class label of unseen test instances, based upon their correspondence with the training data.

Several supervised machine learning classifiers exist. The "no free lunch" theorem suggests that there is not a universally best learning algorithm [205]; in other words, the choice of an appropriate classification algorithm should be based on its performance for the particular problem at hand, and the properties of data that characterise that problem.

As such, the suitability of eight supervised machine learning algorithms was investigated. The selection of these algorithms was based on their ability to support multi-class classification problems - this is essential due to the multi-layered nature of the proposed IDS, high-dimensional feature space - this is necessary due to the vast amount of packet features included within the dataset, and low computational complexity - this is critical in IoT environments as the IDS needs to be lightweight to be able to be deployed in low powered IoT devices. In addition, the classifiers included generative models that consider conditional dependencies in the dataset or assume conditional independence (e.g. Bayesian Network, Naive Bayes) and discriminative models that aim to maximise information gain or directly map data to their respective classes without modeling any underlying probability or structure of the data (e.g. J48 Decision Tree, Support Vector Machine). Finally, the algorithms were also chosen as they produce classification models that can be easily interpreted, allowing a better understanding of the classification

results.

- **Bayesian network** is a probabilistic model that represents a set of variables and their conditional dependencies via a Directed Acyclic Graph (DAG). These models are mostly used for taking an event that occurred and predicting the likelihood that any one of several possible known causes was the contributing factor.

- **Naïve Bayes** classifiers are simple probabilistic models which are based on Bayes' theorem. Such models assume conditional independence, i.e. each individual feature, independent of other features, is assumed to be an indication of the target value. Subsequently, the goal is to analyse the relationships between the features and the target value to estimate a conditional probability that correspond unseen data points to a target value.

- **Support Vector Machines (SVM)** are discriminative classifiers formally defined by a separating hyperplane. In other words, given labeled training data, the algorithm outputs an optimal hyperplane which categorizes unseen data points. In two dimensional space, this hyperplane is a line dividing the plane in two parts where each target value lays on either side.

- **C4.5 (J48) Decision Tree** is an algorithm used to generate a decision trees and is mostly used for classification problems. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

- **Random Forest** is an ensemble method as it generates many decision trees, referred to as a forest. Each tree is constructed using a different sample from the original data using a tree classification algorithm. Once the forest is formed, unseen data points are classified by traversing the nodes of each tree. Each tree provides a vote which indicated the tree's decision as to whether the unseen data points should be assigned to the target value of its nodes. Finally, a voting al-

gorithm processes all the votes to determine the final decision as to which target value the unseen data points is classified as.

- **Zero R** is considered to be one of the simplest classification method, which relies on the target and ignores all predictors. ZeroR classifier simply predicts the majority category (class). Although there is no predictability power in ZeroR, it is useful for determining a baseline performance as a benchmark for other classification methods.

- **One R** which is also known as "One Rule", is a simple, yet accurate, classification algorithm that generates one rule for each predictor in the data. It then selects the rule with the smallest total error as its "one rule". To create a rule for a predictor, a frequency table for each predictor against the target is constructed. It has been shown that OneR produces rules only slightly less accurate than state-of-the-art classification algorithms while producing rules that are simple for humans to interpret.

- **Simple Logistic** is a type of linear logistic regression model and is used to predict the categorical dependent variable with the help of independent variables.

4.7.1 Experimental Results

To explore how well classification algorithms can use network traffic to learn to distinguish between IoT device, whether outbound packets from such devices are malicious or benign, and given malicious packets, the type of attack which has occurred, the performance of a range of supervised classifiers were evaluated. Classifiers included generative models that consider conditional dependencies in the dataset or assume conditional independence (e.g. Bayesian Network, Naive Bayes), and discriminative models that aim to maximise information gain or directly maps data to their respective classes without modeling any underlying probability or structure of the data (e.g. J4.8 Decision Tree, SVM). Moreover, the aforementioned algorithms were also chosen as they

produce classifications models that can be easily interpreted, allowing a better understanding of the classification results. Given a test dataset, network traffic classification can be evaluated relative to the training dataset, producing four outputs:

- True Positives (TP) - packets are predicted as being malicious, when they are indeed malicious.
- True Negatives (TN) - packets are predicted as being benign, when they are indeed benign.
- False Positives (FP) - packets are predicted as being malicious, when in fact, they are benign.
- False Negatives (FN) - packets are predicted as being benign, when in fact, they are malicious.

These four counts constitute a confusion matrix shown in Table 4.2.

		Predicted	
		Benign	**Malicious**
Actual	**Benign**	True positive	False negative
	Malicious	False positive	True negative

Table 4.2: Confusion matrix for binary classification

There are several measures which can be used to evaluate the performance of a classifier. The goal is to maximise all measures, which range from 0 to 1. Therefore, higher values correspond to better classification performance. The most common measures are precision, recall, F-measure, and accuracy.

Precision (P) measures the proportion of malicious packet identification was correct, whereas recall (R) measures what proportion of malicious packets were identified cor-

rectly. Both metrics can be calculated using the equations in (4.1).

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN}, \quad F = 2 \cdot \frac{P \cdot R}{P + R} \tag{4.1}$$

Precision and recall are not often taken into account alone. The two measures may be used together in F-measure, which provides a single weighted metric to evaluate the overall classification performance. A specific version of the F-measure, F1-score (F), is measured by calculating the harmonic mean of precision and recall (Equation (4.2)).

$$F1 - score = 2 \times \frac{P \times R}{P + R} \tag{4.2}$$

Others use accuracy as a measure of performance. Accuracy (Equation (4.3)) measures the number of packets that were correctly classified. However, the problem of using accuracy to measure the effectiveness of a classifier is that if the classifier always predicts a particular class, a strategy that defeats the purpose of building a classifier, it will achieve high accuracy. This is also known as the accuracy paradox.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{4.3}$$

When classifying network traffic, evaluating the performance of the classifier concerns both measuring its effectiveness and efficiency. It is crucial that the classifier it not only effective in its predictions, but is also efficient in making such predictions. As a result, the classifier's ability to correctly predict unseen network traffic in an optimal time frame is evaluated.

Weka (version 3.8.3) was used to perform classification experiments using the default hyper-parameters. To comply with other comparable research which employ machine learning techniques to detect cyber attacks in traditional and IoT networks (e.g. [197, 173]), eight supervised classifiers were selected based on their ability to support multi-class classification and high-dimensional feature spaces.

Table 4.3 demonstrates the results following 10-fold cross-validation on the balanced gold standard created in Section 4.5, reporting classification models with the highest performance with each using the default parameters. The overall performance represents weighted-averaged results. Overall, Weka's implementation of the J4.8 Decision Tree method [163] without pruning achieved the best performance, resulting in an F1-score of 97.4%, 98.6%, and 98.6% for each experiment respectively.

	Device Classification			**Malicious or Benign**			**Attack Type**		
Classifier	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
Naive Bayes	81.4	79.8	78.6	94.4	93.8	93.7	89.7	86.3	85.6
Bayesian Network	97.3	97.1	97.1	95.6	95.6	95.6	98.9	98.9	98.9
J48	**97.6**	**97.4**	**97.4**	**99.9**	**99.9**	**99.9**	**98.6**	**98.6**	**98.6**
Zero R	0.0	14.3	0.0	0.0	50.0	0.0	0.0	25.0	0.0
One R	90.5	80.4	82.4	92.4	92.2	92.1	97.8	97.7	97.7
Simple Logistic	94.1	93.5	93.5	97.3	97.2	97.2	99.2	99.2	99.2
SVM	93.2	92.6	92.6	97.4	97.3	97.3	99.1	99.1	99.1
Random Forest	97.4	97.4	97.3	99.9	99.9	99.9	98.5	98.1	98.1

Table 4.3: Weighted average results following cross-validation

Confusion matrices given in Tables 4.4-4.6 show how classification outcomes are redistributed across the classes in each experiment. When classifying which device, the classifier demonstrated a high percentage of correct predictions, thus less often misclassifying devices. This may be explained by the fact that such devices are distinct, and therefore, so are their network behaviours. In this case, features may exist in some packets from one device, but are missing in packets from others. For example, the behaviour of the TP-Link NC200 is notably different in comparison to the behaviour of the TP-Link SmartPlug as their use cases are different. In this case, features within the TP-Link NC200 packets include the connection-less protocol, UDP; whereas the TP-link SmartPlug uses TCP. Confusion occurs in some instances where devices are incorrectly classified. For example, the Hive Hub is misclassified as the Belkin Net camera, and vice versa. These confusions may be explained by the fact that such

devices may have incurred similar network behaviour during data collection, such as when firmware updates were deployed.

Detecting whether network packets are malicious or benign and identifying the type of attack which has occurred also demonstrated very little confusion. For attack type classification, these results may be explained by the fact that the attacks that were performed during data collection were off-the-shelf attacks distributed as part of open source tools such as *hping, nmap, iot-toolkit* and resources which may not be considered as being sophisticated in their implementations. In this case, the features of malicious and benign packets may be distinct, and therefore few classification confusions have occurred. For instance, malicious packets may contain distinct flag values indicating an attack has occurred, which subsequently increases their discrimination from benign packets.

			Predicted						
			a	**b**	**c**	**d**	**e**	**f**	**g**
Actual	Amazon Echo Dot	**a**	9,990	3	5	2	0	0	0
	Belkin Net	**b**	1	9,838	154	5	0	0	2
	Hive Hub	**c**	5	1,485	8,495	3	1	2	9
	Samsung Smart Things Hub	**d**	0	0	0	9,981	0	1	18
	Lifx Smart Lamp	**e**	0	0	0	0	9,983	0	17
	TP-Link NC200	**f**	2	2	0	2	0	9,904	90
	TP-Link SmartPlug	**g**	1	0	0	0	0	0	9,999

Table 4.4: Confusion matrices when classifying IoT devices

			Predicted	
			a	**b**
Actual	Malicious	**a**	39,984	16
	Benign	**b**	20	39,980

Table 4.5: Confusion matrices when classifying network traffic as malicious or benign.

			Predicted			
			a	**b**	**c**	**d**
Actual	DoS	**a**	9,910	90	0	0
	MITM	**b**	0	10,000	0	0
	Scanning	**c**	0	160	9,849	0
	iot-toolkit	**d**	0	191	0	9,809

Table 4.6: Confusion matrices when classifying attack type

The classification results herein have demonstrated the efficiency of a supervised machine learning IDS tailored towards an IoT smart home. More specifically, the results show that the J48 classifier can classify the IoT devices on the network, whether a packet is malicious or not, and subsequently, what type of attack has occurred with high accuracy. In a real use case, this means that very few packets would be incorrectly identified as being benign or malicious, which means that the false-positive rates are very low. This is an important characteristic of an IDS, as it increases its reliability towards detecting malicious behaviour and subsequently launching the appropriate countermeasures at the most critical times.

However, it is important to note, based on the aforementioned analysis of the confusion matrices, with more similar device types on the network, it is possible that the overall accuracy of the model may decrease as the behaviours of each device will be more challenging to distinctly identify, subsequently increasing the model's confusion. Similarly, it is also important to highlight that the reason behind the high accuracy of the model when classifying benign and malicious packets and attack types is due to the unsophisticated nature of the attacks deployed on the testbed. In particular, as these attacks were deployed using off-the-shelf tools, their packets include distinct feature values which aid in discriminating between benign and malicious packets. If more sophisticated attacks are deployed, i.e. having packets with less distinct feature values, the accuracy of the model may decrease. The following Section provides an insight into the features that aid the J48 model to distinctly discriminate between benign and

malicious packets. Finally, Sections 4.7.3 and 4.7.4 further discuss how the model behaves when unseen data is presented, as well as the real-world use case of the IDS presented herein.

4.7.2 Feature Analysis

To gain a better understanding of the classifier's high performance, particularly in detecting the attack and its type, the *InfoGain Ratio Attribute Evaluation Filter* provided in Weka was used to identify which features best discriminate between the attack types. This filter evaluates the importance of the features in the training dataset by measuring their information gain with respect to the classes. In more detail, this filter measures how each feature contributes in decreasing the overall Entropy H(x) in the dataset - a measure to calculate the degree of disorder or uncertainty. Subsequently, an important feature is one which holds the most information and reduces the most the entropy [178]. Both the Entropy and InfoGain are defined respectively:

$$H(Class) = -\sum p(Class) \log p(Class)$$

$$I(Class, Attribute) = H(Class) - H(Class|Attribute)$$

Following the application of the InfoGain filter on the dataset, Table 4.7 illustrates the top features which best discriminate between the different attack types (see Table A1 in the Appendix for the descriptions of these features).

More specifically, among the top which affect the decision tree are: packet length, ip.ttl, tcp.time_delta, IP and TCP flags, ICMP fields, and TCP/UDP ports. In more detail, packet length can be an indicator of malicious behaviour, specifically when the packet is significantly larger or smaller than usual. The ip.ttl feature can also indicate the presence of an attack specifically when packets have unusual TTL values. In this

Attribute	Weight
len	1.48
tcp.time_delta	0.876
ip.flags	0.789
ip.ttl	0.781
ip.flags.df	0.625
tcp.srcport	0.567
tcp.seq	0.533
tcp.dstport	0.527
tcp.window_size	0.442
tcp.window_size_value	0.440
tcp.hdr_len	0.411
tcp.ack	0.410
tcp.flags.ack	0.395
tcp.flags.syn	0.388
ip.flags.mf	0.238
tcp.flags.push	0.149
udp.srcport	0.077
icmp.type	0.031
icmp.code	0.031
tcp.flags.cwr	0.015
tcp.flags.enc	0.015
tcp.flags.urg	0.015
tcp.flags.fin	0.012
tcp.flags.reset	0.011

Table 4.7: Feature importance ranking

case, the majority of the packets of the attacks deployed seemed to have very specific TTL values ranging between 60 and 64. Different attacks also have different tcp delta times. Higher rates of delta time is often an indication of a DoS attack. Moreover, the destination port of a packet is another useful feature for detecting activity such as port scanning which generally involves several probes to one or more ports. When present, ICMP code options such as fragment protection and packet protection can also indicate a DoS attack. Scanning methods and DoS (e.g. syn flood) mostly involve having modified TCP flags to invalid or improper settings. Additionally, specific TCP flag responses such as TCP SYN check and TCP SEQ check or ICMP packet transmission, can indicate a MITM attack. As a result, the various combinations of flags are crucial indicators of malicious activity. IP flags are indicators of IP fragmentation attacks and can take several forms such as UDP (an attack used against the IoT). These may be considered as being properties of a DoS attack as they make the device unavailable.

4.7.3 Unseen Validation Experiments

To ensure that the J48 classifier is not over-fitting, we performed additional experiments using the original datasets which resulted in no change in the classification performances:

- Classification using an *unpruned* decision tree.
- As the feature space is relatively large, all packet features may not be relevant. In this case, a known method that aids in avoiding over-fitting is to reduce the feature dimensionality to reduce noise and randomness from the data. Here, the feature space was reduced to the top 10 features as shown in Table 4.7 and the models were re-trained and tested.

To evaluate the performance of the trained models generated in Section 4.7.1 even further, the trained classifiers were applied to unseen datasets. Such datasets included packets that were represented in the original datasets discussed Section 4.5.

More specifically, for device type classification, the unseen dataset contained 40,000 packets in total, with 10,000 packets generated from each of the four IoT devices on the testbed. For classifying malicious packets, the unseen dataset contained a total of 4,200 packets, 2,100 malicious and 2,100 benign packets. Finally, for classifying the attack type, the unseen dataset contained 436 packets, 109 packets for each of the four attacks.

As shown in Table 4.8, the results demonstrate that for the device type classification and for identifying malicious packets, the accuracy of the classifiers decreased (from 98.8% to 96.2% and from 97.0% to 90.0% respectively). However, the performance of the classifier in distinguishing the types of attacks, did not change significantly (from 99.0% to 98.0%).

Given the minor decrease in classification performances, such results demonstrate that the model can generalise the data well and it is therefore assumed that it is not overfitting.

Device Classification			Malicious or Benign			Attack Type		
P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
96.2	96.8	96.9	90.0	89.9	88.8	98.0	99.0	99.0

Table 4.8: Weighted average results for each experiment on unseen validation data using the trained J48 models.

4.7.4 Scalability

The main use case for the IDS proposed in this Chapter is to be able to detect real time malicious behaviour in smart home IoT environments and identify the type of attack which has occurred. However, IoT in its own right is a large concept which includes a significant number of heterogeneous devices.

Larger networks with several other IoT devices are traditionally segmented into sub-

networks, each including a set of devices. In this case, when considering the scaling up of the proposed IDS, to detect malicious activity in environments with more devices, the IDS can be deployed on each sub-network. Having several instances of the IDS may ultimately lead to sharing network activity data between each sub-network. The data from one sub-network containing different devices to other sub-networks may be used to train the IDS to identify malicious activity in such devices when they are newly connected to the sub-network.

4.8 Limitations

One of the main limitations surrounding this work includes the over-head associated with accurately labelling packet data and the feature engineering required to represent such data to a machine learning model. In addition, the effectiveness of the system needs to be evaluated against more sophisticated attacks and within a much larger IoT network.

Furthermore, the model may face some limitations when new attack types and when new devices are added to the network. The current model may be able to detect malicious behaviour if such behaviours are similar to those that it previously seen. However, if a new type of attack has occurred, the model will attempt to classify such attack as one of the four it has been trained upon. This is a similar case when a new device is added. This increases the risk of false positives and may impact on the response to the attack. As supervised machine learning was used to support the experiments, the data from new and unseen attack categories and new devices must be collected and labeled. Due to the limited number of available devices and given that the focus was to demonstrate the feasibility of detecting malicious behaviour and device type using individual network packet data, this was not evaluated in this work. In this case, the model must be re-trained to reflect the current state of the network. To address this, commercially deployed supervised IDSs (e.g. [8]) are set to automatically re-train the

model periodically or when a new device is added to the network. To avoid interrupting the smooth functionality of the system, this can be scheduled to occur at non-peak times (e.g. middle of the night).

Finally, as with Chapter 3, the limited financial resources restricted the number of devices that could be purchased to be included in the testbed. As such, it is worth noting that the work presented in this Chapter was designed with such devices in mind.

4.9 Summary

The aim of this Chapter was to examine the reliability and effectiveness of supervised machine learning algorithms in classifying the IoT devices on a network, identifying whether packets are malicious or benign, and given malicious packets, the type of attack which has occurred. These experiments form the basis of the architecture of a novel supervised three layered IDS which can significantly enhance the security of IoT and has the potential to reduce the incident response rate. More specifically, the three layers of the proposed system address the limitations of existing work which lack focus on device profiling, focus on detecting a limited set of attacks, and most importantly, do not attempt to identify the exact type of attack that has occurred. Without this information, significant human effort is needed to respond to alerts, determine the severity of an attack, and launch countermeasures, which may have severe consequences ranging from data and financial loss to physical harm.

To support the classification experiments, real network data was collected from a smart home testbed consisting of eight commercially available IoT devices. In order to generate a representative sample of malicious network behaviour, a range of cyber attacks which were categorised into four main attack types were selected, configured, and targeted towards the IoT devices on the testbed. Once collected, the packets were processed individually and represented as feature vectors consisting of 121 features. In order to evaluate the performance of supervised classifiers, three classification experi-

ments were performed using eight classifiers that are comparative to other research. To explore how well supervised classifiers can learn to differentiate between IoT devices and network behaviour, 10-fold cross-validation experiments were performed. Overall, the J4.8 Decision Tree outperformed other classifiers, achieving an F1-score of 97.4%, 99.9%, and 98.6% respectively. This analysis indicates the potential of using supervised classifiers to support IDSs in IoT networks using features extracted from individual packets. Given the positive findings of the initial study, the next step is to implement this system in real time, so that it can be deployed in a real, much larger, heterogeneous IoT and Industrial IoT environment. This will allow the system to be further evaluated on more complex and more sophisticated attacks. Most importantly, it is critical to evaluate the robustness of supervised IDSs against AML attacks, which target the learning model itself and may allow adversaries to bypass such detectors. These attacks will be discussed in Chapter 5.

Chapter 5

Adversarial Attacks on Machine Learning Cybersecurity Defences in IoT

5.1 Introduction

As discussed and demonstrated in Chapter 4, supervised IDS systems have emerged as a successful attack detection and identification method in IoT networks. However, the implementation of such systems has introduced an additional attack vector; the trained models may also be subject to attacks. The act of deploying attacks towards machine learning based systems is known as Adversarial Machine Learning (AML). The aim of AML is to exploit the weaknesses of the pre-trained model. More specifically, by automatically introducing perturbations to the unseen data points, the model may cross a decision boundary and classify the data points as a different class to its true class. As a result, the model's effectiveness may be reduced, subsequently increasing the number of misclassifications.

Although there has been substantial research on machine learning based IDSs for IoT, there has been significantly less focus on AML. In the context of IoT, AML can be used to manipulate data and network traffic from the devices, causing malicious data to be classified as benign, consequently bypassing the machine learning based detector. This

has the potential to significantly delay attack detection, and given the success of the attack that otherwise could have been prevented, this may lead to personal information leakage, damaged hardware, and financial loss.

Given the impact that these attacks may have, the motivation behind this work is to evaluate the robustness of a supervised IDS against AML attacks. More specifically, this Chapter explores the application of AML in the context of IoT, as well as presenting an approach towards generating malicious adversarial packets. These adversarial samples are subsequently used to measure the robustness and effectiveness of the IDS presented in Chapter 4. This analysis answers the following research question:

RQ5 *Can AML techniques be used to evaluate the robustness of a supervised IDS for the IoT?*

In answering this question, the following contribution was made:

C3 A first approach towards investigating how AML techniques can be applied to evaluate the robustness of a supervised IDS tailored towards the IoT.

Given the positive results from the adversarial experiments which showed that the classifier's performance can be decreased by a maximum of 31.7 percentage points, this Chapter also investigates adversarial training. This method can be used to enhance the robustness of the model by introducing perturbed samples within the original training set and evaluating the retrained model. This answers the following research question:

RQ6 *Can adversarial training enhance the robustness of a supervised IDS for the IoT?*

In answering this question, the following research contribution was made:

C4 A first approach towards investigating how adversarial training can be used to enhance the robustness of supervised machine learning algorithms.

The remainder of this Chapter is divided into the following main sections: Section 5.2 discusses AML attack types and approaches, Section 5.3 presents an approach to generate malicious adversarial packets, Section 5.4 evaluates the performance of the model against AML samples, Section 5.5 evaluates the performance of the model following adversarial training, and Section 5.7 summarises the findings and highlights the research contributions of this Chapter.

5.2 Adversarial Machine Learning

To reiterate, the aim of AML is to automatically introduce perturbations to unseen data points in order to exploit the weaknesses of a pre-trained machine learning model. The following sections introduce the different types of AML attacks, as well as the methods used to automatically generate adversarial samples.

5.2.1 Adversarial Attack Types

Depending on the phase and aspect of the model that is being targeted, AML attacks can be described in terms of four primary vectors: [75, 115]:

- **Attack Influence**: An attack can have a causative influence if it aims to introduce vulnerabilities to be exploited at the classification phase by manipulating training data, or an exploratory influence if the attack aims to find and subsequently exploit vulnerabilities at classification phase. The attacker's capabilities might also be influenced by the presence of data manipulation constraints.

- **Security Violations** affect either the integrity of the model when the adversarial samples cause misclassifications, or when the high rate of misclassifications causes the model to become unusable.

- **Specificity** refers to targeted attacks, where the adversarial samples aim to target a specific target value, or indiscriminate attacks, where the samples do not target a specific target value.
- **Privacy** refers to attacks where the adversary's goal is to extract information from the classifier.

Papernot et al. [155] further categorise adversarial attacks based on:

- Their **complexity**. The consequences of such attacks can range from slightly reducing the confidence of a model's predictions to causing it to misclassify all unseen data points.
- The **knowledge** an adversary may have. A *white box* attack refers to when an attacker has useful knowledge related to the learning model, such as its architecture, the network traffic it reads, and the features used to support its training. An attack is considered as *gray box* when some aspects of the system are know (e.g. only the features used). Finally, an attack is considered as being a *black box* attack when an adversary has no information about the internal workings of the target model.

5.2.2 Adversarial Sample Generation Methods

There exists various methods by which adversarial samples can be generated. Such methods vary in complexity, the speed of their generation, and their performance. The methods discussed in Chapter 2 provide sophisticated approaches towards generating adversarial samples. However, such approaches may be too complex as an initial approach towards understanding the difference in behaviours between benign and malicious IoT packets, as well as generating adversarial network packets.

Given that the training dataset and its features are known, as well as the targeted classifier, the AML attack discussed herein focuses on grey box methods. In this case, two

relevant techniques towards automatically generating perturbed samples in a white box attack scenario include the Fast Gradient Sign Method (FGSM) and the Jacobian based Saliency Map (JSMA), presented by Goodfellow et al. [102] and Papernot et al. [155] respectively. Rigaki et al. [169] evaluate the aforementioned methods on the NSL-KDD dataset for traditional IT systems and demonstrate that such approaches can successfully generate adversarial samples that reduce the performance of the supervised classifier. Given these findings, these approaches form the basis of the methodology behind generating malicious adversarial samples in IoT presented herein.

Both FGSM and JSMA follow the methodology, that when adding small perturbations (δ) to the original sample (X), the resulting sample (X^*) can exhibit adversarial characteristics ($X^* = X + \delta$) [169] in that X^* is now classified differently by the targeted model. Moreover, both methods are usually applied by using a pre-trained Multilayer Perceptron (MLP) network as the underlying model for the adversarial sample generation.

The FGSM method aims to target each of the features of the input data by adding a specified amount of perturbation. The perturbation noise is computed by the gradient of the cost function J with respect to the input data. Let θ represent the model parameters, x are the inputs to the model, y are the labels associated with the input data, ϵ is a value which represents the extent of the noise to be applied, and $J(\theta,x,y)$ is the cost function used to train the targeted neural network.

$$x^* = x + \epsilon \operatorname{sign}\left(\nabla_x J(\theta, x, y)\right) \tag{1}$$

On the other hand, the JSMA method generates perturbations using saliency maps. A saliency map identifies which features of the input data are the most relevant to the model's decision being one class or another. These features, if altered, most likely affect the classification of the target values. More specifically, an initial percentage of features (θ) is chosen to be perturbed by an amount of noise (γ). Then, the model establishes whether the added noise has caused the targeted model to misclassify or

not. If the noise has not affected the model's performance, another set of features is selected and a new iteration occurs until an adversarial sample affects the algorithm's performance [155].

Given that the JSMA method may take a few iterations to generate adversarial samples, the FGSM is computationally faster [155]. However, as opposed to FGSM which alters each feature, JSMA is a more complex and elaborate approach which represents more realistic attacks as it progressively alters a small percentage of features at a time. This method often allows for more finer grained AML attacks, as adversaries are able to define both the percentage of features to perturb and the amount of perturbation to include when generating the adversarial samples.

Such methods have recently been successful in exploiting the weaknesses of machine learning based detection systems in ICS environments. Presenting a pre-trained model with AML samples generated from a dataset of industrial IoT device measurements demonstrated to significantly reduce its performance by 20 percentage points [67]. Given measurement data from IoT devices, such as recorded temperatures from a sensor, these approaches may also be applicable.

However, such approaches assume that all features can be equally perturbed by the same predefined constant. Thus, when considering network packet features, this may mean that perturbing these values outside of their valid ranges may jeopardise the validity of the packet, and subsequently the attack. For instance, a flag can only be 0 or 1 and the packet length must have a maximum integer value of 64 Kilobytes. Therefore, the aforementioned methods for generating adversarial samples may be ineffective when applied on network packets.

5.3 Generating Adversarial Samples

With the limitations of the approaches discussed in Section 5.2.2 in mind, this Chapter proposes a rule-based approach towards generating AML attack samples which aim to

target the IDS presented in Chapter 4.

The proposed approach is evaluated using malicious DoS packets against IoT devices. The rationale for choosing this type of attack is twofold; 1) DoS is one of the most catastrophic attacks against IoT devices [202, 84, 91], and 2) DoS attacks are not connection based; therefore, the packets are self contained and their features can be manipulated without voiding the attack. It is also important to highlight that it is significantly more challenging to manipulate the packet features in other attacks when tools such as Ettercap or iot-toolkit are used to deploy them. When an attack is launched from an application, packet level attributes are often inaccessible and it may not be possible to manipulate them.

Inspired by the JSMA and FGSM methods, the proposed approach aims to manipulate DoS attack packet features by considering:

1. **Feature Importance** - identifying the most important features that aid in attack detection.
2. **Practicality** - perturbing packet features that an adversary is able to modify using packet crafting tools such as Scapy [38].
3. **Validity** - given their practicalities, perturbing packet feature values between their valid ranges.

5.3.1 Feature Selection

In order to demonstrate how AML can be applied in the context of bypassing a supervised based IoT IDS, the dataset discussed in Section 4.3 was used. More specifically, all benign and DoS packets were extracted. The dataset consisted of 41,236 DoS and 110,390 benign data points. Given the uneven number of classes, the dataset was balanced to consist of 41,236 samples of both packet types. Subsequently, a random subset of approximately 60% of the dataset was selected for training, with 24,741 samples of

each class. The remaining 40% of the dataset was used for testing, with 16,495 samples of each class.

Table A1 in the Appendix reports the feature set which contained 121 features. For this analysis, it is essential to highlight that capture related features provided by the network sniffer (e.g. *tcpdump*) such as: caplen, frame.enacp_type, frame.offset_shift, frame.len, frame.cap_len, frame.marked, frame.ignored), excluding the tcp.delta time, were omitted from the feature space. The rationale behind this is that such features are not included in the original packet feature space and are generated by the network traffic tool. Therefore, these features cannot be manipulated by an attacker directly. However, the tcp.delta time feature can be indirectly manipulated by an attacker who may want to increase or delay the time between the sending of DoS packets.

Having removed the aforementioned attributes from the dataset, the *InfoGain Ratio Attribute Evaluation Filter* was used. This filter was also discussed in the previous Chapter 4 in Section 4.7.2. Following the application of the InfoGain filter on the dataset, Table 5.1 illustrates the top 15 features which best discriminate between benign and DoS packets with their respective information weight ranking (see Table A1 in the Appendix for a detailed description of the attributes). The remaining features resulted in a much lower importance score. Given that the full training dataset is accessible and the target model is known, the AML approach presented in this work is classified as a being a white-box attack.

Based on the feature importance results, as well as domain knowledge and practicality, the following features were chosen to be manipulated in order to generate adversarial packets; len, tcp.time_delta, ip.flags.df, ip.flags.mf, ip.ttl, tcp flags.urg, tcp flags.cwr, and tcp flags.enc. More specifically, adversaries may increase the network packet size by introducing padding to the packet header or they may reduce its size by fragmenting a single packet into more packets [122]. TCP delta time measures how much time has elapsed between the prior and current packet. Lower values of delta times correspond to higher rates of transmitted packets, which may indicate that a DoS attack is occur-

Attribute	Weight
len	0.873
tcp.time_delta	0.731
ip.flags.df	0.675
ip.flags.mf	0.298
ip.frag_offset	0.278
ip.ttl	0.178
tcp.seq	0.169
ip.proto	0.091
icmp.type	0.040
icmp.code	0.040
tcp.window_size	0.021
tcp.flags.urg	0.021
tcp.flags.cwr	0.021
tcp.len	0.021
tcp.flags.ecn	0.021

Table 5.1: Feature importance ranking

ring. Although this is a feature calculated by the network sniffer tool (i.e. *tcpdump*), it will be used in this work to explore how a lower rate flow of packets can affect the supervised classifier. The IP flags df can be set to indicate that a packet cannot be fragmented for transmission. The IP flags mf can be set to indicate that the packet contains more fragments. Time To Live (TTL) refers to the amount of time or number of hops a packet is set to exist inside a network before being discarded by a router. When crafting or manipulating packet features, the TTL value can be specified and set between 0 and 255. Finally, the TCP flags can also be set or unset; the URG flag is used to indicate to abort other segments so that the given segment is given priority, the CWR indicates that the host received a TCP segment with the ECE flag set and had responded in congestion control mechanism, and the ECN flag is used to echo back

the congestion indication. An adversary is able to craft packets where an invalid or unusual combination of flags can be set.

In order to better understand the structure of benign IoT packets, and subsequently, define the ranges in which these features can be perturbed, the distribution of the values of the benign packets for each of the given features that do not have binary value and excluding tcp.delta_time were analysed. These features are len and ip.ttl.

Figure 5.1 reports the distributions of the values for the len feature for both packet types. The minimum len value for benign packets was reported as 52, with the maximum value being 1,514. A significantly large number of len values for the benign packets (22,921) fall between the ranges of 54 and 194.

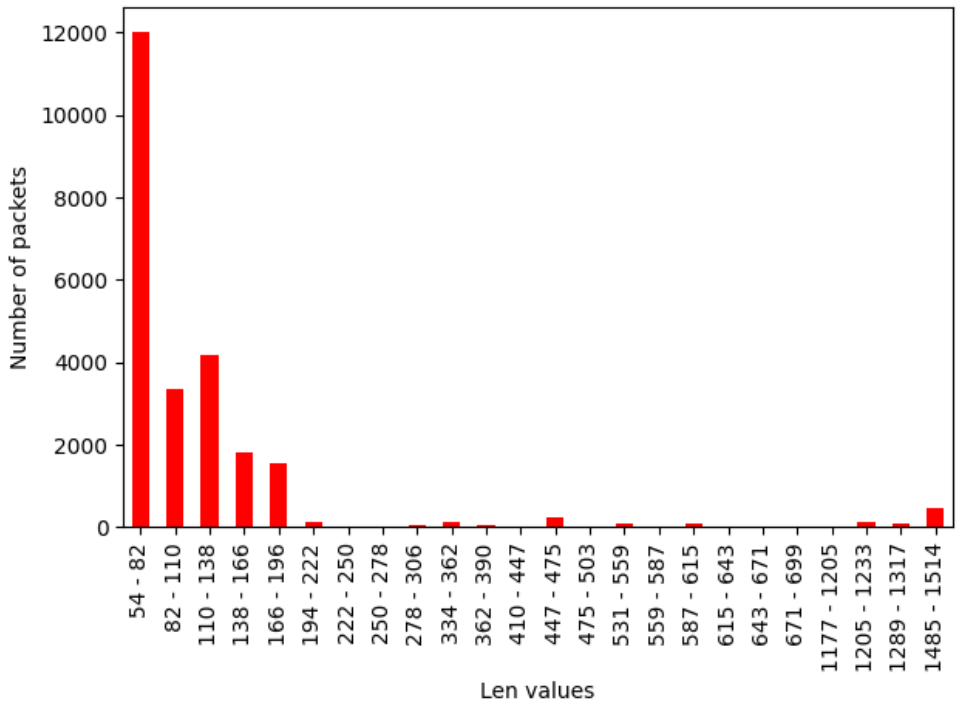

Figure 5.1: The distribution of len values for benign packets

Figure 5.2 reports the distributions of the values for the ip.ttl feature for both packet types. The minimum ttl value for benign packets was reported as 1, with the maximum value being 255. A significantly large number of ttl values for the benign packets

(20,633) fall between the ranges of 30 and 70.

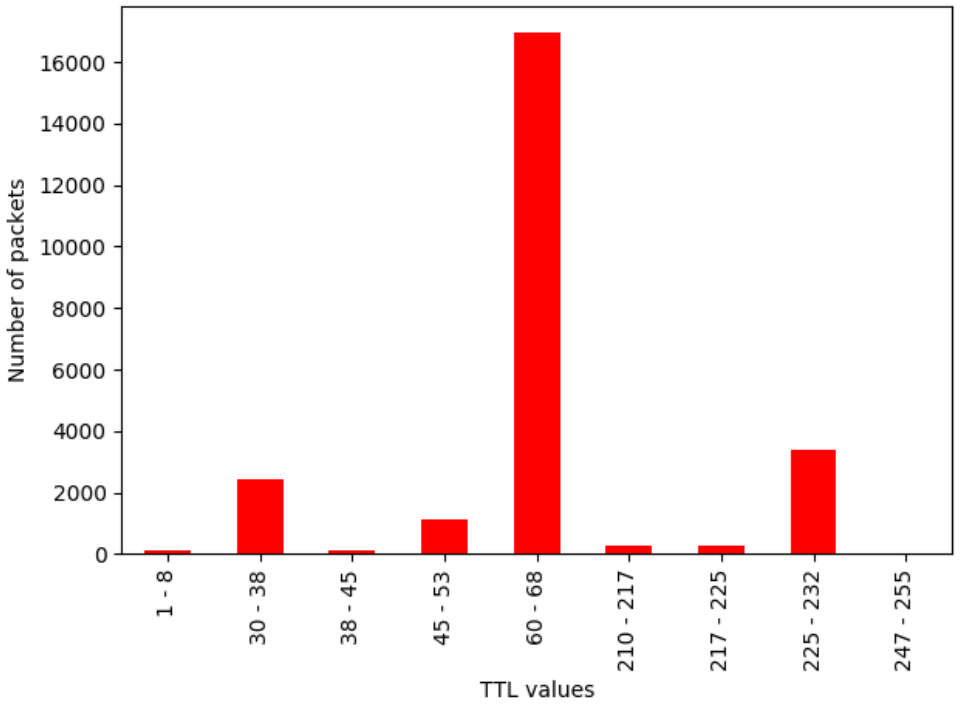

Figure 5.2: The distribution of ip.ttl values for benign packets

5.3.2 Generating Perturbed Samples

Based on the above observations, and to support the initial AML experiments within this Chapter, a range of feature combinations were perturbed, forming nine datasets. Firstly, to investigate how perturbing individual features may affect the classifier, adversarial samples were generated where only one of the features was modified at a time.

The aim of this approach is to mask adversarial samples to benign packets as closely as possible. Given the distributions in Section 5.3.1, the packet length and ip.ttl feature values were perturbed between the ranges of 54 and 194 and 30 and 70 respectively. For the flag features, the adversarial samples were generated by randomly setting the flag (1) or unsetting the flag (0). To explore whether a lower rate of packet flow can

affect the classifier's performance, the delta time feature was altered by increasing their values incrementally by five percent up to 50%. Finally, to explore whether DoS packets with lower delta time values are misclassified as being benign, adversarial samples were generated when all features, excluding delta time, were perturbed. Table 5.2 shows an example of how a malicious DoS packet may be perturbed given this approach.

Packet	len	ip.ttl	ip.flag.mf	ip.flag.df
Original Packet	50	64	0	0
Perturbed Packet	60	72	1	0

Table 5.2: An example of how malicious packet features are perturbed

It is worth highlighting, although such method of perturbation may be considered forceful, this level of perturbation is possible to be achieved by an adversary, specifically in IoT networks. This is due to the fact that the behaviour of the devices are not considered as being variable and do not have extreme deviations. As a result, an attacker can employ passive sniffing techniques to observe the activity of the IoT network, and thus craft and deploy AML attacks.

5.4 Evaluating the Model on Adversarial Samples

Given the findings in Chapter 4, the J48 classifier was first evaluated on the training dataset using 10-fold cross-validation and applied on the original testing dataset. The F1-scores achieved were 99.9% and 99.9% respectively.

To explore how an AML attack affects the performance of the trained classifier, adversarial samples were generated for all malicious DoS data points present in the testing data by individually perturbing each of the features discussed in Section 5.3.1, as well as perturbing all features, excluding delta time. The rationale behind this is to investigate the model's behaviour when the adversary only alters packet features and

not the rate of the attack. The original malicious packets were excluded from the testing data. The adversarial samples were subsequently included along with the benign testing data points and presented to the trained model.

Table 5.3 reports the average precision, recall, and F1-score following 20 iterations of generating perturbed samples for each feature-set.

Perturbed Features	**Precision**	**Recall**	**F1-Score**
all_features (excluding time_delta)	77.8	60.2	52.7
ip.flags.df	83.3	75.0	73.3
ip.flags.mf	99.9	99.9	99.9
tcp.flags.cwr	99.9	99.9	99.9
tcp.flags.ecn	99.9	99.9	99.9
tcp.flags.urg	99.9	99.9	99.9
ip.ttl	81.6	70.9	68.2
len	99.1	99.1	99.1
tcp.time_delta	99.9	99.9	99.9

Table 5.3: Classification performance on generated adversarial samples

When the tcp.flags.cwr, tcp.flags.ecn, tcp.flags.urg, tcp.time_delta, and len were perturbed individually, the model achieves the same F1-score of 99.9%. This demonstrates that perturbing these features individually has no impact on the model's performance. This may be explained by the fact that such features have a lower importance score (see Table 5.1) and also may rely on the values of other features in order to distinctly discriminate between both packet types.

When all features excluding tcp.time_delta were perturbed, the classification performance of the J48 model achieved an F1-score of 52.7%, a decrease of 47.2% percentage points in comparison to its performance when classifying the original testing data. This may be because the malicious DoS packets were significantly modified, therefore their similarity to the benign packets was increased. In addition, when perturbing the

ip.flags.df and ip.ttl features individually, the model's performance achieved an F1-score of 73.3% and 68.2% respectively, a decrease in 26.6 and 31.7 percentage points. This may be explained by the fact that the majority of the benign packets and a small number of DoS packets had the ip.flags.df set. Due to the tools used to deploy the DoS attacks, the default ttl value for these packets were set as 64. As a result, perturbing the ttl value between the aforementioned ranges significantly altered the distribution of the feature values. Subsequently, this demonstrated to impact the classifier's performance. Given these results, in order to achieve the most impact, an adversary would have to perturb all selected features excluding modifying the rate of the attack (tcp.time_delta) in order to successfully reduce the performance of the machine learning based IDS, and subsequently divert malicious data points.

The confusion matrices in Tables 5.5, 5.6, and 5.7 provide a better insight into the performance of the classifier across the experiments. In comparison to the original classification distributions in Table 5.4, the model demonstrates a significant increase in false positives when all features excluding tcp.time_delta and only ip.ttl are perturbed. That is, data points with an actual target value of DoS are misclassified as being benign. In addition to these results, when the ip.flags.df feature is perturbed, the model reports a higher false positive rate of almost 50%.

			Predicted	
			a	**b**
Actual	DoS	**a**	16,495	0
	Benign	**b**	1	16,494

Table 5.4: Confusion matrix for the original test set

The experiments herein have demonstrated the efficiency of AML against a supervised detector for IoT. More specifically, following the analysis of the malicious DoS packet values against the benign, it was identified that simultaneously perturbing nine packet features had a significant effect on the model's performance, leading to malicious DoS packets being misclassified as being benign.

In a real use case, this means, if an adversary has successfully gained access to the IoT smart home network, it is possible for them to monitor the network activity of the home over a period of time, even if the traffic is encrypted [51]. This allows the adversary to observe the network's normal behaviour and subsequently craft and adjust their attacks to map very closely to the benign activity. As such, if a supervised IDS is deployed on the network, an adversarial attack as described can be deployed successfully, where malicious packets bypass the detector. This may have a significant effect on the homeowner, as the attacker's behaviour on the network may go undetected. This also demonstrates, although supervised-based detectors may show high efficiency, their robustness against adversarial attacks must be extensively evaluated as they can significantly reduce their reliability and may cause severe consequences. To address this, the following Section investigates the effectiveness of adversarial training in defending against such attacks.

			Predicted	
			a	**b**
Actual	DoS	**a**	3,345	13,150
	Benign	**b**	1	16,494

Table 5.5: Confusion matrix after perturbing all select features excluding tcp.time_delta.

			Predicted	
			a	**b**
Actual	DoS	**a**	6,901	9,594
	Benign	**b**	1	16,494

Table 5.6: Confusion matrix after perturbing ip.ttl

			Predicted	
			a	**b**
Actual	DoS	**a**	8,245	8,250
	Benign	**b**	1	16,494

Table 5.7: Confusion matrix after perturbing ip.flags.df

5.5 Defending against Adversarial Machine Learning

A few methods towards defending against AML attacks have been proposed in the literature. Two of the most popular techniques include adversarial training and adversarial sample detection. The former has been explored in the field of visual computing, where Goodfellow et al. [103] demonstrated that re-training the neural network on a dataset containing both the original and adversarial samples significantly improves its efficiency against adversarial samples. The latter technique involves the implementation of mechanisms that are capable of detecting the presence of such samples using direct classification, neural network uncertainty, or input processing [222]. However, these detection mechanisms have been found to be weak in defending again AML [70, 222].

Subsequently, given the positive findings of how AML affects supervised detectors, the robustness of the IDS presented in Chapter 4 against AML is further evaluated using adversarial training. In this case, a random sample of 10% of the adversarial data points (1,650 packets) when all features excluding tcp.time_delta were perturbed, which decreased the model's performance the most (iteration 1, F1-Score = 52.6%), were included in the original training dataset. The experiments described in Section 5.4 were repeated by retraining the model on the newly generated training data and applying the model on the unseen adversarial samples generated in the remaining iterations (iterations 2-20). Table 5.8 reports the average precision, recall, and F1-score following 20 iterations which included newly selected random perturbed samples in the training set.

The results demonstrate that including adversarial samples in the training data in-

Perturbed Features	Precision	Recall	F1-Score
all_features (except time_delta)	99.5	99.5	99.5
ip.flags.df	99.7	99.7	99.7
ip.flags.mf	99.9	99.9	99.9
tcp.flags.cwr	99.9	99.9	99.9
tcp.flags.ecn	99.9	99.9	99.9
tcp.flags.urg	99.9	99.9	99.9
ip.ttl	99.8	99.8	99.8
len	99.1	99.1	99.1
tcp.time_delta	99.9	99.9	99.9

Table 5.8: Classification performance on generated adversarial samples

creased the performances of the J48 model. For each combination of features, the classification performance achieved an F1-score of over 90%, an increase of over 25% percentage points in comparison to the classification performances reported in Table 5.8. These results may be intuitive, as during adversarial training, the classifier is trained upon both types of packets with a range of feature values.

This approach has demonstrated the efficiency of applying adversarial training to enhance the robustness of a supervised machine learning based detector against AML. More specifically, following the inclusion of adversarial samples into the training data and subsequently re-training the J48 model, the resilience of the detector against such attacks was increased. In a real use case, this means that perturbed malicious packets may be detected with high accuracy. Although this approach demonstrated to be efficient, its applicability in a real use case would be challenging. This is because it requires the anticipation of every possible combination of valid malicious packets so that they can be included within the training data. As a result, increasing the volume of training data to attempt to capture all possible adversarial attacks may impact the computational time taken to train the model.

5.6 Limitations

One of the main limitations surrounding this work is the crude approach towards the perturbation of the chosen features. That is, here, we assume that the adversary has full knowledge of the dataset and the trained model. Therefore, following the analysis of the benign packets, the adversary can identify the ranges in which the feature values fall into and subsequently map the malicious packets to mimic the behaviours of the benign. The manual overhead associated with this approach may be addressed by utilising a more sophisticated method of generating perturbed packets (e.g. Iterative Gradient Sign, Carlini Wagner, Generative Adversarial Networks).

The work presented in this Chapter focuses on perturbing malicious DoS packets to bypass the detector. However, this is only the tip of the iceberg; therefore, the applicability of such an approach and other AML approaches of bypassing machine learning-based IDSs need to be further investigated for other attack types.

Lastly, with regards to adversarial training, the results demonstrated the efficiency of such an approach to increase the robustness of the IDS. However, it is important to highlight that this method may not always be sufficient as it is difficult to anticipate all possible types of AML attacks against a given system. Therefore, there is a need to investigate other possible defence mechanisms.

5.7 Summary

Due to their effectiveness and flexibility, machine learning based IDSs are now recognised as fundamental tools for detecting cyber attacks in IoT systems. Nevertheless, such systems are vulnerable to attacks that may severely undermine or mislead their capabilities, commonly known as AML. Such attacks may have serious consequences in IoT infrastructures, as adversaries could potentially modify malicious data points in order to bypass the IDSs, causing delayed attack detection, sensitive information leak-

age, and extensive damages. Thus, it is evident that understanding the applicability of these attacks in IoT systems is necessary in order to develop more robust machine learning based IDSs.

This Chapter explored how adversarial attacks can be used to target supervised models by generating adversarial samples, presenting such samples to a trained model, and understanding their classification behaviours. To support the experiments presented herein, an IoT network dataset containing benign and DoS packets were used to train and test a J48 Decision Tree, the best performing classifier for detecting malicious and benign packets in Chapter 4. The experiments focused on DoS attack packets as it is one of the most severe attacks against IoT devices, it is feasible to deploy by crafting custom packets, and finally, due to the nature of DoS, an adversary can manipulate packet features without voiding the attack. To identify which features can be manipulated, the importance of the features for discriminating against both packet types was measured. Based on these results, the top ranked features were selected for perturbation to generate adversarial packets. Firstly, to investigate how individual features may affect the classifier, adversarial samples were generated where only one of the features at a time was modified. In addition, an adversarial dataset was generated where all features excluding time delta were perturbed. Such samples were evaluated against the trained model. The results demonstrate that perturbing all features, excluding time delta, achieved the highest impact as the classification performance decreased by 47.2 percentage points.

Given these positive findings, the analysis also included the exploration of how adversarial samples can enhance the robustness of supervised models using adversarial training. A random sample of 10% of the generated adversarial data points when all the features were perturbed was included in the original training dataset. The model was retrained and applied to all unseen adversarial samples, excluding the adversarial samples included in the training set. Overall, the classification performance significantly increased when adversarial samples were present in the model's training.

Chapter 6

Research Contributions

The research presented in this thesis forms four contributions that can be grouped into two primary areas: (i) detecting and (ii) defending against cyber attacks in a smart home environment. Together, these two areas support the holistic focus of this thesis: to improve upon the limitations found in various approaches towards enhancing the robustness and completeness of IoT smart home security models.

6.1 Defending IoT Smart Home Environments

Current insufficient security measures employed to defend smart devices make IoT one of the weakest links to breaking into a secure infrastructure, and therefore an attractive target to attackers [64]. As illustrated in Figure 3.1 in Chapter 3 , in a traditional network configuration within a smart home, a number of heterogeneous IoT devices are all connected to a centralised point with little to no means of security. One of the key challenges in increasing the cyber-defence in these networks is the heterogeneity of IoT devices, and subsequently, the application of hubs as a unified security mechanism in the ecosystem. Therefore, this thesis proposed the first design and prototype implementation of a unified, heterogeneity-aware hub for the IoT environment (Chapter 3), implemented and validated within the context of a smart home network. Revisiting the previous situation shown in Figure 3.3 of Chapter 3, Figure 6.1 illustrates the novel hub design which now extends the robustness of previous heterogeneity-aware frameworks

[147, 176, 101, 167, 58, 107]. These are often vendor-specific and therefore expose the security vulnerabilities of other devices. The novel work produced in Chapter 3 enabled new dynamically loadable add-on modules to communicate with a range of diverse IoT devices at once to provide the following key security mechanisms: secure authentication, access control using a policy server, limiting the exposure of local IoT devices through *cloaking*, employs cryptographic protocols to ensure confidentiality, and offers a *canary-function* capability to monitor user and attack behaviours. The effectiveness of the hub was successfully evaluated by deploying cyber attacks that complied with attacker objectives that may threaten a traditional smart home network topology. This formed the first main contribution of this thesis, which supports the new knowledge that a range of heterogeneous IoT devices from different vendors may all be defended successfully against malicious behaviour under the proposed hub's infrastructure:

C1 A design and prototype implementation of a novel secure and heterogeneity-aware hub for the IoT, which enhances the security of an IoT smart home network.

6.2 Detecting Malicious Behaviour in IoT Smart Home Environments

In addition to defending smart home environments, the robustness of the hub and the completeness of the overall security model can be further supported by including a machine learning based IDS on the network; with evidence that supervised models can be utilised to support the automatic detection of malicious behaviour and which device may be compromised. The proposed IDS in Chapter 4 extends existing machine learning approaches [190, 91, 193, 166, 180, 158, 152, 141, 145, 144], which are often designed to detect one type of attack at a time and do not attempt to identify the exact

type of attack that has occurred. However, attack type information can aid in reducing the significant human effort needed to respond to alerts, determining the severity of an attack, and launch countermeasures. To enhance the previous contribution, Figure 6.1 also shows how the approach improves existing IoT architectures by adding a novel method which analyses network packets and enables the identification of the exact attack type that has occurred in the system. Previously this was not feasible, therefore, this enhanced the ability to understand how to better respond to evolving IoT attacks tailored to the specific attack type. This leads to the following additional contribution:

C2 An initial exploration towards utilising supervised machine learning algorithms to support a novel three layer IDS tailored towards IoT.

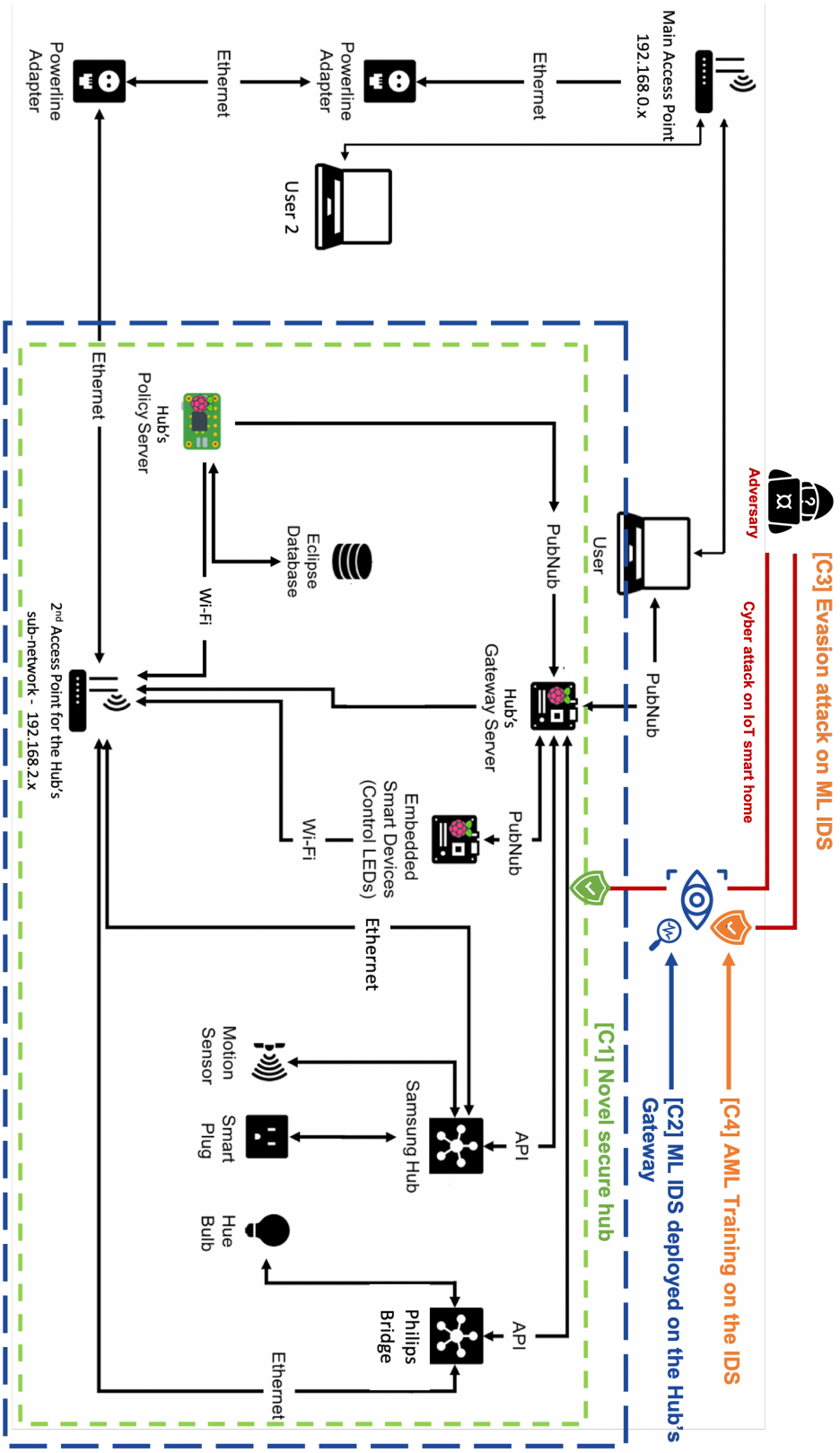

Figure 6.1: A smart home IoT network following the deployment of the hub and the proposed IDS system

While the previous contribution has shown where and how an additional level of IoT security can be enabled by adding a novel machine learning based approach towards classifying the exact type of attack - this has also introduced a new vulnerability - namely, the ability to evade the machine learning detection method (see Figure 6.1). This can occur when attackers manipulate their attack to confuse the output of trained models, also known as Adversarial Machine Learning (AML). The research in Chapter 5 investigated how such attacks can be deployed on an IoT network and found that the robustness of the proposed IDS can indeed be compromised using AML; with evidence that the predictive performance of the models significantly decreased when such attacks were deployed. This leads to the following additional contribution:

C3 An investigation into how AML techniques can be applied to evaluate the robustness of a supervised IDS tailored towards the IoT.

The understanding of the potential impact of AML attacks against supervised IDSs for IoT and the lack of research in this context is the core motivation in investigating how to defend such systems against AML attacks. Therefore, to enhance the security of the proposed IDS, and as shown in Figure 6.1, Chapter 5 also includes the exploration of how adversarial training can enhance the robustness of the supervised IDS; with evidence that its performance significantly increased when adversarial training was deployed, denoting that the IDS can detect AML attacks with high accuracy. This forms the fourth main contribution of this thesis:

C4 An investigation into how adversarial training may enhance the robustness of supervised IDSs.

Previous AML research focuses on other areas in cyber security (i.e. email spam and malware detection) and has yet to address the subject of how AML may affect supervised machine learning based IDSs trained on IoT network traffic data. Both C3 and C4

shed new light upon how an AML attack can be successfully deployed in an IoT network, subsequently bypassing current detection methods. This widens the knowledge base surrounding AML attacks in such environments, which supports the development of mechanisms towards defending against them.

To summarise, Figure 6.1 illustrates how all the contributions are linked together and play a key role in holistically securing an IoT smart home network.

Chapter 7

Conclusions and Future Work

7.1 Introduction

The research in this thesis was motivated by the fact that IoT devices, despite their proliferation in domestic environments and CNIs, introduce tremendous security flaws, and subsequently are subject to a range of cyber attacks. As such devices are often deeply embedded in networks, IoT may be considered as being the 'weakest link' for breaking into a secure infrastructure. Consequently, there is a significant need for the development of novel mechanisms to improve not only the defence of IoT against a range of cyber attacks, but also the detection of such attacks, and subsequently their mitigation from IoT networks.

Due to their limitations in computational power and their heterogeneity, securing the IoT ecosystem is considered as being a great challenge. This is because it is not feasible for IoT devices with restricted computational power to execute computationally intensive and latency-sensitive security tasks. As a result, it is not possible to employ complex and robust security measures. Moreover, the heterogeneity which surrounds IoT devices in terms of their hardware, software, and protocols poses as an obstacle towards developing and deploying security mechanisms that can endure with the scale and range of devices.

The central research of this thesis focused on the designs, implementations, and evaluations of mechanisms that aim to enhance the overall security of the IoT ecosys-

tem, whilst considering the aforementioned limitations. Firstly, this thesis presented a design and prototype implementation of a secure IoT hub. The proposed hub consists of various built-in security mechanisms that satisfy key security properties which can be applied to a range of devices, including authentication, confidentiality, and access control. The effectiveness of the hub was evaluated within a smart home IoT network consisting of a range of IoT devices upon which a number of popular cyberattacks were deployed. The evaluation demonstrated that the hub was successful in defending against such attacks.

Secondly, to complement the aforementioned secure hub and to further enhance the security of the ecosystem, this thesis investigated the feasibility of an IDS tailored for an IoT environment based on machine learning. As such, the initial experiments towards the development of a supervised three-layered IDS were presented. More specifically, the detection system aims to: 1) classify the IoT devices connected on the network, 2) identify whether network packets are malicious or benign, and 3) given malicious packets identified in 2), identify the type of attack which has occurred. To support the classification experiments, real network data was collected from a smart home testbed consisting of a range of IoT devices. The results demonstrate that the J4.8 Decision Tree outperformed other classifiers, achieving an F1-score of 97.4%, 99.9%, and 98.6% respectively.

Lastly, given the high performance of the detector, further evaluation against AML was conducted to investigate its robustness. Within this thesis, the initial experiments and a first approach towards automatically generating malicious DoS adversarial IoT network packets were presented. The results demonstrated that the performance of the proposed IDS is significantly decreased by a maximum of 31.6 percentage points when a range of packet features are perturbed. Subsequently, given the effectiveness of this attack, this work also explored how adversarial training can be used to enhance the robustness of supervised IDSs. Following adversarial training, the results show that the performance of the proposed IDS significantly improves.

Table 7.1 identifies where each research objective described in Chapter 1 was met within this thesis. While the contributions of this thesis have focused on improving upon the limitations found in various other approaches presented in the wider research space, some limitations remain and are discussed in Section 7.3 These form part of the future directions of this work discussed in Section 7.4.

Research Objective	Chapter Number
Analyse the literature surrounding the state-of-the-art approaches towards defending and detecting attacks in IoT environments to identify the limitations of current approaches.	2
Design a prototype of a secure hub framework to defend against cyber attacks that may threaten a smart home network.	3
Design and implement a typical IoT smart home network to deploy and evaluate the proposed hub using a traditional penetration testing methodology that corresponds to the attacker's objectives as defined in the contribution Chapter.	3
Identify the key requirements of a supervised IDS tailored for IoT environments.	4
Evaluate the feasibility and the performance of the proposed IDS using real network data derived from a typical IoT testbed consisting of a range of devices. This performance will be measured using standard machine learning classification metrics; Precision, Recall, F1-score. The goal is to maximise these measures as higher values correspond to better classification performance.	4
Further evaluate the robustness of the IDS against AML attacks. This will be achieved by developing a methodology to generate adversarial DoS packets and observing the classification performance metrics when such packets are presented to a trained supervised machine learning model.	5
Explore the effectiveness of adversarial training in increasing the robustness of machine learning detectors against AML attacks. This will be achieved by evaluating the performance of adversarial training.	5

Table 7.1: A summary of where each research objective was met within this thesis

7.2 Real-World Implementation

The overarching goal of this thesis was to explore how core limitations such as heterogeneity and low computational power, can be bypassed in order to create mechanisms that will aid in enhancing the security within an IoT home. It is evident that the inclusion of heterogeneous IoT devices within a home introduces a new threat vector that may have severe consequences. As a result, creating a well-rounded security infrastructure for IoT homes is crucial. Such an infrastructure consists of mechanisms that aim to secure the system in order to defend it from adversaries. Secondly, there is a need to be able to detect attacks within the IoT networks in order to be able to launch countermeasures faster and more efficiently. However, to increase the trust in these detectors, it is necessary to evaluate them further by presenting to them attacks that may be more challenging to detect (e.g. their behaviour is similar to legit traffic). Subsequently, by overcoming the heterogeneity of the devices and being able to apply a range of security mechanisms in a uniform way such devices, by introducing effective security monitoring solutions to aid in faster incidence response, and by bridging the gap of understanding around how robust these security solutions could be, this work has contributed in creating a more secure IoT environment.

More specifically, in order to apply uniformed security mechanisms to a range of IoT devices within a smart home network, Chapter 3 presents a novel secure hub that can defend a smart home ecosystem against two popular attacker models. The proposed hub is designed to contain five built-in security mechanisms that provide secure authentication, finer-grained policy-based access control, capability to monitor attack behaviours using IoT canary functions, device cloaking, and confidentiality, which enhance the overall security of the IoT ecosystem.

In a real use case, the prototype implementation of the hub may be deployed as it is currently configured. However, the user/admin would have to generate their own API keys for Pubnub and each module that uses an API to control the IoT devices. To add a new device to the hub infrastructure, a new module/interface must be implemented

and added to the gateway and a secure API must be available for the device to be controlled. This process can be completed remotely by the user/admin, by using the hub's dedicated remote access channel. Once the hub has been configured to support the devices within a smart home, users can control and access the devices remotely and securely. This is particularly useful to be able to perform traditional remote tasks such as turning on the thermostat before arriving home. To access the hub, a mobile or web application may be useful to connect and control the devices from one access point. In terms of scaling up the hub to cover several networks, a separate instance may be deployed in each sub-network. This will ensure that the devices are segregated and secured in their own ecosystem, ensuring that if a compromise occurs in one network, it will not propagate to the other networks.

In addition to defending smart home environments, it is equally important to be able to detect malicious activity on the network. The IoT smart home IDS presented in Chapter 4 demonstrated the efficiency of supervised machine learning to support the classification and identification of devices on the network, whether a packet is malicious or not, and subsequently, what type of attack has occurred with high accuracy. This means that very few packets would be incorrectly identified, which means that the false-positive rates are very low. This is an important characteristic of an IDS, as it increases its reliability towards detecting malicious behaviour and subsequently launching the appropriate countermeasures at the most critical times.

In a real use case, this tool may be implemented as a multi-label classification task, where each label represents the classification of devices, malicious or benign behaviour, and type of attack. Before deploying the tool on the network, and due to its supervised nature, a training phase is needed, where data from this specific network is collected and labelled. The labelling of such data can be automated using the MAC address of the devices on the network and associating this information with the vendors of the devices. This is an initial requirement for understanding the network's normal behaviour. To represent malicious network data, prior malicious labelled network packets

from well-known attacks may be integrated into the system. Such data can be accessed from resources such as a cyber range or open-source research datasets (e.g. [9]).

In order for the tool to detect the latest threats, as well as being able to recognise new devices added to the network, the model must be periodically retrained to capture such events. To address this, the system may be set to automatically re-train the model periodically or when a new device is added to the network. To avoid interrupting the smooth functionality of the system, this can be scheduled to occur at non-peak times (e.g. middle of the night).

Finally, given the possibly large number of IoT devices in a smart home, the IDS may be deployed in two ways. First, more devices may be added to the network and the model may be re-trained upon this data. However, this may affect the overall accuracy of the model, as it may be challenging for the model to distinctly recognise the network behaviours. Alternatively, it is good practice to segregate large networks into smaller sub-networks. In this case, the IDS may be deployed on each sub-network. This can ultimately lead to collaboration between the separate instances of the IDS to share their observations of possible attack indicators.

Before deploying the IDS on a smart home network, and having demonstrated how the robustness of such systems can be affected by AML attacks in Chapter 5, AML training can be used proactively to enhance the training dataset and subsequently the IDS's performance. More specifically, the collected network training dataset may be extended to also include possible perturbed malicious network packets. The methodology followed in Chapter 5, where malicious DoS packets were perturbed to mimic benign activity, can be adapted to include all possible DoS perturbations in the original training dataset. This has the potential to significantly increase the IDS's capability in detecting all DoS type of attacks. A similar approach as investigated herein may be applicable to other attack types.

To tie the aforementioned approaches together, in a real-world IoT network within a smart home, the proposed hub and its components may be employed to enhance the

security of the devices in a uniformed way. These mechanisms are key in significantly enhancing the security of IoT devices within a smart home network, subsequently making penetrating the network a challenge. However, adversaries will always continue to develop new attack techniques to compromise such systems. Therefore, in order to develop a more complete security model, the hub may be complemented with the addition of an IDS to automatically detect attacks and their types. The IDS may be positioned on the hub's gateway within the IoT hub's sub-network, where it would monitor the IoT network activity.

7.3 Limitations

Given the contributions and key observations discussed in Chapter 6, it is essential to highlight the limitations of the proposed methods which aim to detect and defend against cyber attacks in smart home networks.

One of the main limitations surrounding the proposed hub implementation is that it relies on a third-party provider to support the core actions within the framework. Subsequently, the hub always requires online access to manage local security which by default may introduce new security risks as it maintains an always open socket connection to every device. In addition, if such third-parties halted their services, the operations on the framework would be affected. Secondly, although APIs provide an accessible and user-friendly interface to access, add, and control the smart devices, they are often subject to offering the control of limited device functionality and may have limits for the number of requests that they can receive.

Regarding the proposed supervised machine learning IDS, one of the main limitations surrounding this work included the effort associated with the data labelling and feature engineering. In addition, the effectiveness of the system needs to be evaluated against more sophisticated attacks and within a much larger IoT network consisting of more devices of different types. Furthermore, as supervised machine learning was used to

support the experiments, the model must be re-trained to consider the classification of new and unseen attack categories. Finally, because the model is classifying individual packets based on packet features, it is possible to bypass such models using AML and it is also possible that the tool will be ineffective against attacks that are sequence-based and deployed over a period of time. For both the proposed Hub and the IDS, the limited financial resources restricted the number of devices that could be purchased to be included in the testbed. As such, it is worth noting that the work presented in this thesis was designed with such devices in mind.

Finally, regarding the proposed methods to evaluate and enhance the robustness of the supervised machine learning detector, one of the main limitations surrounding this work is the crude approach towards the perturbation of the chosen features and the manual overhead associated with analysing benign packet feature values. In addition, the applicability of this approach and other AML approaches of bypassing machine learning-based IDSs need to be further investigated for other attack types. Lastly, with regards to adversarial training, it is important to highlight that this method may not always be sufficient as it is difficult to anticipate all possible types of AML attacks against a given system. Therefore, there is a need to investigate other possible defence mechanisms.

As a whole, the main limitations of the work presented in this thesis include the access to a limited set of IoT devices of different types and the availability of a consistent IoT testbed environment. These two limitations contribute to the fact that the proposed mechanisms presented herein were not tested as part of one infrastructure. Such mechanisms were implemented and tested separately using a subset of common device types and within similar environments (i.e. local network). Given the positive findings, this demonstrates the potential that both mechanisms may be used in one secure framework and complement each other to secure and defend an IoT smart home.

7.4 Future Work

The contributions in this thesis can also be used as a foundation for future work, specifically in investigations towards developing more effective and lightweight security mechanisms for IoT ecosystems. While the security mechanisms implemented herein have demonstrated to successfully improve upon the limitations found in various common conventions in the wider research space, as discussed above, some limitations remain. These form part of the future directions of this work.

More specifically, as discussed in Chapter 3, given the hub's initial positive performance, it would be beneficial to scale up the framework by including more devices. Moreover, it would be beneficial to examine ways to improve its performance by integrating and monitoring other architectural aspects, such as installing a local broker (e.g. Mosquitto (mqtt)) to the gateway as opposed to utilising a third-party cloud provider. Furthermore, it would also be valuable to expand the threat model so that IoT devices are not only protected from cyber attacks deployed outside of the network, but also from attacks that initiate from other devices within the network itself.

Given the positive findings in Chapter 4, future work surrounding the presented IDS framework would include its implementation and evaluation in a real, much larger, heterogeneous IoT environment. This would allow the system to be further evaluated on more complex and sophisticated attacks. Moreover, in order to bypass the extensive need of feature engineering and date labeling, deep learning techniques can also be applied to automatically determine which packet features have an impact on the identification of malicious activity within the IoT environment. Finally, due to the vast attack surface and zero-day attacks, the effectiveness of unsupervised approaches, which do not depend on labeled data, can be explored.

Finally, Chapter 5 demonstrated that AML can pose as a serious threat on IDS systems that employ supervised machine learning. However, it is important to note that there exists several other methods that can potentially be employed to generate ad-

versarial samples in a more automated and sophisticated manner. Such approaches include black-box AML attacks, where the attacker has no knowledge of the system or the dataset, and thus utilise different models as a source for generating adversarial samples. The robustness of supervised IDSs may be demonstrated using adversarial training. Given the limitations associated with having to train a model to defend against all possible AML attack types, the robustness of such IDSs may be further improved by exploring other, more sophisticated defense mechanisms.

Bibliography

[1] 2019 global survey | iot-focused cyberattacks are the new normal. https://irdeto.com/news/new-2019-global-survey-iot-focused-cyberattacks-are-the-new-normal/. (Accessed on 05/06/2020).

[2] Advanced penetration testing methodologies & frameworks | purplesec. https://purplesec.us/penetration-testing-methodologies/. (Accessed on 03/30/2021).

[3] Amazon web services for iot. https://aws.amazon.com/iot/. (Accessed on 06/30/2021).

[4] Cisco visual networking index: Forecast and trends 2017-2022. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html. (Accessed on 03/26/2019).

[5] Crest penetration testing guide. https://www.crest-approved.org/wp-content/uploads/CREST-Penetration-Testing-Guide.pdf. (Accessed on 03/30/2021).

[6] Cyber hackers can now harm human life through smart meters: Smart grid awareness. https://smartgridawareness.org/2014/12/30/hackers-can-now-harm-human-life/. (Accessed on 01/08/2020).

[7] Cyber security incident response guide (csir). https://www.crest-approved.org/wp-content/uploads/2014/11/CSIR-Procurement-Guide.pdf. (Accessed on 07/30/2019).

[8] Darktrace | world-leading ai for cyber security. https://www.darktrace.com/en/. (Accessed on 07/28/2020).

[9] Datasets overview | stratosphere ips. https://www.stratosphereips.org/datasets-overview. (Accessed on 03/16/2021).

[10] dsniff: a network auditing and penetration testing tool suite. https://www.monkey.org/~dugsong/dsniff/. (Accessed on 05/20/2020).

[11] Dweet.io: Share your thing like it ain't no thang. https://dweet.io/. (Accessed on 11/14/2017).

[12] Eclipse kura: An open source framework for iot. https://www.eclipse.org/kura/. (Accessed on 11/14/2020).

[13] Encrypting messages | pubnub chat. https://www.pubnub.com/docs/chat/reference/encryption. (Accessed on 07/12/2020).

[14] Github | a bash script for recon and dos attacks. https://github.com/GinjaChris/pentmenu. (Accessed on 02/17/2020).

[15] Hewlett packard | study reveals 70 percent of internet of things devices vulnerable to attack. https://www8.hp.com/us/en/hp-news/press-release.html?id=1744676. (Accessed on 01/08/2020).

[16] Homeos: Enabling smarter homes for everyone. https://www.microsoft.com/en-us/research/project/homeos-enabling-smarter-homes-for-everyone/. (Accessed on 11/03/2020).

[17] How vulnerable is the internet of things | avast. https://blog.avast.com/how-vulnerable-is-the-internet-of-things. (Accessed on 06/14/2021).

[18] If this then that (ifttt). https://ifttt.com/. (Accessed on 11/03/2020).

[19] An industry standard technology for monitoring high speed switched networks. https://sflow.org/sflow_version_5.txt. (Accessed on 02/04/2020).

[20] Inside the smart home: Iot device threats and attack scenarios. https://www.trendmicro.com/vinfo/us/security/news/internet-of-things/inside-the-smart-home-iot-device-threats-and-attack-scenarios. (Accessed on 07/17/2020).

[21] Internet of things research study. https://d-russia.ru/wp-content/uploads/2015/10/4AA5-4759ENW.pdf. (Accessed on 06/15/2020).

[22] Internet of threats | iot botnets drive surge in network attacks. https://securityintelligence.com/posts/internet-of-threats-iot-botnets-network-attacks/. (Accessed on 06/14/2021).

[23] ios home | apple (uk). https://www.apple.com/uk/ios/home/. (Accessed on 11/03/2017).

[24] Iot reference model white paper. http://cdn.iotwf.com/resources/71/IoT_Reference_Model_White_Paper_June_4_2014.pdf. (Accessed on 07/29/2020).

[25] Kali linux, the penetration testing distribution documentation. https://docs.kali.org/. (Accessed on 02/15/2018).

[26] Literature review and focusing the research. https://www.sagepub.com/sites/default/files/upm-binaries/29986_Chapter3.pdf. (Accessed on 06/07/2021).

[27] Nest learning thermostat. https://nest.com/uk/thermostats/nest-learning-thermostat/overview/. (Accessed on 11/03/2020).

[28] On-premises vs. cloud iot deployment | behrtech blog. https://behrtech.com/blog/on-premises-vs-cloud-iot-deployment-which-is-best/. (Accessed on 06/30/2021).

[29] openhab: Open source automation software for your home. https://www.openhab.org/. (Accessed on 11/03/2020).

[30] Owasp internet of things project. https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project. (Accessed on 01/08/2020).

[31] Owasp top 10 internet of things. https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project#tab=IoT_Top_10. (Accessed on 07/26/2020).

[32] Pdml - the wireshark wiki. https://wiki.wireshark.org/PDML. (Accessed on 03/27/2020).

[33] Privacy policy. https://www.smartthings.com/privacy. (Accessed on 03/29/2021).

[34] Publish-subscribe (pub/sub). https://www.pubnub.com/learn/glossary/what-is-publish-subscribe/. (Accessed on 11/05/2020).

[35] Pubnub | making realtime innovation simple. https://www.pubnub.com. (Accessed on 11/07/2020).

[36] Samsung smartthings. https://www.samsung.com/uk/smartthings/. (Accessed on 03/23/2020).

[37] Scapy. https://scapy.net/. (Accessed on 05/05/2020).

[38] Scapy: Packet encapsulation. https://thepacketgeek.com/scapy-p-04-looking-at-packets/. (Accessed on 05/14/2020).

[39] Shodan. https://www.shodan.io/. (Accessed on 03/29/2021).

[40] Si6 networks. https://www.si6networks.com/tools/iot-toolkit/. (Accessed on 05/05/2020).

[41] Smart home devices and home automation systems, google store. https://store.google.com/gb/category/connected_home. (Accessed on 03/23/2020).

[42] Statista internet of things: The number of connected devices worldwide 2012-2025. https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. (Accessed on 02/14/2020).

[43] Synopsys report. https://www.synopsys.com/content/dam/synopsys/sig-assets/reports/medical-device-security-ponemon-synopsys.pdf. (Accessed on 03/16/2020).

[44] Veracode | your guide to application security solutions. https://info.veracode.com/. (Accessed on 07/28/2020).

[45] Weka 3 - data mining with open source machine learning software in java. https://www.cs.waikato.ac.nz/ml/weka/. (Accessed on 20/20/2020).

[46] Wireshark. https://www.wireshark.org/. (Accessed on 07/18/2018).

[47] Zoomeye - cyberspace search engine. https://www.zoomeye.org/. (Accessed on 03/29/2021).

[48] Mohamed Abomhara et al. Cyber security and the internet of things: vulnerabilities, threats, intruders and attacks. *Journal of Cyber Security and Mobility*, 4(1):65–88, 2015.

[49] Nasser S Abouzakhar, Andrew Jones, and Olga Angelopoulou. Internet of things security: A review of risks and threats to healthcare sector. In *2017 IEEE Inter-*

national Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pages 373–378. IEEE, 2017.

[50] Ahmed Abusnaina, Aminollah Khormali, Hisham Alasmary, Jeman Park, Afsah Anwar, and Aziz Mohaisen. Adversarial learning attacks on graph-based iot malware detection systems. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 1296–1305. IEEE, 2019.

[51] Abbas Acar, Hossein Fereidooni, Tigist Abera, Amit Kumar Sikder, Markus Miettinen, Hidayet Aksu, Mauro Conti, Ahmad-Reza Sadeghi, and Selcuk Uluagac. Peek-a-boo: I see your smart home activities, even encrypted! In *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 207–218, 2020.

[52] Amaal Al Shorman, Hossam Faris, and Ibrahim Aljarah. Unsupervised intelligent system based on one class support vector machine and grey wolf optimization for iot botnet detection. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–17, 2019.

[53] Atif Alamri, Wasai Shadab Ansari, Mohammad Mehedi Hassan, M Shamim Hossain, Abdulhameed Alelaiwi, and M Anwar Hossain. A survey on sensor-cloud: architecture, applications, and approaches. *International Journal of Distributed Sensor Networks*, 9(2):917923, 2013.

[54] Patrick Albers, Olivier Camp, Jean-Marc Percher, Bernard Jouga, Ludovic Me, and Ricardo Staciarini Puttini. Security in ad hoc networks: a general intrusion detection architecture enhancing trust based approaches. In *Wireless Information Systems*, pages 1–12, 2002.

[55] Mouhammd Alkasassbeh, Ghazi Al-Naymat, Ahmad Hassanat, and Mohammad Almseidin. Detecting distributed denial of service attacks using data mining

techniques. *International Journal of Advanced Computer Science and Applications*, 7(1):436–445, 2016.

[56] O. Alrawi, C. Lever, M. Antonakakis, and F. Monrose. Sok: Security evaluation of home-based iot deployments. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1362–1380, 2019.

[57] Omar Alrawi, Chaz Lever, Manos Antonakakis, and Fabian Monrose. Sok: Security evaluation of home-based iot deployments. In *2019 IEEE symposium on security and privacy (sp)*, pages 1362–1380. IEEE, 2019.

[58] Asma Alshehri and Ravi Sandhu. Access control models for cloud-enabled internet of things: A proposed architecture and research agenda. In *Collaboration and Internet Computing (CIC), 2016 IEEE 2nd International Conference on*, pages 530–538. IEEE, 2016.

[59] Mohammad Abu Alsheikh, Shaowei Lin, Dusit Niyato, and Hwee-Pink Tan. Machine learning in wireless sensor networks: Algorithms, strategies, and applications. *IEEE Communications Surveys & Tutorials*, 16(4):1996–2018, 2014.

[60] Amar Amouri, Vishwa T Alaparthy, and Salvatore D Morgera. Cross layer-based intrusion detection based on network behavior for iot. In *Wireless and Microwave Technology Conference (WAMICON), 2018 IEEE 19th*, pages 1–4. IEEE, 2018.

[61] Ioannis Andrea, Chrysostomos Chrysostomou, and George Hadjichristofi. Internet of things: Security vulnerabilities and challenges. In *Computers and Communication (ISCC), 2015 IEEE Symposium on*, pages 180–187. IEEE, 2015.

[62] Eirini Anthi, Shazaib Ahmad, Omer Rana, George Theodorakopoulos, and Pete Burnap. Eclipseiot: A secure and adaptive hub for the internet of things. *Computers & Security*, 78:477–490, 2018.

[63] Eirini Anthi, Amir Javed, Omer Rana, and George Theodorakopoulos. Secure data sharing and analysis in cloud-based energy management systems. In *Cloud Infrastructures, Services, and IoT Systems for Smart Cities*, pages 228–242. Springer, 2017.

[64] Eirini Anthi, Lowri Williams, and Pete Burnap. Pulse: An adaptive intrusion detection for the internet of things. *Living in the Internet of Things: Cybersecurity of the IoT-2018*, pages 1–4, 2018.

[65] Eirini Anthi, Lowri Williams, Amir Javed, and Pete Burnap. Hardening machine learning denial of service (dos) defences against adversarial attacks in iot smart home networks. *Computers & Security*, page 102352, 2021.

[66] Eirini Anthi, Lowri Williams, Gosia Malgortzata, George Theodorakopoulos, and Pete Burnap. A supervised intrusion detection system for smart home iot. *IEEE Internet of Things & Journal*, 78:477–490, 2018.

[67] Eirini Anthi, Lowri Williams, Matilda Rhode, Pete Burnap, and Adam Wedgbury. Adversarial attacks on machine learning cybersecurity defences in industrial control systems. *arXiv preprint arXiv:2004.05005*, 2020.

[68] Noah Apthorpe, Dillon Reisman, Srikanth Sundaresan, Arvind Narayanan, and Nick Feamster. Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic. *arXiv preprint arXiv:1708.05044*, 2017.

[69] Kevin Ashton et al. That âinternet of thingsâ thing. *RFID journal*, 22(7):97–114, 2009.

[70] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

[71] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[72] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Z. Yang. Automatic device classification from network traffic streams of internet of things. In *2018 IEEE 43rd Conference on Local Computer Networks (LCN)*, pages 1–9, 2018.

[73] Mario Baldi and Fulvio Risso. Using xml for efficient and modular packet processing. In *GLOBECOM'05. IEEE Global Telecommunications Conference, 2005.*, volume 1, pages 6–pp. IEEE, 2005.

[74] Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Amir Safavi, and Rui Zhang. Detecting poisoning attacks on machine learning in iot environments. In *2018 IEEE International Congress on Internet of Things (ICIOT)*, pages 57–64. IEEE, 2018.

[75] Marco Barreno, Blaine Nelson, Russell Sears, Anthony D Joseph, and J Doug Tygar. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, pages 16–25. ACM, 2006.

[76] Muhammad Bilal. A review of internet of things architecture, technologies and analysis smartphone-based attacks against 3d printers. *arXiv preprint arXiv:1708.04560*, 2017.

[77] Abdur Rahim Biswas and Raffaele Giaffreda. Iot and cloud convergence: Opportunities and challenges. In *2014 IEEE World Forum on Internet of Things (WF-IoT)*, pages 375–376. IEEE, 2014.

[78] Damiano Bolzoni, Sandro Etalle, and Pieter H Hartel. Panacea: Automating attack classification for anomaly-based network intrusion detection systems. In *International Workshop on Recent Advances in Intrusion Detection*, pages 1–20. Springer, 2009.

[79] Sacha Brostoff and M Angela Sasse. âten strikes and you're outâ: Increasing the number of login attempts can improve password usability. 2003.

[80] Olivier Brun, Yonghua Yin, and Erol Gelenbe. Deep learning with dense random neural network for detecting attacks against iot-connected home environments. *Procedia computer science*, 134:458–463, 2018.

[81] McAfee Cloud BU. 11 advantages of cloud computing and how your business can benefit from them. *Retrieved from in 2022 https://www.skyhighnetworks.com/cloud-security-blog/11-advantages-of-cloud-computing-and-how-your-business-can-benefit-from-them*, 2015.

[82] Janice Canedo and Anthony Skjellum. Using machine learning to secure iot systems. In *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, pages 219–222. IEEE, 2016.

[83] Alvaro A Cárdenas, Saurabh Amin, Zong-Syun Lin, Yu-Lun Huang, Chi-Yen Huang, and Shankar Sastry. Attacks against process control systems: risk assessment, detection, and response. In *Proceedings of the 6th ACM symposium on information, computer and communications security*, pages 355–366, 2011.

[84] Qifeng Chen, Haoming Chen, Yanpu Cai, Yanqi Zhang, and Xin Huang. Denial of service attack on iot system. In *2018 9th International Conference on Information Technology in Medicine and Education (ITME)*, pages 755–758. IEEE, 2018.

[85] Benoit Claise and Stewart Bryant. Specification of the ip flow information export (ipfix) protocol for the exchange of ip traffic flow information. Technical report, RFC 5101, January, 2008.

[86] Benoit Claise, Ganesh Sadasivan, Vamsi Valluri, and Martin Djernaes. Cisco systems netflow services export version 9. 2004.

[87] Luigi Coppolino, Valerio DAlessandro, Salvatore DAntonio, Leonid Levy, and Luigi Romano. My smart home is under attack. In *2015 IEEE 18th International Conference on Computational Science and Engineering*, pages 145–151. IEEE, 2015.

[88] Laura Daniele, Monika Solanki, Frank den Hartog, and Jasper Roes. Interoperability for smart appliances in the iot world. In *International Semantic Web Conference*, pages 21–29. Springer, 2016.

[89] Ria Das and Indrajit Das. Secure data transfer in iot environment: Adopting both cryptography and steganography techniques. In *2016 Second International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, pages 296–301. IEEE, 2016.

[90] Abebe Abeshu Diro and Naveen Chilamkurti. Distributed attack detection scheme using deep learning approach for internet of things. *Future Generation Computer Systems*, 82:761–768, 2018.

[91] Rohan Doshi, Noah Apthorpe, and Nick Feamster. Machine learning ddos detection for consumer internet of things devices. *arXiv preprint arXiv:1804.04159*, 2018.

[92] IBM Cloud Education. Benefits of cloud computing. *Retrieved from in 2022 https://www.ibm.com/cloud/learn/benefits-of-cloud-computing*, 2018.

[93] Alessandro Erba, Riccardo Taormina, Stefano Galelli, Marcello Pogliani, Michele Carminati, Stefano Zanero, and Nils Ole Tippenhauer. Real-time evasion attacks with physical constraints on deep learning-based anomaly detectors in industrial control systems. *arXiv preprint arXiv:1907.07487*, 2019.

[94] Evolvit. Cloud vs local servers: Weighing up the pros and cons. *Retrieved from in 2022 https://evolvit.co.uk/it-support/server-support/cloud-vs-local-servers-weighing-up-the-pros-and-cons/*, 2019.

[95] Ashfaq Hussain Farooqi and Farrukh Aslam Khan. Intrusion detection systems for wireless sensor networks: A survey. *Communication and networking*, pages 234–241, 2009.

[96] Kassem Fawaz, Kyu-Han Kim, and Kang G Shin. Protecting privacy of {BLE} device users. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 1205–1221, 2016.

[97] Mohamed Ali Feki, Fahim Kawsar, Mathieu Boussard, and Lieven Trappeniers. The internet of things: the next technological revolution. *Computer*, 46(2):24–25, 2013.

[98] David Formby, Preethi Srinivasan, Andrew M Leonard, Jonathan D Rogers, and Raheem A Beyah. Who's in control of your control system? device fingerprinting for cyber-physical systems. In *NDSS*, 2016.

[99] Andreas Fuchsberger. Intrusion detection systems and intrusion prevention systems. *Information Security Technical Report*, 10(3):134–139, 2005.

[100] Dimitris Geneiatakis, Ioannis Kounelis, Ricardo Neisse, Igor Nai-Fovino, Gary Steri, and Gianmarco Baldini. Security and privacy issues for an iot based smart home. In *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1292–1297. IEEE, 2017.

[101] André Glória, Francisco Cercas, and Nuno Souto. Design and implementation of an iot gateway to create smart environments. *Procedia Computer Science*, 109:568–575, 2017.

[102] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[103] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[104] Quandeng Gou, Lianshan Yan, Yihe Liu, and Yao Li. Construction and strategies in iot security system. In *2013 IEEE international conference on green computing and communications and IEEE internet of things and IEEE cyber, physical and social computing*, pages 1129–1132. IEEE, 2013.

[105] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. Adversarial examples for malware detection. In *European Symposium on Research in Computer Security*, pages 62–79. Springer, 2017.

[106] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

[107] Shang Guoqiang, Chen Yanming, Zuo Chao, and Zhu Yanxu. Design and implementation of a smart iot gateway. In *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing*, pages 720–723. IEEE, 2013.

[108] Anne Håkansson. Portal of research methods and methodologies for research projects and degree projects. In *The 2013 World Congress in Computer Science, Computer Engineering, and Applied Computing WORLDCOMP 2013; Las Vegas, Nevada, USA, 22-25 July*, pages 67–73. CSREA Press USA, 2013.

[109] Jun Han, Albert Jin Chung, Manal Kumar Sinha, Madhumitha Harishankar, Shijia Pan, Hae Young Noh, Pei Zhang, and Patrick Tague. Do you feel what i hear? enabling autonomous iot device pairing using different sensor types. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 836–852. IEEE, 2018.

[110] Te Han, Chao Liu, Wenguang Yang, and Dongxiang Jiang. A novel adversarial learning framework in deep convolutional neural network for intelligent diagnosis of mechanical faults. *Knowledge-Based Systems*, 165:474–487, 2019.

[111] Ian G Harris. Social engineering attacks on the internet of things. *IoT Newsletter*, 2016.

[112] Blake T Henderson. A honeypot for spies: Understanding internet-based data theft. Technical report, NAVAL POSTGRADUATE SCHOOL MONTEREY CA MONTEREY United States, 2018.

[113] Chia-Chun Hsu. Enhanced access control in lte advanced systems, January 20 2015. US Patent 8,938,233.

[114] Weiwei Hu and Ying Tan. Generating adversarial malware examples for black-box attacks based on gan. *arXiv preprint arXiv:1702.05983*, 2017.

[115] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and J Doug Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, pages 43–58. ACM, 2011.

[116] Philokypros Ioulianou, Vasileios Vasilakis, Ioannis Moscholios, and Michael Logothetis. A signature-based intrusion detection system for the internet of things. *Information and Communication Technology Form*, 2018.

[117] Qi Jing, Athanasios V Vasilakos, Jiafu Wan, Jingwei Lu, and Dechao Qiu. Security of the internet of things: Perspectives and challenges. *Wireless Networks*, 20(8):2481–2501, 2014.

[118] Chen Jun and Chen Chi. Design of complex event-processing ids in internet of things. In *Measuring Technology and Mechatronics Automation (ICMTMA), 2014 Sixth International Conference on*, pages 226–229. IEEE, 2014.

[119] Oleg Kachirski and Ratan Guha. Intrusion detection using mobile agents in wireless ad hoc networks. In *Proceedings. IEEE Workshop on Knowledge Media Networking*, pages 153–158. IEEE, 2002.

[120] Prabhakaran Kasinathan, Claudio Pastrone, Maurizio A Spirito, and Mark Vinkovits. Denial-of-service detection in 6lowpan based internet of things. In

Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on, pages 600–607. IEEE, 2013.

[121] Rafiullah Khan, Sarmad Ullah Khan, Rifaqat Zaheer, and Shahid Khan. Future internet: the internet of things architecture, possible applications and key challenges. In *2012 10th international conference on frontiers of information technology*, pages 257–260. IEEE, 2012.

[122] Richard Lippmann Engin Kirda and Ari Trachtenberg. Recent advances in intrusion detection. *Lecture Notes in Computer Science*, 5758, 2009.

[123] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. Ddos in the iot: Mirai and other botnets. *Computer*, 50(7):80–84, 2017.

[124] Jahoon Koo, Se-Ra Oh, and Young-Gab Kim. Device identification interoperability in heterogeneous iot platforms. *Sensors*, 19(6):1433, 2019.

[125] Thomas Kothmayr, Corinna Schmitt, Wen Hu, Michael Brünig, and Georg Carle. A dtls based end-to-end security architecture for the internet of things with two-way authentication. In *Local Computer Networks Workshops (LCN Workshops), 2012 IEEE 37th Conference on*, pages 956–963. IEEE, 2012.

[126] Deepak Kumar, Kelly Shen, Benton Case, Deepali Garg, Galina Alperovich, Dmitry Kuznetsov, Rajarshi Gupta, and Zakir Durumeric. All things considered: an analysis of iot devices on home networks. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 1169–1185, 2019.

[127] Shantha Kumari. Agility on cloud â a vital part of cloud computing. *Retrieved from in 2022 https://blog.sysfore.com/agility-on-cloud-a-vital-part-of-cloud-computing/*, 2015.

[128] Changmin Lee, Luca Zappaterra, Kwanghee Choi, and Hyeong-Ah Choi. Securing smart home: Technologies, security challenges, and security requirements.

In *2014 IEEE Conference on Communications and Network Security*, pages 67–72. IEEE, 2014.

[129] In Lee and Kyoochun Lee. The internet of things (iot): Applications, investments, and challenges for enterprises. *Business Horizons*, 58(4):431–440, 2015.

[130] Shancang Li. Chapter 1 - introduction: Securing the internet of things. In Shancang Li and Li Da Xu, editors, *Securing the Internet of Things*, pages 1–25. Syngress, Boston, 2017.

[131] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24, 2013.

[132] Donggang Liu, Peng Ning, and Rongfang Li. Establishing pairwise keys in distributed sensor networks. *ACM Transactions on Information and System Security (TISSEC)*, 8(1):41–77, 2005.

[133] Yu Liu, Kin-Fai Tong, Xiangdong Qiu, Ying Liu, and Xuyang Ding. Wireless mesh networks in iot networks. In *2017 International Workshop on Electromagnetics: Applications and Student Innovation Competition*, pages 183–185. IEEE, 2017.

[134] Franco Loi, Arunan Sivanathan, Hassan Habibi Gharakheili, Adam Radford, and Vijay Sivaraman. Systematically evaluating security and privacy for consumer iot devices. In *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy*, pages 1–6, 2017.

[135] Irene Lopatovska, Katrina Rink, Ian Knight, Kieran Raines, Kevin Cosenza, Harriet Williams, Perachya Sorsche, David Hirsch, Qi Li, and Adrianna Martinez. Talk to me: Exploring user interactions with the amazon alexa. *Journal of Librarianship and Information Science*, 51(4):984–997, 2019.

[136] Rongxing Lu, Kevin Heung, Arash Habibi Lashkari, and Ali A Ghorbani. A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced iot. *IEEE Access*, 5:3302–3312, 2017.

[137] Gordon Fyodor Lyon. *Nmap network scanning: The official Nmap project guide to network discovery and security scanning*. Insecure, 2009.

[138] J Ma, Y Guo, J Ma, J Xiong, and T Zhang. A hierarchical access control scheme for perceptual layer of iot, jisuanji yanjiu yu fazhan/comput. *Res. Dev*, 50(6):1267–1275, 2013.

[139] Parikshit N Mahalle, Neeli Rashmi Prasad, and Ramjee Prasad. Threshold cryptography-based group authentication (tcga) scheme for the internet of things (iot). In *2014 4th International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace & Electronic Systems (VITAE)*, pages 1–5. IEEE, 2014.

[140] Antonio Mangino, Morteza Safaei Pour, and Elias Bou-Harb. Internet-scale insecurity of consumer internet of things: An empirical measurements perspective. *ACM Transactions on Management Information Systems (TMIS)*, 11(4):1–24, 2020.

[141] Christopher D McDermott, Farzan Majdani, and Andrei V Petrovski. Botnet detection in the internet of things using deep learning approaches. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[142] James P McDermott. Attack net penetration testing. In *Proceedings of the 2000 workshop on New security paradigms*, pages 15–21, 2001.

[143] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, 2008.

[144] Yair Meidan, Michael Bohadana, Yael Mathov, Yisroel Mirsky, Asaf Shabtai, Dominik Breitenbacher, and Yuval Elovici. N-baiotânetwork-based detection of iot botnet attacks using deep autoencoders. *IEEE Pervasive Computing*, 17(3):12–22, 2018.

[145] Yair Meidan, Michael Bohadana, Asaf Shabtai, Juan David Guarnizo, Martín Ochoa, Nils Ole Tippenhauer, and Yuval Elovici. Profiliot: a machine learning approach for iot device identification based on network traffic analysis. In *Proceedings of the Symposium on Applied Computing*, pages 506–509. ACM, 2017.

[146] Daniele Midi, Antonino Rullo, Anand Mudgerikar, and Elisa Bertino. Kalisâa system for knowledge-driven adaptable intrusion detection for the internet of things. In *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*, pages 656–666. IEEE, 2017.

[147] Mohammad-Mahdi Moazzami, Guoliang Xing, Daisuke Mashima, Wei-Peng Chen, and Ulrich Herberg. Spot: A smartphone-based platform to tackle heterogeneity in smart-home iot systems. In *Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on*, pages 514–519. IEEE, 2016.

[148] Blaine Nelson, Marco Barreno, Fuching Jack Chi, Anthony D Joseph, Benjamin IP Rubinstein, Udam Saini, Charles A Sutton, J Doug Tygar, and Kai Xia. Exploiting machine learning to subvert your spam filter. *LEET*, 8:1–9, 2008.

[149] Edith CH Ngai, Jiangchuan Liu, and Michael R Lyu. On the intruder detection for sinkhole attack in wireless sensor networks. In *ICC*, volume 6, pages 3383–3389. Citeseer, 2006.

[150] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

[151] TJ OConnor, William Enck, and Bradley Reaves. Blinded and confused: uncovering systemic flaws in device telemetry for smart-home internet of things. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, pages 140–150, 2019.

[152] Doohwan Oh, Deokho Kim, and Won Woo Ro. A malicious pattern detection engine for embedded security systems in the internet of things. *Sensors*, 14(12):24188–24211, 2014.

[153] Se-Ra Oh and Young-Gab Kim. Security requirements analysis for the iot. In *2017 International Conference on Platform Technology and Service (PlatCon)*, pages 1–6. IEEE, 2017.

[154] Suad Mohammed Othman, Fadl Mutaher Ba-Alwi, Nabeel T Alsohybe, and Amal Y Al-Hashida. Intrusion detection model using machine learning algorithm on big data environment. *Journal of Big Data*, 5(1):34, 2018.

[155] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

[156] Tamas Pflanzner and Attila Kertész. A survey of iot cloud providers. In *2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 730–735. IEEE, 2016.

[157] Abhishek Pharate, Harsha Bhat, Vaibhav Shilimkar, and Nalini Mhetre. Classification of intrusion detection system. *International Journal of Computer Applications*, 118(7), 2015.

[158] Pavan Pongle and Gurunath Chavan. Real time intrusion and wormhole attack detection in internet of things. *International Journal of Computer Applications*, 121(9), 2015.

[159] Segun I Popoola, Bamidele Adebisi, Ruth Ande, Mohammad Hammoudeh, Kelvin Anoh, and Aderemi A Atayero. Smote-drnn: A deep learning algorithm for botnet detection in the internet-of-things networks. *Sensors*, 21(9):2985, 2021.

[160] Pawani Porambage, Corinna Schmitt, Pardeep Kumar, Andrei Gurtov, and Mika Ylianttila. Two-phase authentication protocol for wireless sensor networks in distributed iot applications. In *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 2728–2733. Ieee, 2014.

[161] H Pranata, R Athauda, and Geoff Skinner. Securing and governing access in adhoc networks of internet of things. In *Proceedings of the IASTED International Conference on Engineering and Applied Science, EAS*, pages 84–90, 2012.

[162] Kathy Pretz. Exploring the impact of the internet of things: A new ieee group is taking on the quest to connect everything. *The Institute*, 2013.

[163] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[164] Sebastian M Rasinger. Quantitative methods: Concepts, frameworks and issues. *Research methods in linguistics*, pages 49–67.

[165] Shailendra Rathore and Jong Hyuk Park. Semi-supervised learning based distributed attack detection framework for iot. *Applied Soft Computing*, 72:79–89, 2018.

[166] Shahid Raza, Linus Wallgren, and Thiemo Voigt. Svelte: Real-time intrusion detection in the internet of things. *Ad hoc networks*, 11(8):2661–2674, 2013.

[167] Cristanel Razafimandimby, Valeria Loscrí, Anna Maria Vegni, and Alessandro Neri. A bayesian and smart gateway based communication for noisy iot scenario. In *Computing, Networking and Communications (ICNC), 2017 International Conference on*, pages 481–485. IEEE, 2017.

[168] Francesco Restuccia, Salvatore DâOro, and Tommaso Melodia. Securing the internet of things in the age of machine learning and software-defined networking. *IEEE Internet of Things Journal*, 5(6):4829–4842, 2018.

[169] Maria Rigaki. Adversarial deep learning against intrusion detection classifiers, 2017.

[170] Markus Ring, Sarah Wunderlich, Deniz Scheuring, Dieter Landes, and Andreas Hotho. A survey of network-based intrusion detection data sets. *Computers & Security*, 2019.

[171] Martin Roesch et al. Snort: Lightweight intrusion detection for networks. In *Lisa*, volume 99, pages 229–238, 1999.

[172] Brian Russell and Drew Van Duren. *Practical Internet of Things Security: Design a security framework for an Internet connected ecosystem*. Packt Publishing Ltd, 2018.

[173] Maheshkumar Sabhnani and Gürsel Serpen. Application of machine learning algorithms to kdd intrusion detection dataset within misuse detection context. In *MLMTA*, pages 209–215, 2003.

[174] Ahmed Saeed, Ali Ahmadinia, Abbas Javed, and Hadi Larijani. Intelligent intrusion detection in low-power iots. *ACM Transactions on Internet Technology (TOIT)*, 16(4):1–25, 2016.

[175] Leonel Santos, Carlos Rabadao, and Ramiro Gonçalves. Intrusion detection systems in internet of things: A literature review. In *2018 13th Iberian Conference on Information Systems and Technologies (CISTI)*. IEEE, 2018.

[176] Navrati Saxena, Abhishek Roy, Bharat JR Sahu, and HanSeok Kim. Efficient iot gateway over 5g wireless: A new design with prototype and implementation results. *IEEE Communications Magazine*, 55(2):97–105, 2017.

[177] Pallavi Sethi and Smruti R Sarangi. Internet of things: architectures, protocols, and applications. *Journal of Electrical and Computer Engineering*, 2017, 2017.

[178] Anuj Sharma and Shubhamoy Dey. Performance investigation of feature selection methods and sentiment lexicons for sentiment analysis. *IJCA Special Issue on Advanced Computing and Communication Technologies for HPC Applications*, 3:15–20, 2012.

[179] Tariqahmad Sherasiya and Hardik Upadhyay. Intrusion detection system for internet of things. *International Journal of Advance Research and Innovative Ideas in Education*, 2(3).

[180] Dharmini Shreenivas, Shahid Raza, and Thiemo Voigt. Intrusion detection in the rpl-connected 6lowpan networks. In *Proceedings of the 3rd ACM International Workshop on IoT Privacy, Trust, and Security*, pages 31–38. ACM, 2017.

[181] Prachi Shukla. Ml-ids: A machine learning approach to detect wormhole attacks in internet of things. In *Intelligent Systems Conference (IntelliSys), 2017*, pages 234–240. IEEE, 2017.

[182] Tobby Simon. Chapter seven: Critical infrastructure and the internet of things. *Cyber Security in a Volatile World*, 93, 2017.

[183] Anna Kornfeld Simpson, Franziska Roesner, and Tadayoshi Kohno. Securing vulnerable home iot devices with an in-hub security manager. In *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 551–556. IEEE, 2017.

[184] Jayveer Singh and Manisha J Nene. A survey on machine learning techniques for intrusion detection systems. *International Journal of Advanced Research in Computer and Communication Engineering*, 2(11):4349–4355, 2013.

[185] Arunan Sivanathan, Daniel Sherratt, Hassan Habibi Gharakheili, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. Charac-

terizing and classifying iot traffic in smart cities and campuses. In *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 559–564. IEEE, 2017.

[186] Vijay Sivaraman, Hassan Habibi Gharakheili, Clinton Fernandes, Narelle Clark, and Tanya Karliychuk. Smart iot devices in the home: Security and privacy implications. *IEEE Technology and Society Magazine*, 37(2):71–79, 2018.

[187] Vijay Sivaraman, Hassan Habibi Gharakheili, Arun Vishwanath, Roksana Boreli, and Olivier Mehani. Network-level security and privacy control for smart-home iot devices. In *2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 163–167. IEEE, 2015.

[188] Tianyi Song, Ruinian Li, Bo Mei, Jiguo Yu, Xiaoshuang Xing, and Xiuzhen Cheng. A privacy preserving communication protocol for iot applications in smart homes. *IEEE Internet of Things Journal*, 4(6):1844–1852, 2017.

[189] Priyanshu Srivastava and Rizwan Khan. A review paper on cloud computing. *International Journal of Advanced Research in Computer Science and Software Engineering*, 8(6):17–20, 2018.

[190] R Stephen and L Arockiam. Intrusion detection system to detect sinkhole attack on rpl protocol in internet of things. *International Journal of Electrical Electronics and Computer Science*, 4(4):16–20, 2017.

[191] Biljana L Risteska Stojkoska and Kire V Trivodaliev. A review of internet of things for smart home: Challenges and solutions. *Journal of Cleaner Production*, 140:1454–1464, 2017.

[192] Basant Subba, Santosh Biswas, and Sushanta Karmakar. A neural network based system for intrusion detection and attack classification. In *2016 Twenty Second National Conference on Communication (NCC)*, pages 1–6. IEEE, 2016.

[193] Douglas H Summerville, Kenneth M Zach, and Yu Chen. Ultra-lightweight deep packet anomaly detection for internet of things devices. In *Computing and Communications Conference (IPCCC), 2015 IEEE 34th International Performance*, pages 1–8. IEEE, 2015.

[194] Harald Sundmaeker, Patrick Guillemin, Peter Friess, and Sylvie Woelfflé. Vision and challenges for realising the internet of things. *Cluster of European research projects on the internet of things, European Commision*, 3(3):34–36, 2010.

[195] Zhiyuan Tan, Aruna Jamdagni, Xiangjian He, Priyadarsi Nanda, and Ren Ping Liu. A system for denial-of-service attack detection based on multivariate correlation analysis. *IEEE transactions on parallel and distributed systems*, 25(2):447–456, 2013.

[196] Nanda Kumar Thanigaivelan, Ethiopia Nigussie, Rajeev Kumar Kanth, Seppo Virtanen, and Jouni Isoaho. Distributed internal anomaly detection system for internet-of-things. In *Consumer Communications & Networking Conference (CCNC), 2016 13th IEEE Annual*, pages 319–320. IEEE, 2016.

[197] Chih-Fong Tsai, Yu-Feng Hsu, Chia-Ying Lin, and Wei-Yang Lin. Intrusion detection by machine learning: A review. *expert systems with applications*, 36(10):11994–12000, 2009.

[198] Mathy Vanhoef and Frank Piessens. Advanced wi-fi attacks using commodity hardware. In *Proceedings of the 30th Annual Computer Security Applications Conference*, pages 256–265, 2014.

[199] Mathy Vanhoef and Frank Piessens. All your biases belong to us: Breaking rc4 in wpa-tkip and {TLS}. In *24th {USENIX} Security Symposium ({USENIX} Security 15)*, pages 97–112, 2015.

[200] Mathy Vanhoef and Frank Piessens. Predicting, decrypting, and abusing wpa2/802.11 group keys. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 673–688, 2016.

[201] John R Venable, Jan Pries-Heje, and Richard L Baskerville. Choosing a design science research methodology. 2017.

[202] Abhishek Verma and Virender Ranga. Machine learning based intrusion detection systems for iot applications. *Wireless Personal Communications*, pages 1–24, 2019.

[203] Michael Vögler, Johannes Schleicher, Christian Inzinger, Stefan Nastic, Sanjin Sehic, and Schahram Dustdar. Leonore–large-scale provisioning of resource-constrained iot deployments. In *2015 IEEE Symposium on Service-Oriented System Engineering*, pages 78–87. IEEE, 2015.

[204] Ben Whitham. Canary files: generating fake files to detect critical data loss from complex computer networks. In *Second International Conference on Cyber Security, Cyber Peacefare and Digital Forensic (CyberSec2013), Malaysia*, 2013.

[205] David H Wolpert. The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390, 1996.

[206] Miao Wu, Ting-Jie Lu, Fei-Yang Ling, Jing Sun, and Hui-Ying Du. Research on the architecture of internet of things. In *2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*, volume 5, pages V5–484. IEEE, 2010.

[207] Liang Xiao, Yan Li, Guoan Han, Guolong Liu, and Weihua Zhuang. Phy-layer spoofing detection with reinforcement learning in wireless networks. *IEEE Transactions on Vehicular Technology*, 65(12):10037–10047, 2016.

[208] Liang Xiao, Xiaoyue Wan, Xiaozhen Lu, Yanyong Zhang, and Di Wu. Iot security techniques based on machine learning. *arXiv preprint arXiv:1801.06275*, 2018.

[209] Teng Xu, James B Wendt, and Miodrag Potkonjak. Security of iot systems: Design challenges and opportunities. In *Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design*, pages 417–423. IEEE Press, 2014.

[210] Shakiba Yaghoubi and Georgios Fainekos. Gray-box adversarial testing for control systems with machine learning components. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, pages 179–184, 2019.

[211] Xuanxia Yao, Xiaoguang Han, Xiaojiang Du, and Xianwei Zhou. A lightweight multicast authentication mechanism for small scale iot applications. *IEEE Sensors Journal*, 13(10):3693–3701, 2013.

[212] Ibrar Yaqoob, Ejaz Ahmed, Ibrahim Abaker Targio Hashem, Abdelmuttlib Ibrahim Abdalla Ahmed, Abdullah Gani, Muhammad Imran, and Mohsen Guizani. Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges. *IEEE wireless communications*, 24(3):10–16, 2017.

[213] Ning Ye, Yan Zhu, Ru-chuan Wang, and Qiao-min Lin. An efficient authentication and access control scheme for perception layer of internet of things. *Institute of Electrical and Electronics Engineers, 2014-07*, 2014.

[214] Shan Yin, Yueming Lu, and Yonghua Li. Design and implementation of iot centralized management model with linkage policy, 2015.

[215] Tianlong Yu, Vyas Sekar, Srinivasan Seshan, Yuvraj Agarwal, and Chenren Xu. Handling a trillion (unfixable) flaws on a billion devices: Rethinking network security for the internet-of-things. In *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*, page 5. ACM, 2015.

[216] Yongguang Zhang and Wenke Lee. Intrusion detection in wireless ad-hoc networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 275–283, 2000.

[217] Yongguang Zhang, Wenke Lee, and Yi-An Huang. Intrusion detection techniques for mobile wireless networks. *Wireless Networks*, 9(5):545–556, 2003.

[218] ZhiKai Zhang, Michael Cheng Yi Cho, ChiaWei Wang, ChiaWei Hsu, Chong-Kuan Chen, and Shiuhpyng Shieh. Iot security: ongoing challenges and research opportunities. In *2014 IEEE 7th international conference on service-oriented computing and applications*, pages 230–234. IEEE, 2014.

[219] Yan Ling Zhao. Research on data security technology in internet of things. In *Applied Mechanics and Materials*, volume 433, pages 1752–1755. Trans Tech Publ, 2013.

[220] Zhiyuan Zheng, Allen Webb, AL Narasimha Reddy, and Riccardo Bettati. Iotaegis: A scalable framework to secure the internet of things. In *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9. IEEE, 2018.

[221] Yan Zhou, Murat Kantarcioglu, Bhavani Thuraisingham, and Bowei Xi. Adversarial support vector machine learning. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1059–1067, 2012.

[222] Giulio Zizzo, Chris Hankin, Sergio Maffeis, and Kevin Jones. Adversarial machine learning beyond the image domain. In *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pages 1–4. IEEE, 2019.

[223] MZWM Zulkifli and Zaid W Mohd. Attack on cryptography. *Comput. Secur*, 12(5):33–45, 2008.

Appendix

Table A1: Packet features

Feature	Description
len	Total packet length
caplen	Length of the capture
frame.encap_type	Frame Encapsulation Type
frame.offset_shift	Time shift for this packet
frame.len	Frame length on the wire
frame.cap_len	Frame length stored into the capture file
frame.marked	Frame is marked
frame.ignored	Frame is ignored
eth.lg	Most significant byte of MAC Address
eth.ig	Least significant byte of MAC Address
ip.version	Internet Protocol (IP) version
ip.hdr_len	IP Header Length
ip.dsfield.dscp	Differentiated Services Codepoint
ip.dsfield.ecn	Explicit Congestion Notification
ip.src	IP Source Address
ip.dst	IP Destination Address
ip.len	Total Length
ip.flags	IP Flags

ip.flags.rb	IP Reserved bit flag
ip.flags.df	Don't fragment flag
ip.flags.mf	More fragments flag
ip.frag_offset	IP Fragment Offset
ip.ttl	IP Time to Live
ip.proto	Protocol
ip.checksum.status	IP Header Checksum
tcp.srcport	TCP Source Port
tcp.dstport	TCP Destination Port
tcp.stream	TCP Stream Index
tcp.len	TCP Segment Len
tcp.seq	TCP Sequence Number
tcp.nxtseq	TCP Next Sequence Number
tcp.ack	TCP Acknowledgment Number
tcp.hdr_len	TCP Header Length
tcp.flags.res	TCP Reserved flag
tcp.flags.ns	TCP Nonce flag
tcp.flags.cwr	Congestion Window Reduced (CWR) flag
tcp.flags.ecn	ECN-Echo flag
tcp.flags.urg	TCP Urgent flag
tcp.flags.ack	TCP Acknowledgment flag
tcp.flags.push	TCP Push flag
tcp.flags.reset	TCP Reset flag
tcp.flags.syn	TCP Synchronisation flag
tcp.flags.fin	TCP Finish flag
tcp.window_size_value	TCP Window Value
tcp.window_size	TCP Window Size
tcp.window_size_scalefactor	TCP Window size scaling factor

tcp.checksum.status	TCP Checksum Status
tcp.urgent_pointer	TCP Urgent Pointer
tcp.options.nop	TCP No Option field
tcp.options.mss_val	Largest amount of data in a single TCP segment
tcp.options.sack_perm	TCP SACK Permitted Option
tcp.analysis.bytes_in_flight	Bytes in TCP flight
tcp.analysis.push_bytes_sent	Bytes sent since last PSH flag
tcp.time_delta	Time since previous frame in this TCP stream
bootp.hw.type	Hardware type
bootp.hw.len	Hardware address length
bootp.hops	Number of Hops
bootp.secs	Seconds elapsed
bootp.flags.bc	Broadcast flag
bootp.flags.reserved	Reserved flags
bootp.dhcp	The frame is marked as DHCP
icmp.resp_in	Response frame
icmp.resp_to	Request frame
data.len	Length of data following TCP stream interpretation
ssl.record.content_type	Content Type
ssl.record.version	SSL Version
ssl.record.length	SSL Record Length
arp.hw.type	ARP Hardware type
arp.proto.type	ARP Protocol size
arp.hw.size	ARP Hardware size
arp.proto.size	ARP Protocol size
arp.opcode	ARP Opcode
http.response.code	HTTP Response status code
http.content_length	HTTP Content length

http.response	HTTP Response
http.response_number	Response number
http.request	HTTP Request
http.request_number	Request number
classicstun.type	Simple Traversal of UDP Through NAT/Type
classicstun.length	Simple Traversal of UDP Through NAT/Length
udp.srcport	UDP Source Port
udp.dstport	UDP Destination Port
udp.length	UDP Length
udp.checksum.status	UDP Checksum Status
udp.stream	UDP Stream index
dns.flags.response	DNS Response
dns.flags.opcode	DNS Opcode value
dns.flags.truncated	Truncated field
dns.flags.recdesired	Recursion desired Option
dns.flags.z	Most significant bit of the Z field on header query
dns.flags.checkdisable	Non-authenticated data option
dns.flags.rcode	DNS Reply code
dns.count.queries	Total number of DNS queries
dns.count.answers	Total number of DNS answers
dns.count.auth_rr	Authority of Resource Records
dns.qry.name.len	Query Name
dns.count.labels	DNS Label Count
dns.resp.type	DNS Response Type
dns.resp.class	Response Class
dns.resp.ttl	DNS Time to live
dns.resp.len	Data length
igmp.version	IGMP Version

igmp.type	Type of IGMP
igmp.max_resp	IGMP Maximum Response Time
igmp.checksum.status	IGMP Checksum Status
ntp.flags.li	NTP Leap Indicator
ntp.flags.vn	Version number
ntp.flags.mode	NTP Flag Mode
ntp.stratum	Peer Clock Stratum
ntp.ppoll	Peer Polling Interval
ntp.rootdelay	Root Delay
ntp.rootdispersion	Root Dispersion
ntp.precision	Peer Clock Precision
bootp.type	Bootstrap Message type
tcp.payload	TCP payload
icmp.type	ICMP Type
icmp.code	ICMP Code
icmp.ident	Identifier (Big Endian)
icmp.checksum.status	Checksum Status
icmp.seq	Sequence Number (Big Endian)
icmp.seq_le	Sequence Number (Little Endian)

Device	**Module**	**Functions**
Philips Hue	*philapi.py*	show_hues() - lamps available, light_switch() - turn on/off, light_brightness() - adjust by x%
NC200 Camera	*ip_camera.py*	take_snapshot() - takes a snapshot, uploads it to a folder, and forwards the folder url to the user
Samsung Smart Things	*smart_things.py*	list_types() - show sensor types (e.g. switch) list_devices() - show connected devices toggle_switch() - Switch device can be toggled on/off. device_state() - Show device's state
LG Smart TV	*lg.tv.py*	get_volume() - Shows current volume set_volume() - Sets volume to an x% volume_up() - Turns volume up volume_down() - Turns volume down open_url() - opens the TVâs browser get_apps() - Displays all the available apps on TV launch_app() - Launches an app current_app() - Returns name of current open app close_app() - Closes an app get_services() - Displays services (e.g. tv guide) get_inputs() - shows available inputs (e.g.HDMI1) set_input()- switch to a specific input switch_3d_on - Turns on the 3D mode switch_3d_off()- Turns the 3D off power_on() - Turns the TV on power_off() - Turns the TV off
Embedded device	*embedded.py*	led_on() - Turns the LED light on led_off() - Turns the LED light off blink() - Blinks the LED light

Table A2: The add-on modules implemented for each IoT device along with the functions they contained.

Attack	Type	Description	Command
Reconnaissance	Intense Scan	Scans a list consisting of 1,000 of the most common TCP ports in order to determine the Operating System (OS) type as well as the services running.	nmap -T4 -A -v /target
	Intense Scan Plus UDP	Performs similarly to the Intense Scan, as well as scanning the UDP ports.	nmap -sS -sU -T4 -A -v /target/
	Intense scan, no ping	Performs similarly to the aforementioned scanning types. However, this particular type of scan is useful if the target is blocking ping requests and it is known that the host is active.	nmap -T4 -A -v -Pn /target/
	Intense Scan - all TCP ports	Scans with the default settings. It issues a TCP SYN scan for 1,000 of the most common TCP ports using ICMP Echo requests (pinging) for host detection.	nmap -v /target/
	Regular Scan	Scans with the default settings. It issues a TCP SYN scan for 1,000 of the most common TCP ports using ICMP Echo pinging for host detection.	nmap -v /target/
	Quick Scan	Scans a list consisting of 100 of the most common TCP ports.	nmap -T4 -F -v /target/
	Quick Scan Plus	Scans a list consisting of 100 of the most common TCP ports in order to determine the Operating System (OS) type as well as the services running.	nmap -sV -T4 -O -F -version-light -v /target/
	Ping	Pings the target and does not perform port scanning.	nmap -sn -v /target/
	Quick Traceroute	Traceroutes and pings all the specified hosts. This option is used to determine hosts and routers in a network scan.	nmap -sn -traceroute -v /target/
DoS	ICMP Echo Flood	Floods the target by sending ICMP echo-requests.	hping3 -1 -flood -rand-source /target/
	ICMP Blacknurse attack	performs similarly to ICMP Echo Flood, but sends ICMP packets by specifying the ICMP type as 3.	hping3 -1 -C 3 -K 3 -flood -rand-source /target
	TCP SYN Flood	Specifies the number of data bytes to send to a specific port. It also splits the packets into more fragments and sets the SYN TCP flag.	hping -d 3000 - flood -frag -p 80 -S /target/
	TCP ACK Flood	Specifies the number of data bytes to send and splits packets into more fragments. Moreover, it sets the ACK TCP flag. The source address is spoofed as being a random IP address and targets a specific port.	hping3 -flood -d 3000 -frag -rand-source -p 80 -A /target/
	TCP RST Flood	Runs in flood mode. It specifies the number of data bytes to send and splits the packets into more fragments. Finally it sets the RST TCP flag.	hping3 -flood -d 3000 -frag -rand-source -p 80 -R /target/
	TCP XMAS Flood	Sets the most common TCP flags, namely FIN, SYN, RST, PUSH, ACK, URG, XMAS, and YMAS. It targets a specific port and randomises the source address. Finally, it specifies the number of data bytes to send.	hping3 -flood -d 3000 -rand-source -p 80 -F -S -R -P -A -U -X -Y /target/
	UDP Flood	Sends UDP packets to the target host. It runs in flood mode and randomises the source IP address. It targets a specific port and fills the first signature length bytes of data with signature Flood.	hping3 -flood -rand-source -udp -sign 'Flood' -p 80 /target/

Table A3: The commands and configurations used to deploy the Nmap scans and DoS attacks