A Supervised Intrusion Detection System for Smart Home IoT Devices

Eirini Anthi*, Lowri Williams, Małgorzata Słowińska, George Theodorakopoulos, Pete Burnap
Cardiff University, School of Computer Science & Informatics, 5 The Parade, Roath, Cardiff, CF24 3AA

***Abstract*—The proliferation in Internet of Things (IoT) devices, which routinely collect sensitive information, is demonstrated by their prominence in our daily lives. Although such devices simplify and automate every day tasks, they also introduce tremendous security flaws. Current insufficient security measures employed to defend smart devices make IoT the 'weakest' link to breaking into a secure infrastructure, and therefore an attractive target to attackers. This paper proposes a three layer Intrusion Detection System (IDS) that uses a supervised approach to detect a range of popular network based cyber-attacks on IoT networks. The system consists of three main functions: 1) classify the type and profile the normal behaviour of each IoT device connected to the network, 2) identifies malicious packets on the network when an attack is occurring, and 3) classifies the type of the attack that has been deployed. The system is evaluated within a smart home testbed consisting of 8 popular commercially available devices. The effectiveness of the proposed IDS architecture is evaluated by deploying 12 attacks from 4 main network based attack categories such as: Denial of Service (DoS), Man-In-The-Middle (MITM)/Spoofing, Reconnaissance, and Replay. Additionally, the system is also evaluated against 4 scenarios of multi-stage attacks with complex chains of events. The performance of the system's three core functions result in an F-measure of: 1) 96.2%, 2) 90.0%, and 3) 98.0%. This demonstrates that the proposed architecture can automatically distinguish between IoT devices on the network, whether network activity is malicious or benign, and detect which attack was deployed on which device connected to the network successfully.**

***Index Terms*—Internet of Things (IoT), Smart Homes, Networking, Security, Intrusion Detection, Anomaly Detection, Supervised Machine Learning, Classification, Heterogeneity**

I. Introduction

The popularity of Internet of Things (IoT) devices has significantly increased over the past few years. This is due to their ubiquitous connectivity, allowing them to communicate and exchange information with other technologies, their intelligence, and their decision making capabilities to invoke actions [1]. This provides seamless user experiences which significantly enhance people's every day lives, and is demonstrated by how prominent such devices are today. However, 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) [2]. More specifically, CNI concepts such as smart homes, smart cities, intelligent transport, smart grids, and health care systems are heavily dependent on smart technologies and IoT devices. Nevertheless, although these concepts support the tasks of everyday life, their dependency on Information Communication Technology (ICT) and IoT devices come with tremendous security risks [3]. A survey by Synopsys in May 2017 revealed a lack of confidence in the security of medical devices with 67% manufacturers believing that an attack on a medical device is likely to occur within 12 months, and only 17% of manufacturers taking steps to prevent them [4].

The insufficient security measures and lack of dedicated anomaly detection systems for these heterogeneous networks make them vulnerable to a range of attacks such as data leakage, spoofing, disruption of service (DoS/DDoS), energy bleeding, insecure gateways, etc. These can lead to disastrous effects; causing damage to hardware, disrupting the system availability, causing system blackouts, and even physically harm individuals [5], [6]. Therefore, it is clear that the scale of impact of the attacks performed on IoT networks can vary significantly. For example, a relatively simple and seemingly harmless deauthentication attack can cause no significant damage, but if performed on a device with critical significance, such as a steering wheel in a wireless car, it can pose a threat to human life. Consequently, it is obvious that there is a major gap between security requirements and security capabilities of currently available IoT devices. Two of the main reasons that make these devices insecure include restriction in computational power and heterogeneity in terms of hardware, software, and protocols [7]. 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 [8]. As a result, it is not possible to employ complex and robust security measures. Additionally, given the diversity of these devices, it is very challenging to develop and deploy a security mechanism that can endure with the scale and range of devices [9].

A traditional IT security ecosystem consists of static perimeter network defences (e.g. firewalls, IDS), ubiquitous use of end-point defences (e.g. anti-virus), and software patches from vendors. However, these mechanisms cannot handle IoT deployments due to the heterogeneity in devices and of their use cases, and device/vendor constraints [10], [11]. This means that traditional approaches of discovering attack signatures (e.g. honeypots), will be insufficient and/or non-scalable [10]. Furthermore, as IoT devices operate deep inside the network, traditional perimeter defences are inadequate as they can help block external attacks, but they often fail to prevent attacks

*Corresponding author: anthies@cardiff.ac.uk

from internal devices or applications [12]. As the number of IoT devices increases exponentially [13], the number of unknown vulnerabilities and threats also increases, resulting in perimeter defences becoming weaker. Traditional anomaly detection systems are also ineffective within IoT ecosystems, since the range of possible normal behaviours of devices is significantly larger and more dynamic than traditional IT environments. Popular Intrusion Detection Systems (IDS) such as SNORT and Bro, only work on traditional IP-only networks as they are static and use signature-based techniques [10], [14]. Finally, IDSs developed for Wireless Sensor Networks (WSN) would also be ineffective in an IoT ecosystem mainly because of their inability to adapt, their applicability only to a single platform and protocol, and their small and specific range of detection techniques [15], [10]. Despite major security flaws related to IoT, according to Gartner [16] this sector is expected to grow to 20.4 billion devices by 2020. As these technologies have a direct impact on our lives, security and privacy considerations must become a higher priority. There is a need for an IDS to monitor malicious activity or policy violations within a network of heterogeneous IoT devices and subsequently understand their impact.

This paper is motivated by three main points, which align with Zarpelão et al. [17] who provide a comprehensive literature review on the matter. Firstly, the majority of the proposed systems focus on detecting a limited set of attacks; in particular, routing attacks and DoS. In this case, the proposed system aims to identify a larger set of attacks including multi-stage attacks that represent complex combinations of attack behaviour, which is significantly more challenging to detect. Specifically, the IDS presented in this paper is evaluated against 12 popular attacks from 6 categories found within the IoT domain, but also against 4 scenarios of scripted multi-stage attacks with complex chains of events. Secondly, existing literature lack focus on device profiling. Detecting malicious traffic is a challenging task without profiling the 'normal' behaviour of devices connected to the network. Therefore, in this paper, the behaviour of 8 different IoT devices is profiled so that unusual behaviour can be detected, and subsequently, so can cyber-attacks. Thirdly, current IDSs fail to identify the type of attack that has occurred. Without this information, significant human effort is needed to respond to alerts and determine the severity of an attack. However, in this paper, a machine learning approach demonstrates that it is possible to address this limitation by not only automatically distinguishing between benign and malicious network traffic, thus detecting whether an attack has been deployed, but also to automatically identify the type of the attack that has occurred and against which device. These two factors provide crucial information that can help determine the severity of the cyber-attack, and subsequently accelerate the launch of countermeasures to defend against it. Thus, these features are implemented as part of the proposed IDS. The experiments conducted in this paper show that the performance of the system's three core functions result in an average F-measure of: 1) 99.7%, 2) 97.0%, and 3) 99.0%. This demonstrates that the proposed architecture can automatically distinguish between IoT devices on the network, whether network activity is malicious or benign, and detect which attack was deployed on which device connected to the network successfully.

To the best of our knowledge, the architecture of the IDS proposed here is novel and addresses most of the aforementioned limitations of the existing systems. The main contributions of the work presented in this paper are:

- A three layer architecture for a lightweight, standalone IDS tailored towards IoT devices within a smart home network.
- An investigation into which attributes best represent packets as features in the context of supervised learning, so that devices, maliciousness, and attacks can automatically be identified.
- Resources that can further support research into automating IoT-based cyber-attack detection, such as benign and malicious network activity datasets and a set of scripts for automatically deploying attacks.

II. Related Work

A. Signature/Event/Rule based IDSs

Several studies revolving around IoT security have attempted to design IDS systems tailored specifically for the IoT ecosystem. Stephen and Arockiam [18] suggest a lightweight, hybrid, and centralised approach aiming to detect 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 number of packets received and transmitted to validate the Intrusion Ratio (IR) by the IDS agent. Raza et al. [19] implemented a real-time IDS for the IoT called SVELTE. This system consists of a 6LoWPAN Mapper (6Mapper), 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. owever, it has only been tested to detect spoofed or altered information, sinkhole, and selective-forwarding attacks. Shreenivas et al. [20], [21] 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 [22] 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 [23] 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 [24] 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 [21]. The system is evaluated by deploying four attacks and demonstrates a very low false-positive rate.

Midi et al. [15] 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 [21]. Results showed that the system had a high accuracy in detecting mainly DoS and routing attacks. Furthermore, Thanigaivelan et al. [25] 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. [26], 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. [27] proposed a hybrid lightweight signature-based IDS, in an attempt to mitigate two variations of denial of service attacks; "Hello" flood and version number modification. However, although their results look promising, their system is tested in a simulated environment using Cooja

B. Machine Learning IDSs

Amouri, Alaparthy, and Morgera [28] developed an IDS for IoT networks by applying supervised machine learning. The IDS attempts to profile the benign behaviour of the nodes and identify any anomalies on 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. [29], also employ machine learning algorithms in IoT networks to detect Distributed Denial of Service (DDoS) attacks. They show that by focusing on IoT-specific network behaviors (e.g., limited number of endpoints and regular time intervals between packets) to inform feature selection results in high accuracy of DDoS detection in IoT network traffic with a variety of machine learning algorithms. Nevertheless, they experiments solely focus on this type of attack. Additionally, Shukla [30] 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. Nonetheless, results of this work are promising, the evaluation of the proposed IDS was based on a simulation and the effectiveness of the IDS has not been tested against other attacks.

Meidan et al.[31] and McDermott et al. [32] both focus on the detection of botnets in the IoT ecosystem and employ deep learning techniques 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. [33] review the security threats in IoT networks and discuss a potential security solution which employs machine learning to detect and mitigate attacks using polymorphic software and hardware. However, no description of the experimental setup, implementation, and subsequently, evaluation of the proposed system is provided. Brun et al. [34] designed a deep learning-based approach using dense Random Neural Networks for the detection of network attacks. Although this approach often successfully detects attacks, the system was evaluated on a testbed consisting of only 3 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, limiting the attack space.

C. Attack Type Classification

Few approaches to classifying attack types currently exist. Such approaches, however, have only been employed and evaluated in traditional networks. Therefore, as these approaches were not designed to consider the specific requirements and computational capabilities of IoT, it is challenging to employ them in such environments. Bolozoni et al. [35] propose a machine learning approach to classify the difference types of cyber-attacks detected by Alert Based Systems (ABS). 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 traditional systems, such approach relies on the alerts produced by the ABS, which are not effective in IoT environments, for reasons discussed in Section I. Additionally, as the detection method uses payload values to detect attacks, attacks which IoT systems are vulernable to and which do not alert the payload (e.g. DoS) are not detected. Subba et al. [36] implemented a model that uses feed forward and the back propagation algorithms to detect and classify cyber-attacks in desktop networks. However, to evaluate their system they used the NSL-KDD dataset and attempted to classify probe, DoS, User to Root, and Remote to User attack. Nevertheless, there is no evidence that this system would be as effective if deployed in a heterogeneous IoT environment, which consists of many more protocols, devices, and network behaviours.

To summarise these approaches, Table I shows existing IDSs for IoT and categorises them according to detection method, security threat, validation strategy, and attack type classification. 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, and not classify 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	Attack Type Classification
Stephen & Arockiam [18]	Hello Flood/ Sybil	Packet Metrics	-	-
Raza et al. [19]	Sinkhole & Selective forwarding	Hybrid	Simulation	-
Shreenivas et al. [20]	Routing attacks against RPL protocol	Hybrid	Simulation	-
Pongle & Chavan [22]	Wormhole	Anomaly-based	Simulation	-
Jun & Chi [23]	-	Specification-based	-	-
Summerville et al. [24]	Worm propagation, SQL code injection, and directory traversal	Anomaly-based	Empirical (2 devices)	-
Midi et al. [15]	ICMP flood, Replication, Smurf	Hybrid	Empirical (2 devices)	-
Thanigaivelan et al. [25]	-	Anomaly-based	-	-
Oh et al. [26]	Routing Attacks	Signature-based	Empirical (1 device)	-
Shukla [30]	Wormhole	Machine Learning	Simulation	-
Doshi et al. [29]	DDoS	Machine Learning	Empirical (2 devices)	-
Amouri et al. [28]	Identifies Malicious Nodes	Machine Learning	Simulation	-
McDermott et al.[32]	Botnets	Machine Learning	Simulation	-
Meidan et al. [31]	Botnets	Machine Learning	Empirical (9 devices, 3 types: doorbell, camera, thermostat)	-
Restuccia et al. [33]	-	Machine Learning	-	-
Brun et al. [34]	UDP Flood, TCP SYN, Sleep Deprivation Attack, Barrage Attack, and Broadcast Attack	Deep Learning	Empirical (3 devices)	-
Proposed system	various reconnaissance (quick scan, intense scan, etc.) iot-scanner, various DoS (tcp/udp/hello flood), various man-in-the-middle (ettercap, ARP) , replay attack, ARP & DNS spoofing, 4 multi-stage scripts	Machine Learning	Empirical (8 devices, 6 types: plugs, cameras, hubs, sensors, voice controlled, lamps)	yes

TABLE I: Summary of current work on Intrusion Detection Systems for Internet of Things

III. Methodology

A. System Overview

Figure 1 provides an overview of the proposed IDS architecture. Specifically, the first layer of the tool will scan the network, identify the connected IoT devices based on their MAC addresses, and classify them based on their network behaviour. At the second layer, the packets from such devices are classified as whether they are benign or malicious. Finally, if malicious packets have been detected in the second layer, the third layer will classify these malicious packets as one of four main attack types. As a result, in an event of an attack, the output of the system is: 1) the MAC address of the device under attack, 2) whether the packet is malicious, and 3) the type of attack which has occurred, which is one of the four main categories that the model was trained on.

B. IoT Smart Home Testbed

According to Cisco's VNI report [37], in 2017, the average household in North America, Western Europe, and Central and Eastern Europe has on average 8, 5.4, and 2.5 smart devices respectively. The testbed used to support the experiments provided in this paper consists of 8 commercially popular IoT devices; and thus is a representative example of a traditional smart home. Such devices included the Belkin NetCam camera, TP-Link NC200 Camera, TP-Link Smart Plug, Samsung Smart Things hub, Amazon Echo Dot, British Gas Hive connected to two sensors: a motion sensor and a window/door sensor, and Lifx Lamp. Additionally, a laptop was also connected to the network to perform two tasks: 1) continually record the network traffic and automatically generate and save the log files, and 2) deploy various network based attacks. Figure 2 displays the architecture of the smart home testbed.

IoT device	Type	Protocol(s)
Amazon Echo Dot	Multimedia	Ethernet
Belkin NetCam	Multimedia	WiFi
TP-Lik NC200	Multimedia	WiFi
Hive Hub	Sensors	Ethernet & ZigBee
Samsung Smart Things Hub	Sensors	Ethernet & BLE
TP-Link SmartPlug	Sensors	WiFi
Apple TV	Multimedia	WiFi
Lifx Smart Lamp	Lamp	WiFi & ZigBee

TABLE II: IoT devices included in the smart home testbed

In order to collect the network traffic from the IoT testbed, *tcpdump* was scheduled to run on the access point (P1) as shown in the same Figure. The collected PCAP logs were then transferred and stored in the syslog server.

C. Data Collection

1) Benign Network Data: To conform to other comparable research (e.g. [38]), 3 weeks worth of benign data and 2 weeks of malicious data was collected from the IoT testbed. The testbed described in Section III-B was designed and implemented so that all the packets on the network (local-to-local or local-to-remote) were captured. All the inbound and

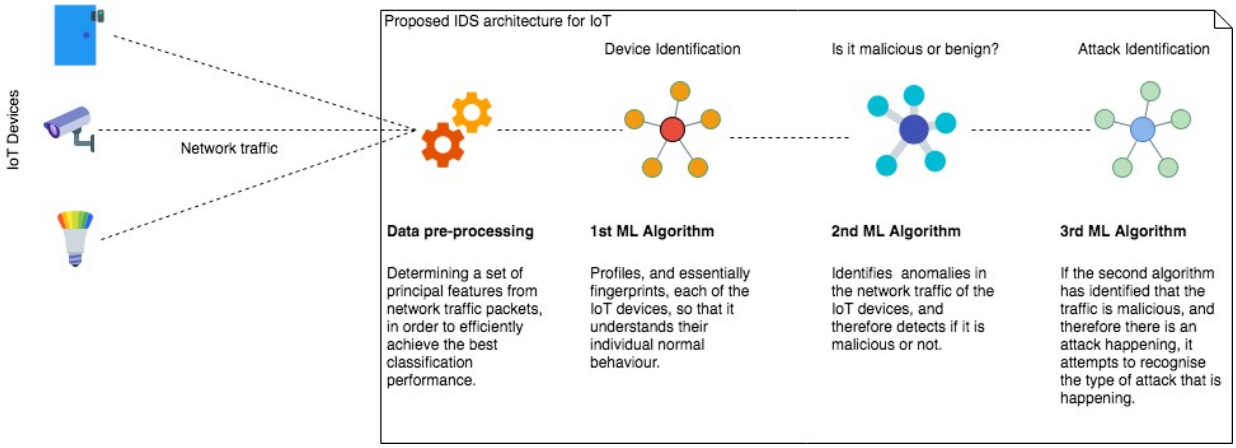

Fig. 1: Overview of the proposed architecture for the three layer Intrusion Detection System.

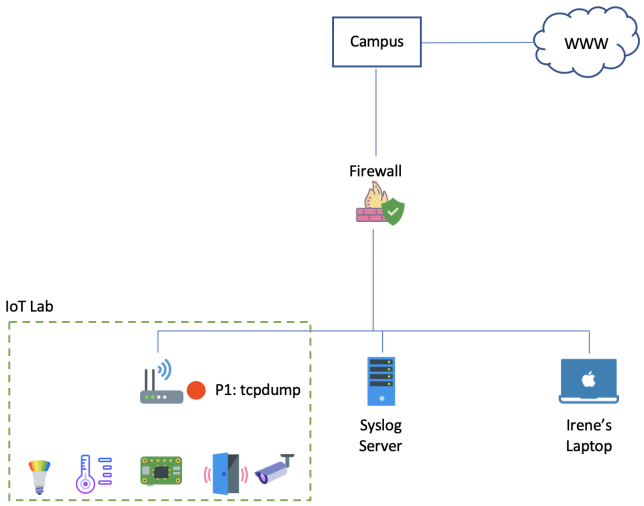

Fig. 2: IoT smart home testbed network architecture.

outbound traffic from the smart devices was captured using the tcpdump [39] tool, which was continually running on the Access Point (indicated in Figure 2 with red circular marker). The data collection process was automated using Cron jobs and bash scripts. Data frames were continuously captured and saved to the Syslog Server in a pcap format. Files were generated in one-minute intervals and were accessed remotely by using Secure Shell (SSH) to connect to the Syslog Server. For the purposes of benign data collection, pcap files were automatically transferred and merged to the Syslog Server using Cron jobs that invoke a series of bash scripts.

2) Cyber-Attacks in IoT ecosystems: Multiple studies (e.g. [22], [40], [41], [15]) have demonstrated that IoT devices are vulnerable to a wide range of attacks including network attacks, physical attacks, firmware attacks, and data leakage. Some of the reasons why such devices are insecure include: limitations in computational power, lack of transport encryption, insecure web interfaces, lack of authentication/authorisation mechanisms, and their heterogeneity which makes applying uniform security mechanisms extremely challenging [42]. Consequently, several IoT attack categories have emerged:

- **Denial of Service (DoS)**: aims to make IoT devices unavailable to its intended users by temporarily or indefinitely disrupting their services [43].
- **Distributed Denial of Service (DDoS)/Botnets**: an attacker aims to compromise a large amount of vulnerable IoT devices in order to be able to deploy a significantly more severe DoS or other attacks [44].
- **Man-In-The-Middle**: compromises the communication channel between the IoT device and the intended recipient of its data. Once the connection is compromised, the attacker is able to act as a proxy and therefore read, insert, and modify the transmitted data [45].
- **Spoofing**: manipulates fake identities to compromise the effectiveness of the IoT device by forging a large number of identities to act as legal nodes [8].
- **Insecure Firmware**: compromises user data, control over the IoT device, and attacks against other devices [42].
- **Data Leakage**: Many IoT devices suffer from the lack of transport encryption. This can result in data loss, and depending on the data exposed, could lead to complete compromise of the device or user accounts [42].

Acquiring 3 weeks of malicious activity required the design and deployment of a range of malicious attacks. The machine designated to run the malicious attacks was a Lenovo Thinkpad configured to run the Kali Linux operating system [46]. Although most IoT devices are connected to the Internet via WiFi, they also support other communication protocols such as Ethernet, IEEE 802.15.4, Bluetooth, ZigBee, Z-Wave, LoRaWAN, and Cellular (GPRS/2G/3G/4G). However, in this paper, WiFi and Ethernet communications are used. Table III demonstrates all the attacks performed and the tools used within this work.

To ensure that the IDS was tested appropriately, it was important to generate a broad data-set, representative of the performed attacks. In particular, it was essential to introduce some randomness to the deployed attacks in order to avoid model overfitting. For this reason, bash scripts were implemented to automate and randomize the malicious attacks. Randomization

Attack Category	Method
Reconnaissance	Nmap (Quick Scan, Intense Scan, etc.), iot-scanner
DoS/DDoS	TCP Flood/UDP Flood, Hello flood attacks
MITM	Ettercap, SSL Strip, Burpsuit
Replay	mitmframework suite
Spoofing	DNS, ARP

TABLE III: Cyber-attacks that were deployed on the IoT testbed

was achieved by implementing a timer that launched the attacks at random for a random period of time (between 5 seconds and 20 minutes). The idle time in between each attack launch was also randomized using the same principle. For some attacks, such as the iot-toolkit toggle attack, the intensity of the attack (e.g. the amount of malicious packets sent to the device) was also randomized. Moreover, four automated multi-level malicious scenarios were implemented and deployed on the network. This is to increase the complexity of the attacks, but also to represent the steps that a real adversary would follow when attacking the devices.

1) Scenario 1: network scanning
   The attacker performs either one quick scan or two scans, with the second one being a more in-depth and targeted reconnaissance attempt. The script will perform the second attack with probability 0.5. The rationale for this scenario is that the attacker will usually commence their attack with a quick scan to determine available hosts and then decide whether to proceed to a more complex one to search for vulnerabilities if needed.
2) Scenario 2: network scanning & Denial(s) of Service
   This scenario also incorporates a quick scan but the attacker also performs one or more of the most common DoS attacks on the target network. Up to 6 DoS attacks can be performed in a row. Random attack duration as well as random wait times in between the attacks are used. The scenario is targeting a random IP address identified on the predefined network.
3) Scenario 3: network scanning & MITM
   This scenario represents a quick reconnaissance but is followed by a MITM attack performed via ARP spoofing, either with passive monitoring only or also using the packet injection (chosen at probability of 0.5). Random attack times, wait times, as well as random number of injected packets are selected automatically. The MITM is always set in between the access point and one of the IP addresses present on the network (identified at the beginning of the script).
4) Scenario 4: complete attack with iot-toolkit
   An end-to-end automation of the iot-toolkit attacks from the previously described framework. It targets the TP-Link devices for reconnaissance and performs toggle/get_info on the TP-Link smart plug. Again, random duration, intensity, and wait times are selected automatically.

Another crucial concept that was considered during the development of the scripts was to generate logs of when each type and variation of attack took place. This was necessary for further labeling tasks needed for supervised machine learning, and for validation that the attacks worked as expected. A general log was generated to provide an outline of the dates and types of attacks performed. Additionally, logs of all the outputs generated during the attacks (including output returned by the tools) were created for debugging purposes.

D. Feature Selection

The main requirements to consider when developing a machine learning based IDS for IoT are:

- Lightweight - not require considerable computational processing power.
- Stand-alone - not dependent on any other software or alert based system.
- Fast - malicious activity must be detected in almost real time to reduce impact.
- To work over encrypted traffic - most commercial IoT devices employ transport encryption.

Given the above requirements, it was decided to initially investigate whether it is possible to detect malicious behaviour from single packets. The reasoning behind this approach is that, as single packets are the smallest piece of network information, they are quicker to processes, and subsequently improve the speed of identifying malicious activity.

The raw PCAP files containing the network packets were initially converted and represented in a Packet Description Markup Language (PDML) [47] format. PDML conforms to the XML standard and contains details about the packet dissection/layers. As a result, it allows access to all the packet attributes that can be used as features. A network packet consists of a series of layers (Physical, Data Link, Network, Transport, and Application), each layer being a child of the previous layer, built from the lowest layer up [48] (see Figure 3). Each layer, has its own header composed of a range of different fields providing information, and a payload. For the classification experiments discussed in this work, all the fields that compose each of the aforementioned layers were extracted, in order to investigate which ones are most relevant in detecting benign and malicious behaviour on IoT.

In addition to these attributes, few more fields were also included such as: frame information [39] and *packet_type* - which specifies whether the data packet was inbound or outbound to an IoT device on the testbed. Additionally, features that represented identifying properties were removed (e.g. source IP address, time, packet ID) to ensure 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, because the network traffic is encrypted, the payload information from the Application Layer was not considered as a feature. In total, 121 features were extracted from each packet and represented as a feature vector (see Table IX in Appendix A).

E. Data Labeling

Supervised machine learning requires labelled training data. In this paper, 3 classification experiments were conducted for each dataset: (1) device type classification, (2) malicious

Fig. 3: An example of how layers are structured within a packet.

packet detection classification, and (3) attack type classification.

For (1) and (2), it was detected that the IP address of the IoT devices on the testbed would change repeatedly under specific attacks. IP addresses were therefore not suitable indicators to associate a class label to packets. The MAC addresses of such devices were therefore used to associate packets. For (3), as attacks were systematically performed, packets were labelled as their attack type upon completion. To ensure that the labeling of the malicious packets was implemented as accurately as possible, two parameters were considered: the launch time of the attack and the MAC address of the attacker's machine. As a result, every time that an attack was launched, we noted the exact time and associated it with the MAC address of the laptop used to deploy it. Therefore any packets with a time-stamp within a specific attack time frame that also had the attacker's MAC address, were labeled as malicious. Finally, on the attacker's machine services/applications such as mail and web browsers were deactivated, in order to avoid mislabeling any benign packets from the same machine as malicious. The class labels for each classification experiments are as follows:

- (1): Amazon Echo Dot, Belkin Net, TP-Lik NC200, Hive Hub, Samsung Smart Things Hub, TP-Link SmartPlug, Lifx Smart Lamp, Firewall, Access Point.
- (2): A packet was labeled as malicious if it was collected during an attack which targeted a device on the IoT testbed. The packet was labeled as benign if otherwise.
- (3): DoS, MITM, Scanning, iot-toolkit.

Figures 4 - 6 show the distribution of packets across all classes for each experiment.

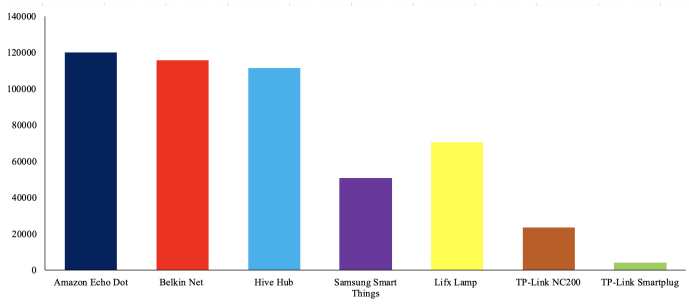

Fig. 4: Distribution of packets across IoT devices

F. Class Balancing and Sample Size Reduction

An uneven balance of class labels across each classification experiment (Figures 4 - 6) has the potential to negatively affect classification performance. Additionally, datasets containing a significantly large number of packets such as those produced here, require high computational power and processing time when applying machine learning algorithms.

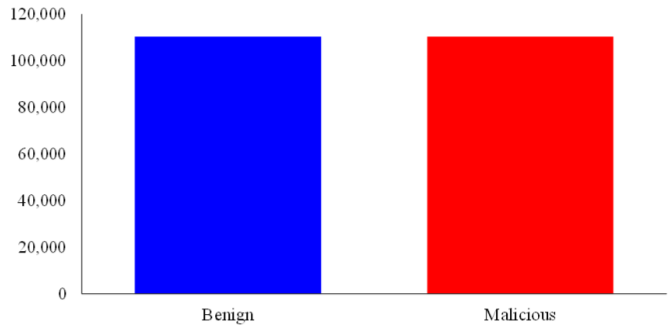

Fig. 5: Distribution of packets across attack detection

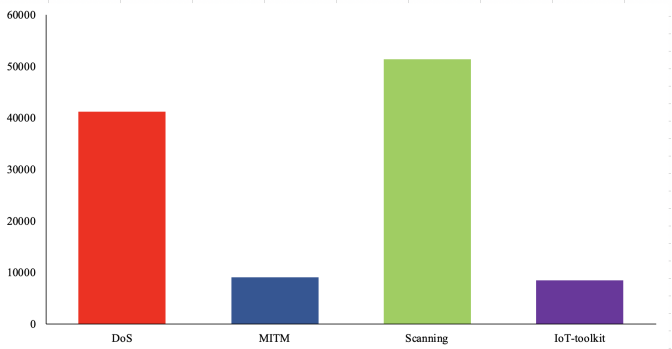

Fig. 6: Distribution of packets across attack types

Weka [49], a popular suite of machine learning software, was used to support classification experiments. Given the significant uneven balance across the datasets and the significantly large number of packets to be classified, the spread subsampling and class balancing filters available in Weka were applied to generate a random subsample of packets and to subsequently balance the distribution of classes within those samples.

For device type classification, the sample size was acquired at random from a total of 2,004,657 packets. The final sample size was 10,000 packets, with 1,000 packets per device. For detecting whether an attack is malicious or not, the dataset was sampled at random from a total of 220,785 packets to contain 80,000 packets (40,000 and 40,000 of benign and malicious packets respectively). Finally, for classifying the type of attack, the final sample size was set to acquire a sample of 50,000 packets (10,000 packets per attack) from a total of 220,785 packets.

IV. Algorithm Selection and Classification Experiments

To explore how well classification algorithms can learn to profile IoT devices on the network, detect wireless attacks, and classify the type of such attacks, the performance of supervised machine learning when the corresponding network activity data was used to train and evaluate the classification model.

In the case of identifying whether a packet is malicious or benign, classification is 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.

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 identifications was correct, whereas recall (R) measures what proportion of malicious packets were identified correctly. The two measures are often used together in F-measure (F), which calculates the harmonic mean of precision and recall, and provides a single weighted metric to evaluate the overall classification performance. Such measures are calculated using equations in Equation 1.

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN}, \quad F = 2 \cdot \frac{P \cdot R}{P + R} \quad (1)$$

Others use accuracy as a measure of performance. Accuracy 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.

In order to perform classification experiments, a random subset of 60% of each balanced dataset described in Section III-F were selected for training, with the remaining 40% used for testing. The "no free lunch" theorem suggests that there is no universally best learning algorithm [50]. In other words, the choice of an appropriate algorithm should be based on its performance for that particular problem and the properties of data that characterize the problem. In this case, a variety of classifiers distributed as part of Weka were evaluated.

To comply with other IDSs which employ machine learning techniques to detect cyber-attacks in the traditional and IoT networks (e.g. [51], [52]), 9 classifiers were selected based on their ability to support multi-class classification, high-dimensional feature space, and the time it takes for the classification model to classify unseen data. 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. J48 Decision Tree, Support Vector Machine). 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.

V. Results and Discussion

Table IV reports the overall weighted-averaged performance for all 9 classifiers, including their classification time. Overall, Weka's implementation of J48 decision tree method [53] with pruning achieved the best performance, resulting in an F-measure of 99.7%, 97.0%, and 99.0% and a classification time of 0.1 seconds, 0.4 seconds, and 0.2 seconds for each experiment respectively.

To ensure that the J48 classifier is not over-fitting, we performed additional experiments which result 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. Two main feature selection methods were used to identify the most relevant features; Correlation Attribute Evaluation Filter and Gain Ratio Attribute Evaluation Filter provided in Weka. The latter evaluates the worth of an attribute by measuring the correlation between it and the class and the former evaluates the worth of an attribute by measuring the information gain with respect to the class. Results showed, that from 121 features (Figure 8, 7), 10 were ranked to have the highest correlation within the feature space. Further classification was performed using only the highly correlated features are present.
- 10-fold cross validation experiments.

Figures 7 and 8 show the features among the top 10 which affect the decision tree: icmp fields, IP and TCP flags, packet and frame length, and TCP destination port. Specifically, when present, ICMP code options such as fragment protection and packet protection can indicate a DoS attack. Moreover, 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, 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) or ICMP packet transmission. This ultimately can be considered as being a type of DoS as they make the device unavailable. 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. Packet length is also an indicator of malicious behaviour, specifically when the packet is significantly larger or smaller than usual.

	Device profiling				Detect wireless attacks				Attack type			
Classifier	**P**	**R**	**F**	**Time (sec)**	**P**	**R**	**F**	**Time (sec)**	**P**	**R**	**F**	**Time (sec)**
Naive Bayes	79.0	57.0	65.0	40.4	93.0	93.0	93.0	1.5	92.0	91.0	91.0	6.2
Bayesian Network	96.0	96.0	96.0	28.3	96.3	96.4	96.4	1.2	96.5	96.5	96.0	3.1
J48	**98.8**	**98.0**	**98.0**	**41**	**97.0**	**97.0**	**97.0**	**0.4**	**99.0**	**99.0**	**99.0**	**0.2**
Zero R	17.0	17.0	17.0	0.2	29.0	31.0	49.0	0.6	50.0	50.0	50.0	0.2
OneR	79.0	84.0	87.0	20	93.0	93.0	93.0	0.2	92.0	92.0	92.0	0.2
Simple Logistic	96.0	96.0	96.0	65.0	97.0	97.0	97.0	46.0	96.4	96.5	96.5	45.0
Support Vector Machine	89.0	70.0	82.0	>30mins	N/A	N/A	N/A	>30mins	N/A	N/A	N/A	>30mins
Multi-Layer Perceptron	N/A	N/A	N/A	>30mins	N/A	N/A	N/A	>30mins	N/A	N/A	N/A	>30mins
Random Forest	96.0	96.0	96.0	2.24mins	N/A	N/A	N/A	>30mins	N/A	N/A	N/A	>30mins

TABLE IV: Weighted average of the results of all nine classifiers, following 60-40 percentage split testing for all three experiments.

```
0.7539337098592495      20 ip.flags.df
0.670969816858671       18 ip.flags
0.5453674374931726      28 tcp.stream
0.5267887954178639      42 tcp.flags.syn
0.519048422963477       39 tcp.flags.ack
0.4431280884191186      40 tcp.flags.push
0.36192646613589197     56 icmp.code
0.31261694599677714     21 ip.flags.mf
0.3015532160783864      23 ip.ttl
0.24184030347824367     27 tcp.dstport
```

Fig. 7: Top 10 features following correlation attribute filtering

```
0.41522167265    20 ip.flags.df
0.32329326427    18 ip.flags
0.2383259639     42 tcp.flags.syn
0.22979275587    39 tcp.flags.ack
0.22300989862    21 ip.flags.mf
0.21955115226    22 ip.frag_offset
0.19106985295     6 frame.cap_len
0.19106985295     2 caplen
0.19106985295     5 frame.len
0.19106985295     1 len
```

Fig. 8: Top 10 features following gain ratio attribute filtering

To gain a better insight into the performance of the classifier across the experiments, the confusion matrices in Tables V-VII, which show how the predicted classes for individual packets compare against the actual ones, were analysed.

When profiling devices, the classifier demonstrated a high percentage of correct predictions, thus less often misclassifying devices. For example, Lifx Smart Lamp, Samsung Smart Things Hub, and Belkin Net demonstrated few confusion and were generally correctly classified. This may be explained by the fact that such devices are distinct, and thus, 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 the tasks they exist to perform are different. In this case, a feature within the TP-Link NC200 packets include the connectionless protocol, User Datagram Protocol (UDP), whereas the TP-link SmartPlug packets use Transmission Control Protocol (TCP). However, in some cases, confusion often occurred where Belkin Net and Hive Hub, were misclassified. 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 wireless attacks demonstrated very little confusion. This could be explained by the fact that the attacks that were performed during data collection were off-the-shelf attacks, i.e. resources which include attacks that are freely available, such as *hping, nmap, iot-toolkit*, etc., and are unsophisticated. In this case, the features of malicious and benign packets are distinct, and thus, few classification confusions occurred. For instance, malicious packets may contain different flag values which indicate an attack has occurred as explained earlier.

A. Experiments using Unseen Validation Datasets

To evaluate the performance of the trained models generated in Section V even further, the trained classifiers were applied to unseen datasets. Such datasets included packets that were collected in Section III-F, but were not included as part of the sample set used to originally train and test the classifiers.

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

			Predicted						
			a	**b**	**c**	**d**	**e**	**f**	**g**
Actual	Amazon Echo Dot	**a**	1,647	0	2	3	0	0	0
	Belkin Net	**b**	1	1,647	0	1	0	1	0
	Hive Hub	**c**	0	304	1,389	4	0	0	2
	Samsung Smart Things Hub	**d**	0	0	0	1,679	0	0	1
	Lifx Smart Lamp	**e**	0	0	0	0	1,679	0	0
	TP-Link NC200	**f**	0	0	0	0	0	1,610	9
	TP-Link SmartPlug	**g**	0	0	0	0	0	0	1,677

TABLE V: Device type confusion matrix which demonstrates how the predicted classes for individual packets compare against the actual ones

			Predicted	
			a	**b**
Actual	Malicious	**a**	43,967	26
	Benign	**b**	4	44,317

TABLE VI: Attack detection confusion matrix which demonstrates how the predicted classes for individual packets compare against the actual ones

			Predicted			
			a	**b**	**c**	**d**
Actual	DoS	**a**	3,392	12	0	0
	MITM	**b**	0	3,484	12	0
	Scanning	**c**	0	15	3,427	0
	iot-toolkit	**d**	0	61	15	3,334

TABLE VII: Identifying attack type confusion matrix which demonstrates how the predicted classes for individual packets compare against the actual ones

dataset contained 436 packets, 109 packets for each of the four attacks.

As shown in Table VIII, the results demonstrate that for the device type classification and for identifying malicious packets, the accuracy of the classifiers dropped notably (from 98.8% to 96.2% and from 97.0% to 90% respectively). However, the performance of the classifier in distinguishing the types of attacks, did not change significantly (from 99.0% to 98%).

Device profiling			Detect wireless attacks			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 VIII: Classification performance for each experiment on unseen validation data using the trained J48 models

To conclude, the key insights of these results are:

- Decision trees (in particular, J48) seem to be the best algorithm for this task as it achieved the best classification results across all three experiments.
- IP and TCP flags are the most important features.
- For device classification, the confusion matrix indicated that the classifier less often misclassified devices.
- For detecting malicious packets, the confusion matrix indicated that the classifier also demonstrated very little confusion.
- The high accuracy of the classifier can be explained by the fact that the deployed attacks were not sophisticated and deployed using out of the self tools. As a result, the traffic and network behavior during these significantly changes.
- When unseen validation datasets are used to further evaluate classification performance, the accuracy notably dropped for device type classification and detecting malicious packets. Though, it did not change significantly when distinguishing attack types.

B. Use Case

The main use case for the IDS proposed in this paper is to be able to detect real time malicious behaviour in smart home IoT devices 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 in this paper, 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 device 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.

VI. Conclusion

In this paper, a novel and intelligent architecture of a three layer IDS is presented. To address the aforementioned limitations of current systems, the IDS presented here includes three main functions: 1) classify the type and profile the normal behaviour of each IoT device connected to the network, 2) detect wireless attacks deployed against the connected IoT devices, and 3) classify the type of the attack that has been deployed. In order to evaluate the performance of applying a supervised machine learning approach to automate each function, network activity data from a real testbed consisting of a variety of commercially available and popular IoT devices was collected. The performance of the systems three core functions result in an F-measure of: 1) 96.2%, 2) 90.0%, and 3) 98.0%. This demonstrates that the proposed architecture can successfully distinguish between IoT devices on the network, whether network activity is malicious or benign, and detect which attack was deployed on which device connected to the network automatically.

In addition to the experimental results, this study provides resources that can further support research into automating IoT-based cyber-attack detection. Such resources include raw PCAP files and flow information for benign and malicious network activity, a set of scripts to automatically launch attacks from five main network attack categories discussed in this paper, and further scripts to automatically launch multi-level attacks which represent the behaviour of an attacker to create an authentic malicious dataset. All resources are freely available to the research community to support further investigations into several aspects of IoT. The scripts are available to download here: https://goo.gl/iCJ525, and https://goo.gl/anB6eU. Due to the vast size of the data collected in this paper, it can be accessed on request by contacting the corresponding author (anthies@cardiff.ac.uk).

VII. Future Work

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 even Industrial IoT environment. This will allow the system to be further evaluated on more complex and more 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.

References

[1] Rafiullah Khan, Sarmad Ullah Khan, Rifaqat Zaheer, and Shahid Khan. Future internet: the internet of things architecture, possible applications and key challenges. In *Frontiers of Information Technology (FIT), 2012 10th International Conference on*, pages 257–260. IEEE, 2012.

[2] Tobby Simon. Chapter seven: Critical infrastructure and the internet of things. *Cyber Security in a Volatile World*, page 93, 2017.

[3] Eirini Anthi, Lowri Williams, and Pete Burnap. Pulse: An adaptive intrusion detection for the internet of things, 2018.

[4] Cybersecurity executive: Medical devices a 'bulls-eye' for cyber-attacks. https://www.digitalhealth.net/2017/12/medical-device-functionality-vs-cybersecurity/. (Accessed on 02/05/2018).

[5] 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.

[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 02/05/2018).

[7] Securing the internet of things: A proposed framework - cisco. https://www.cisco.com/c/en/us/about/security-center/secure-iot-proposed-framework.html. (Accessed on 07/13/2018).

[8] Liang Xiao, Xiaoyue Wan, Xiaozhen Lu, Yanyong Zhang, and Di Wu. Iot security techniques based on machine learning. *arXiv preprint arXiv:1801.06275*, 2018.

[9] 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.

[10] 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.

[11] Michael Vögler, Johannes Schleicher, Christian Inzinger, Stefan Nastic, Sanjin Sehic, and Schahram Dustdar. Leonore–large-scale provisioning of resource-constrained iot deployments. In *Service-Oriented System Engineering (SOSE), 2015 IEEE Symposium on*, pages 78–87. IEEE, 2015.

[12] The limit does not exist: Why defending the perimeter is not feasible in the iot - 2018-03-04 - page 1 - rfid journal. http://www.rfidjournal.com/articles/view?16805. (Accessed on 03/29/2018).

[13] 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.

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

[15] Daniele Midi, Antonino Rullo, Anand Mudgerikar, and Elisa Bertino. Kalisa 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.

[16] Gartner says 6.4 billion connected "things" will be in use in 2016, up 30 percent from 2015. https://www.gartner.com/newsroom/id/3165317. (Accessed on 07/13/2018).

[17] Bruno Bogaz Zarpelão, Rodrigo Sanches Miani, Cláudio Toshio Kawakani, and Sean Carlisto de Alvarenga. A survey of intrusion detection in internet of things. *Journal of Network and Computer Applications*, 84:25–37, 2017.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

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

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

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

[30] 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.

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

[32] 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.

[33] Francesco Restuccia, Salvatore DOro, 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.

[34] 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.

[35] 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.

[36] 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.

[37] Cisco visual networking index: Forecast and trends, 20172022 white paper - cisco. 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).

[38] Arunan Sivanathan, Daniel Sherratt, Hassan Habibi Gharakheili, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. Characterizing and classifying iot traffic in smart cities and campuses. In *Proc. IEEE INFOCOM Workshop SmartCity, Smart Cities Urban Comput.*, pages 1–6, 2017.

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

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

[41] 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.

[42] Owasp internet of things project - owasp. https://www.owasp.org. (Accessed on 05/31/2018).

[43] 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.

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

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

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

[47] Pdml - the wireshark wiki. https://wiki.wireshark.org/PDML. (Accessed on 03/27/2019).

[48] Scapy p.04 looking at packets — thepacketgeek. https://thepacketgeek.com/scapy-p-04-looking-at-packets/. (Accessed on 05/14/2019).

[49] Weka 3 - data mining with open source machine learning software in java. https://www.cs.waikato.ac.nz/ml/weka/. (Accessed on 06/03/2018).

[50] David H Wolpert. The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390, 1996.

[51] 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.

[52] 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.

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

Eirini Anthi received a First Class Honours BSc degree in computer science from Cardiff University, UK, in 2016 and is currently working towards the Ph.D. degree in the field of Cyber Security at the same University. Her research revolves around the security and privacy of Internet of Things devices (IoT). Specifically, her work examines the security issues that come along with these devices and tries to identify methods to make them more secure.

Lowri Williams received her Ph.D in Computer Science from Cardiff University, UK, in 2018. Her research interests include natural language processing, sentiment analysis, data mining, machine learning, and language resources.

Małgorzata Słowińska received a First Class Honours BSc degree in computer science with security and forensics from Cardiff University, UK, in July 2018. Her interests lie within the fields of cyber defense, security of Internet of Things (IoT) and quantum cryptography. She currently works as a Cyber Security Analyst for one of the Big 4 companies, helping clients understand and enhance their information and cyber security controls.

George Theodorakopoulos received the Diploma degree from the National Technical University of Athens, Greece, in 2002, and the M.S. and Ph.D. degrees from the University of Maryland, College Park, MD, USA, in 2004 and 2007, all in electrical and computer engineering. He is a Senior Lecturer at the School of Computer Science & Informatics, Cardiff University, since 2012. From 2007 to 2011, he was a Senior Researcher at the Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland. He is a coauthor (with John Baras) of the book Path Problems in Networks (Morgan & Claypool, 2010).

Pete Burnap is a Professor at Cardiff University and is seconded to Airbus Group to lead Cyber Security Analytics Research heading projects involving the application of Artificial Intelligence, Machine Learning and Statistical Modeling to Cyber Security problems (most recently malware analysis). Pete obtained his B.Sc. in Computer Science in 2002 and his Ph.D: Advanced Access Control in support of Distributed Collaborative Working and De-perimeterization in 2010, both from Cardiff University. He has published more than 60 academic articles stemming from funded research projects worth over 8m and has advised the Home Affairs Biographical Sketch Select Committee, Home Office and Metropolitan Police on sociotechnical research outcomes associated with cyber risk and evolving cyber threats.

APPENDIX

Features	
len	icmp.resp$_in$
caplen	icmp.resp$_to$
frame.encap$_type$	data.len
frame.offset$_shift$	ssl.record.content$_type$
frame.len	ssl.record.version
frame.cap$_len$	ssl.record.length
frame.marked	arp.hw.type
frame.ignored	arp.proto.type
eth.lg	arp.hw.size
eth.ig	arp.proto.size
ip.version	arp.opcode
ip.hdr$_len$	http.response.code
ip.dsfield.dscp	http.content$_length$
ip.dsfield.ecn	http.response
ip.src	http.response$_number$
ip.dst	http.request
ip.len	http.request$_number$
ip.flags	classicstun.type
ip.flags.rb	classicstun.length
ip.flags.df	udp.srcport
ip.flags.mf	udp.dstport
ip.frag$_offset$	udp.length
ip.ttl	udp.checksum.status
ip.proto	udp.stream
ip.checksum.status	dns.flags.response
tcp.srcport	dns.flags.opcode
tcp.dstport	dns.flags.truncated
tcp.stream	dns.flags.recdesired
tcp.len	dns.flags.z
tcp.seq	dns.flags.checkdisable
tcp.nxtseq	dns.flags.rcode
tcp.ack	dns.count.queries
tcp.hdr$_len$	dns.count.answers
tcp.flags.res	dns.count.auth$_rr$
tcp.flags.ns	dns.qry.name.len
tcp.flags.cwr	dns.count.labels
tcp.flags.ecn	dns.resp.type
tcp.flags.urg	dns.resp.class
tcp.flags.ack	dns.resp.ttl
tcp.flags.push	dns.resp.len
tcp.flags.reset	igmp.version
tcp.flags.syn	igmp.type
tcp.flags.fin	igmp.max$_resp$
tcp.window$_size_value$	igmp.checksum.status
tcp.window$_size$	ntp.flags.li
tcp.window$_size_scalefactor$	ntp.flags.vn
tcp.checksum.status	ntp.flags.mode
tcp.urgent$_pointer$	ntp.stratum
tcp.options.nop	ntp.ppoll
tcp.options.mss$_val$	ntp.rootdelay
tcp.options.sack$_perm$	ntp.rootdispersion
tcp.analysis.bytes$_in_flight$	ntp.precision
tcp.analysis.push$_bytes_sent$	bootp.type
tcp.payload	bootp.hw.type
icmp.type	bootp.hw.len
icmp.code	bootp.hops
icmp.ident	bootp.secs
icmp.checksum.status	bootp.flags.bc
icmp.seq	bootp.flags.reserved
icmp.seq$_le$	bootp.dhcp

TABLE IX: Packet features