Received: 11 July 2023 | Revised: 19 January 2024 | Accepted: 20 February 2024

DOI: 10.1002/spy2.386

RESEARCH ARTICLE

WILEY

A lightweight Intrusion Detection for Internet of Things-based smart buildings

Amith Murthy[1] | **Muhammad Rizwan Asghar**[2] | **Wanqing Tu**[3]

[1]School of Computer Science, The University of Auckland, Auckland, New Zealand

[2]Surrey Centre for Cyber Security, University of Surrey, Guildford, UK

[3]Department of Computer Science, Durham University, Durham, UK

Correspondence
Muhammad Rizwan Asghar, Surrey Centre for Cyber Security, University of Surrey, Guildford, UK.
Email: r.asghar@surrey.ac.uk

Abstract

The integration of Internet of Things (IoT) devices into commercial or industrial buildings to create smart environments, such as Smart Buildings (SBs), has enabled real-time data collection and processing to effectively manage building operations. Due to poor security design and implementation in IoT devices, SB networks face an array of security challenges and threats (e.g., botnet malware) that leverage IoT devices to conduct Distributed Denial of Service (DDoS) attacks on the Internet infrastructure. Machine Learning (ML)-based traffic classification systems aim to automatically detect such attacks by effectively differentiating attacks from benign traffic patterns in IoT networks. However, there is an inherent accuracy-efficiency tradeoff in network traffic classification tasks. To balance this tradeoff, we develop an accurate yet lightweight device-specific traffic classification model. This model classifies SB traffic flows into four types of coarse-grained flows, based on the locations of traffic sources and the directions of traffic transmissions. Through these four types of coarse-grained flows, the model can extract simple yet effective flow rate features to conduct learning and predictions. Our experiments find the model to achieve an overall accuracy of 96%, with only 32 features to be learned by the ML model.

KEYWORDS

communication security, cybersecurity issues affecting communications

1 | INTRODUCTION

Internet of Things (IoT) technology has equipped conventional building devices such as lights, switches, CCTV, locks, actuators, and sensors with networking capabilities, so as to collect environment data in a real-time fashion to support automatic building monitoring and management, for example, energy and water management. The convergence of cloud computing with big data analytics has enabled the extraction of actionable insights from the various IoT sensor data available in Smart Buildings (SBs) and such cloud services are leveraged to more efficiently manage building operations.[1] Building automation systems (BAS) are traditionally 'air-gaped' from public access (i.e., the Internet) to shield them from external threats. With the advancement of cloud computing and IoTs, BAS deployments range from cloud servers to

Muhammad Rizwan Asghar and Wanqing Tu contributed equally to this study.

This is an open access article under the terms of the Creative Commons Attribution-NonCommercial-NoDerivs License, which permits use and distribution in any medium, provided the original work is properly cited, the use is non-commercial and no modifications or adaptations are made.
© 2024 The Authors. *Security and Privacy* published by John Wiley & Sons Ltd.

on-premise networks. Thus, the resulting changes in SB network architecture require additional safeguards to protect publicly facing endpoints in the SB network. Ensuring such protection is more challenging as the vulnerable nature of IoT devices and their aggregation constitutes a lucrative target for adversaries aiming to leverage the combined computational power available in the network. Among various security threats, the reflective volumetric Distributed Denial of Service (DDoS) is a major attack that compromises data confidentiality and integrity as well as device availability. For example, a smart lock malfunctioning as a result of DDoS can cause the lock to be unavailable, resulting in physical access issues.

The concern of DDoS attacks in SB networks motivates the development of new techniques to identify and block attack traffic. Current studies on anomaly-based intrusion detection systems (IDSs) have shown that machine learning (ML) techniques are an effective approach to identifying malicious network traffic.[2] However, there remains a small body of work dedicated to engineering ML models with features specifically geared to distinct IoT traffic characteristics. These current studies primarily focus on smart home environments.[2–4] Similar to IoT traffic in SBs, IoT traffic in smart homes follows repetitive patterns in state change commands (on/off) sent from BAS based on configured rule sets or network pints at fixed intervals to transmit data, that is, limited flow types. However, as compared to smart home networks, SB networks have a larger scale and often consist of more heterogeneous device types,[5,6] posing more challenges in developing ML strategies to understand attacks or benign traffic.

Many existing traffic classification models rely on supervised learning techniques that employ attributes at the packet level or the flow level or their combination to characterize network traffic patterns of devices.[2,3,7] While these approaches achieve high performance in classifying benign or attack traffic, supervised classifiers only learn the differences between benign and attack traffic in the dataset. They cannot effectively identify new or unknown attacks. As for unsupervised solutions, while they hold the promise of detecting new attacks, they rely on deep neural networks to model the entropy of device traffic.[8] Neural networks are computationally intensive. For a large-scale network with a wide range of heterogeneous devices, they may not scale well if implementing packet-level feature learning (e.g., inter-packet arrival time). Therefore, our study will explore flow-level features for IDSs in SB networks. More specifically, this paper will answer the following two research questions.

- Given the heterogeneity of commercial SB networking devices, how to leverage the variance in traffic characteristics between different devices so as to aid intrusion detection?
- As feature extractions often require intensive computation tasks and subsequently delays in decision making, can we consider flow features instead of packet features to enhance efficiency while guaranteeing the security performance?

To address the above research questions, we analyze IoT traffic activities in SB networks based on which we develop insights to form a lightweight traffic classification model for each individual device. In detail, with our model, each traffic flow generated by a device in the SB network is first coarse-grained into a novel 2-flow tuple, recording the location and the direction of traffic flows at this device. This 2-flow tuple helps to extract novel yet simple traffic rate attributes from coarse-grained flows. Traffic rate features, such as total, mean, and standard deviation of bytes, incur less computational overhead than extracting features via deep packet inspection techniques thus providing a more efficient approach to the feature extraction process. In order to achieve further concise features that are yet accurate in capturing necessary IoT traffic patterns, we evaluate the exacted attributes by the feature set analysis and find that some attributes may not yield more accuracy for the model to capture traffic patterns. Hence, we are able to reduce the size of our feature set, generating the most suitable feature set (with 32 features) for the task of identifying attack traffic in SB networks. This novel feature set include features such Internet input byte counts, the standard deviation (SD) of byte counts, etc. We conclude our contributions below.

- We propose a novel fingerprint for IoT device traffic, where device flows are coarse-grained over location and direction of traffic to reveal distinct traffic patterns. We leverage these flows to accurately train the traffic classification system by extracting more concise features than those in the literature.
- We design a clustering algorithm for outlier detection to detect behavioral changes, using a small number of features to achieve per-device classification. Our experiments show that this model is able to achieve 96% accuracy in offline detection, with only 32 features to be learned by the ML algorithm.
- We evaluate the factors that impact the detection accuracy of the classification models by examining 4 feature sets with different sliding window sizes, sampling rates, and traffic rate attributes. The comprehensive evaluation of the feature sets reveals that smaller sliding windows increase the accuracy of characterizing highly variable traffic in IoT devices.

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

The rest of this article is organized as follows. Section 2 reviews state-of-the-art on intrusion detection for IoT devices in smart environments. Section 3 reviews IoT threat models in SB. Section 4 characterizes IoT network traffic behavior to gain insights into IoT traffic patterns and construct a novel fingerprint of IoT devices. Section 5 presents the lightweight clustering model to detect behavioral changes in IoT devices. The detection framework is trained on benign traffic and is validated on a mix of benign and attack traffic to evaluate the efficacy of the model. Section 6 discusses the solution and possible tradeoffs in the context of the system requirements. Section 7 concludes the study and provides limitations and directions for future work.

2 | RELATED WORK

IoT devices are being integrated as sensors and actuators in buildings and cities to create smart environments like SB that are incorporated into a wider smart city. Such a large number of vulnerable devices therefore represent an attractive target for attackers to exploit them. The literature in this area demonstrates the vulnerabilities in IoT and profiles their network traffic characteristics so that smart defenses can be developed to aid in botnet detection. Table 1 summarises IoT traffic classification systems.

Sivanathan et al.[9] explore profile-specific IoT traffic characteristics in smart environments, such as the number of unique domain names IoT devices query and how often these queries are repeated, whether cipher suites are used, and if so what type and the number of different Internet servers they communicate with. These statistics help form a distinctive signature for identifying IoT devices. An ML classification system, utilizing Naive Bayes Multinominal and random forest trees, is developed based on these statistical features to identify IoT devices based on their traffic in real-time, which is proven to be extremely accurate. Thus, the authors demonstrate that with the use of simple statistical features IoT devices and their traffic can be identified with a high degree of accuracy with the use of ML algorithms.

TABLE 1 IoT traffic classification systems.

Solution	IoT Environment	Focus	Method	Feature set size	Features per flow	Year
10	General IoT	DDoS detection	Anomaly-based CNN and RNN	N/A	Yes 5	2017
9	General IoT	IoT classification	Naive Bayes multinominal forest trees	11	Yes	2017
2	Smart home	DDoS detection	Random forests neural network decision tree linear support vector machine	6	No per packet	2018
11	General IoT	DDoS detection	Long short term memory based RNN	N/A	No per packet	2018
3	Smart home	DDoS detection	Deep autoencoders	N/A	No per packet	2018
12	Smart home	Botnet detection	Supervised classification algorithms (ZeroR, OneR)	20-30	No per packet	2018
13	General IoT	Botnet detection	Blockchain	N/A	N/A	2018
14	General IoT	DDoS detection ARP spoofing	X-means clustering Specification-based IDS	N/A	Yes 14	2019
15	General IoT	Botnet detection	Frequency-based dependency graph features	N/A	N/A	2019
4	Smart home	DDoS Spoofing MitM detection	Supervised classification algorithms (random forest, Naive Bayes, and support vector machine)	N/A	Hybrid	2019
16	General IoT	DGA detection	Siamese networks and neural networks	N/A	N/A	2020

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

There are two main techniques for IoT traffic classification relying on ML-based techniques: (1) supervised and (2) unsupervised learning. Both techniques rely on flow-based features in the network to model legitimate traffic. While these works are important and provide useful insight into IoT botnet behaviors, the main focus is on developing security solutions for the smart home. This leaves a gap in developing effective solutions aimed at addressing the requirements of SB environments. More specifically, the issue of scalability is not considered and leads to computationally expensive approaches to classify IoT traffic via the extraction of large feature sets and use neural networks.

Doshi et al.[2] develop a DDoS detection system utilizing a variety of ML algorithms. IoT-specific network behaviors are leveraged, such as regular time intervals between packets, to inform feature selection for ML algorithms that identify anomalous behavior in the network. A consumer IoT device network is constructed to collect realistic benign and malicious traffic to form an unlabeled dataset upon which the ML module is trained.

However, the focus of the detection scheme is limited to identifying DDoS attack traffic through simulating flood attacks and thus fails to capture communication patterns exhibited by botnets in the network. Moreover, the set of IoT-specific network features leveraged is small and relies on packet-level signatures that are computationally intensive to compute. While the authors provide an insightful first look at how ML algorithms can be applied to IoT networks to detect DDoS traffic, there is a lack of how similar algorithms can be applied to detect botnet behaviors to identify infected hosts in the network. Moreover, the use of consumer IoT devices means that the nature of their communication patterns will be different to those in SB as consumer devices typically have more human interaction.

Anthi et al.[4] investigate an IDS specifically for the smart home environment based on header features from various protocols in the network stack. A supervised ML approach is employed and thus the limitation of this approach is that the solution is unlikely to detect more sophisticated attacks outside of the training set. Moreover, relying on header features means that attackers can evade detection by changing packet structure. Thus, the high accuracy of the classifier in this study is a result of unsophisticated attacks. However, the study successfully demonstrates the merit of using ML techniques to classify IoT traffic and characterize attack types based on the traffic generated in a smart home environment.

Joshi et al.[17] propose a graph-based solution while taking into account partial network information. This is because constructing a full communication graph in the real world is unlikely. Therefore, relying on such a technique to detect botnet behavior can restrict where and how such algorithms can be deployed. The study successfully demonstrates that some community-based detection algorithms can be effective for botnet detection with partial network information. However, this detection is focused at the ISP layer and thus cannot leverage more nuanced data that is available at the private network layer.

Median et al.[3] successfully apply deep learning techniques using deep autoencoders to accurately identify compromised IoT devices that have been added to a botnet. However, this method relies on training a deep autoencoder for each device in the network to learn the normal behavior of devices. Moreover, it uses statistical features to characterize network behavior and thus has the limitations of not being able to fully capture the network communication patterns, which can expose additional aspects of malicious hosts. Thus, such a methodology is not ideal for SB environments that have a larger number of IoT devices. This makes it impractical to assign a deep autoencoder for each device to infer its normal behavior. Instead, a holistic approach to detecting the normal behavior of IoT networks is required, where the communication patterns are most accurately captured to identify malicious behavior.

Hamza et al.[14] investigate a specification-based IDS by utilizing the manufacturer usage description (MUD) profile to construct rule sets for 10 different IoT devices in the network and monitor their compliance in the SDN architecture. Upon policy violations in the rule specifications, ML techniques are applied to detect volumetric attacks such as DoS, reflective TCP/UDP/internet control message protocol (ICMP) flooding, and address resolution protocol (ARP) spoofing. The authors employ a similar clustering based outlier detection methodology to the one proposed in this work. However, the clustering is localized to each flow in the specification and as many as 17 traffic features are extracted to characterize each flow. Therefore, the authors demonstrate how to successfully integrate the use of MUD profiles into a specification-based IDS to achieve high rates of attack detection. However, the extraction of 14 features per flow is computationally expensive and this work aims to reduce the number of feature sets extracted to characterize IoT network activities.

3 | THREAT SURFACE IN IOT NETWORKS, SOLUTION REQUIREMENTS, AND DATASET

SB networks are exposed to a greater attack surface due to the integration of IoT devices, as cyberattacks on IoT can compromise availability via volumetric DoS attacks, integrity and confidentiality of communications via

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

ARP spoofing. Additionally, reflective volumetric attacks bypass SB network access control mechanisms, enabling the adversary to tamper with the network functions that is, nodes in the network should only carry out approved functions and events compromising this principle should be detected. We explain some attacks below.

3.1 | Reflection attacks

Vulnerabilities present in IoT devices due to poorly implemented security mechanisms lead to opportunities for adversaries aiming to exploit their resources. This is achieved by exploiting various protocols through source-spoofed traffic[18] that leverages an IoT device as a reflection vector to direct volumetric traffic attack aimed at target servers, as highlighted in Figure 1. Protocols critical to IoT operations, such as Network Time Protocol (NTP) or Domain Name System (DNS), are used to amplify small queries to generate large responses by exploiting vulnerabilities inherent in the protocols. NTP, for example, is used by devices to synchronize their time with publicly available NTP servers. Figure 1 shows how Simple Service Delivery Protocol (SSDP) and Simple Network Management Protocol (SNMP) can be exploited in an IoT network to generate reflective traffic.

A malware-infected device within the network can configure port forwarding on the IoT gateway to allow external access to IoT devices. The malware can then scouts for reflection-vulnerable UDP ports (1900 SSDP, 161 SNMP) and common TCP ports (22 SSH, 23 Telnet, 80 HTTP, 43 HTTPS) on IoT devices that are present in the building network. This information can then be transferred to the outside attacker. This scenario is illustrated in Figure 1 with steps:

1. **Initiate attack**: The attacker triggers flood messages to the command and control (C&C) server.
2. **Attack command broadcast.** The C&C server then forwards the command to the botnet of infected devices.
3. **Traffic generation.** An infected device in the network transmits IP-spoofed traffic (by forging the destination header field to contain the target server's address) to noninfected IoT devices in the local network.
4. **Traffic reflection.** IoT devices respond to amplify and reflect these packets.
5. **Victim targeted.** The IoT gateway forwards the amplified traffic to the victim server.

While IoT devices have been successfully used to generate DDoS traffic, individually the devices are computationally constrained and sustain attack traffic at different rates. This is an important factor to consider when aiming to detect infected devices in the network through their traffic characteristics, as the attack traffic generated in IoT networks is dependent on the protocol traffic and the device types within the network. For example, a smart light bulb sustains UDP based attacks at different rates (in terms of kbps) than that of a smart camera due to its resource constraints. Thus, when attackers use reflection-based attacks, the amplification factor of infected devices across the network varies and

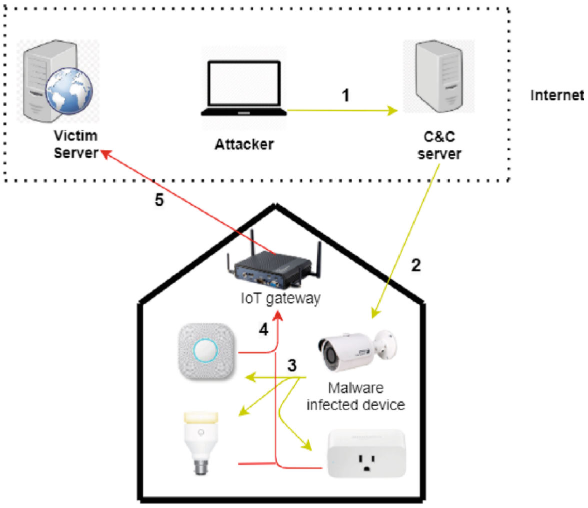

FIGURE 1 The vector of volumetric reflection attacks: initiation (1); attack command transmission (2); traffic generation (3); reflection and amplification of traffic (4) to the target server (5).

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

is dependent on their computational capabilities. The rich functionalities of smart cameras have made them a lucrative target for exploitation, making them more vulnerable to zero-day exploits.[19]

3.2 | Direct attacks

While reflection attacks aim at leveraging the resources of IoT devices, direct attacks target the confidentiality and integrity of communication channels, and device availability to disable its operations. Such attacks present security challenges in SB environments due to the large number of people who typically have access to the local network.

3.2.1 | Availability

DoS attacks are launched by flooding a device with network or transport layer traffic to overwhelm its system resources. Due to the limited capabilities of IoT devices, directing a sufficient traffic rate to cripple a device is not a significant barrier for most adversaries. Seminal works have illustrated how SNMP and SSDP traffic can be exploited to target IoT devices.[20] This raises security concerns in SB environments, where physical security mechanisms such as smart locks could be compromised by a DoS attack.

3.2.2 | Confidentiality and Integrity

Confidentiality and integrity of IoT device communications can be compromised by an adversary performing a MiTM attack, such as sending spoofed (source address) ARP packets to a device that enables the adversary to proxy the device's traffic. In SB environments, such an attack could enable an adversary to intercept video surveillance traffic generated by cameras so that the attacker can reconstruct the packets to view the video feed, or modify a "set" temperature event for a thermostat sent from a cloud server to manipulate the temperature setting value. Consequently, it can cause the Heating, Ventilation, and Air Conditioning (HVAC) systems to change the surrounding environment temperature.

3.3 | System requirements

In order to develop an effective traffic classification system to protect IoT deployments from the threat surface (outlined above) in SBs, a set of system requirements are formulated based on the SB architecture and use cases to address the challenges present in SB environments.

3.3.1 | Accurate classification

Attacks, such as ARP spoofing, target the integrity and authenticity of device communications with Internet servers that BAS rely on to manage building operations. Typically, BAS has predefined rules to manage device states in the building, such as turning off the lighting systems at a certain time of day. MiTM attacks enable an adversary to alter the device state without authorisation. Similarly, legitimate device functions such as software updates present a risk to IoT devices as the protocols, for example, file transfer protocol (FTP), used to transmit genuine updates can be exploited by adversaries to send malware files to the device. Therefore, accurately classifying the network traffic is critical to ensuring the integrity and authenticity of SB operations.

3.3.2 | Scalability and operational viability

The large number of IoT devices in SBs requires a detection system to be capable of scaling efficiently and accurately to monitor the network activities on a large-scale. However, there is an inherent tradeoff between efficiency and accuracy

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

where the algorithmic complexity of a classification model is directly associated with its accuracy and its subsequent overhead (efficiency). Therefore, the ML pipeline, from feature extraction to model training, and classification tasks, must be efficient in order to effectively monitor the hundreds to thousands of nodes present in an SB network.

Furthermore, the traffic classification system must address the variability of traffic patterns present among the nodes as well as the addition and deletion of nodes to the network. As such, the ML pipeline should be able to effectively handle changes to the network structure or software updates to the device which may result in changes to its traffic patterns. Specifically, the re-training of the classification model upon changes to the network and devices must be an efficient process. Additionally, the classification model must achieve a high level of accuracy and ensure the false positive rate is acceptable to be operationally viable. Therefore, a scalable and accurate system must balance this tradeoff to find the optimal solutions for a given environment.

3.4 | Dataset

To analyze and yield new insights on IoT traffic patterns that can be leveraged to build and evaluate a solution adhering to the listed requirements, we use the University of New South Wales IoT dataset.[14,21] The dataset is a set of traffic captures of an IoT network testbed containing over 28 IoT devices ranging from energy management, controllers/hubs, cameras, appliances and health monitoring.[14,21] The dataset contains a set of benign traffic PCAP files for the training of classification models and a set of files containing a mix of volumetric attack and benign traffic to validate the models. The set of attacks range of reflective and direct DDoS traffic and are outlined in Tables 5 and 6, respectively.

4 | PROFILING IOT TRAFFIC

4.1 | Coarse-grained flow-level characteristics

The primary use of IoT in SBs is to remotely manage building operations via the BAS issuing commands to alter device state (boot, active, idle). Command traffic can either be sent in the local network or the Internet, depending on where the BAS is located and what systems are configured to communicate with the devices. SB environments typically place the BAS in the cloud to employ artificial intelligence (AI) techniques to analyze building operations' data for actionable insights that result in altering device states. The flow-level characteristics of device communication pathways are summarized into a flow tuple filter in Equation (1) to capture the traffic patterns at a source node/device. This is done by segregating device traffic on flow directions (incoming and outgoing) and location (Local/Internet) of traffic to create a 2-tuple flow:

$$FlowTupleFilter = [location, direction]. \tag{1}$$

This is to generate features that are protocol agnostic and capture traffic patterns that reflect the functionality of devices. The location filter enables the characterization of protocols typically present in the local/private network but may not be configured to be accessible from the Internet such as SNMP and SSDP. Furthermore, it captures the key IoT-specific trait of communicating with a limited number of Internet endpoints by fine-graining traffic at the source device. The direction filter captures the input/output (bytes up/ bytes down) patterns in a node. As the network protocol stack varies across devices, protocol based features are not ideal due to their inapplicability. For example, nodes that use TCP more than UDP will have different input traffic rates compared to UDP where acknowledgements are not expected. Moreover, a bidirectional flow that contains a command input to alter device state reflects the corresponding output traffic. For example, a command instructing a smart camera to take a photo and upload the image to an Internet host exhibits different outgoing flow characteristics (i.e., smaller flow duration and size), compared to a command instructing the camera to stream video traffic via the cloud, leading to larger outgoing flow size and duration. Intuitively, this assumption makes sense: if a device receives a change state command, the information needs to be acknowledged and the resulting network behavior will reflect the triggered event. As a result, devices with richer functions contain diverse flow types while less capable devices have limited flow types. The filter creates 4 flow types at the source device: local inputs, local outputs, Internet inputs, and Internet outputs. For instance, local inputs consist of all traffic *received* by the device sent from the local network. While local outputs consist of all traffic *sent* by the device to the local network. The same principle applies

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

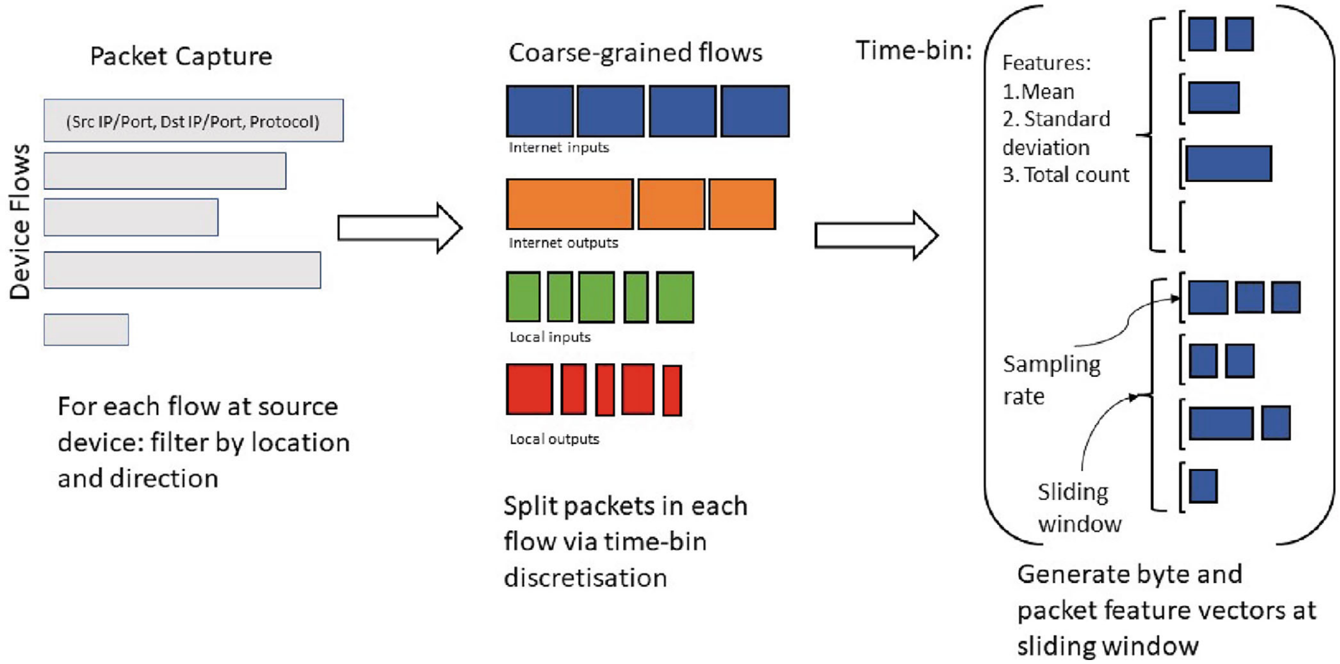
Packet Capture
Device Flows
(Src IP/Port, Dst IP/Port, Protocol)
For each flow at source device: filter by location and direction
Coarse-grained flows
Internet inputs
Internet outputs
Local inputs
Local outputs
Split packets in each flow via time-bin discretisation
Time-bin:
Features:
1.Mean
2. Standard deviation
3. Total count
Sampling rate
Sliding window
Generate byte and packet feature vectors at sliding window

FIGURE 2 Coarse-graining flows and feature extraction pipeline at the source device. A sampling rate of 30 s (smaller brackets) and a sliding window of 120 s (large bracket) are illustrated for a coarse-grained flow (e.g., Internet inputs) time-bin.

to Internet traffic. This process is illustrated in the first 2 steps in Figure 2. Thus, the direction of traffic defines whether the device is the *source* of the traffic or whether it is the *destination*. The location defines the origin network of the end point the device is communicating with, given the BAS can be located on-premise (local) or the Internet.

4.2 | Fingerprinting device flows

Device traffic can be fingerprinted by converting the traffic in the 4 flows produced by the flow tuple filter to time series vectors, where vector elements contain the amount of traffic (bytes) sent or received by the device into s-second samples. The vectors are then further sorted into a larger sliding window to extract 2 feature vectors, mean and SD, of the traffic in the larger w-second window (time-bin discretization). This process is illustrated in Figure 2. This creates clear clusters of traffic patterns present in the flow types, illustrated in Figure 3. This is particularly important for devices with richer features and varied flow types, such as cameras.

Similarly, TP-Link Smart plug and Belkin Wemo switch form distinct clusters despite being single-purpose devices with limited capabilities (on/off). This observation is consistent with findings in Section 4.1, where device types from different vendors have different degrees of byte rates. This methodology is leveraged to fit the insights gained in Section 4.1 to extract feature vectors, mean and SD, for local inputs, local outputs, Internet inputs, and Internet outputs to fingerprint device traffic. The fingerprints show specific patterns between device types and between devices of the same type (produced by different vendors) when a sampling rate of 10 s and a larger time window of 240 s (4 min) is applied. The variety in traffic profiles among the devices is due to the network protocol implementations in the devices, which differs according to the device vendor. The diversity in traffic characteristics among the devices is primarily down to their use cases, functionality, and operating states of the devices. For example, the network characteristics of an idle operational state in a Belkin switch are different to that of iHome. Different network protocol implementations among the devices, both in terms of local network and Internet, impact the traffic profile of the device and its network behavior. This, in turn, determines the possible reflection attacks the device can launch as reflection attacks rely on exploiting protocol vulnerabilities, detailed in Section 3. For example, the Samsung SmartCam uses UDP for video transmission compared to Netatmo Cam that uses TCP. Netatmo Cam receives TCP acknowledgements unlike the SmartCam, leading to higher rates of input traffic in Netatmo. The diverse set of fingerprints and their traffic characteristics serve to demonstrate the challenges faced in classifying IoT traffic for anomaly-based IDS.

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

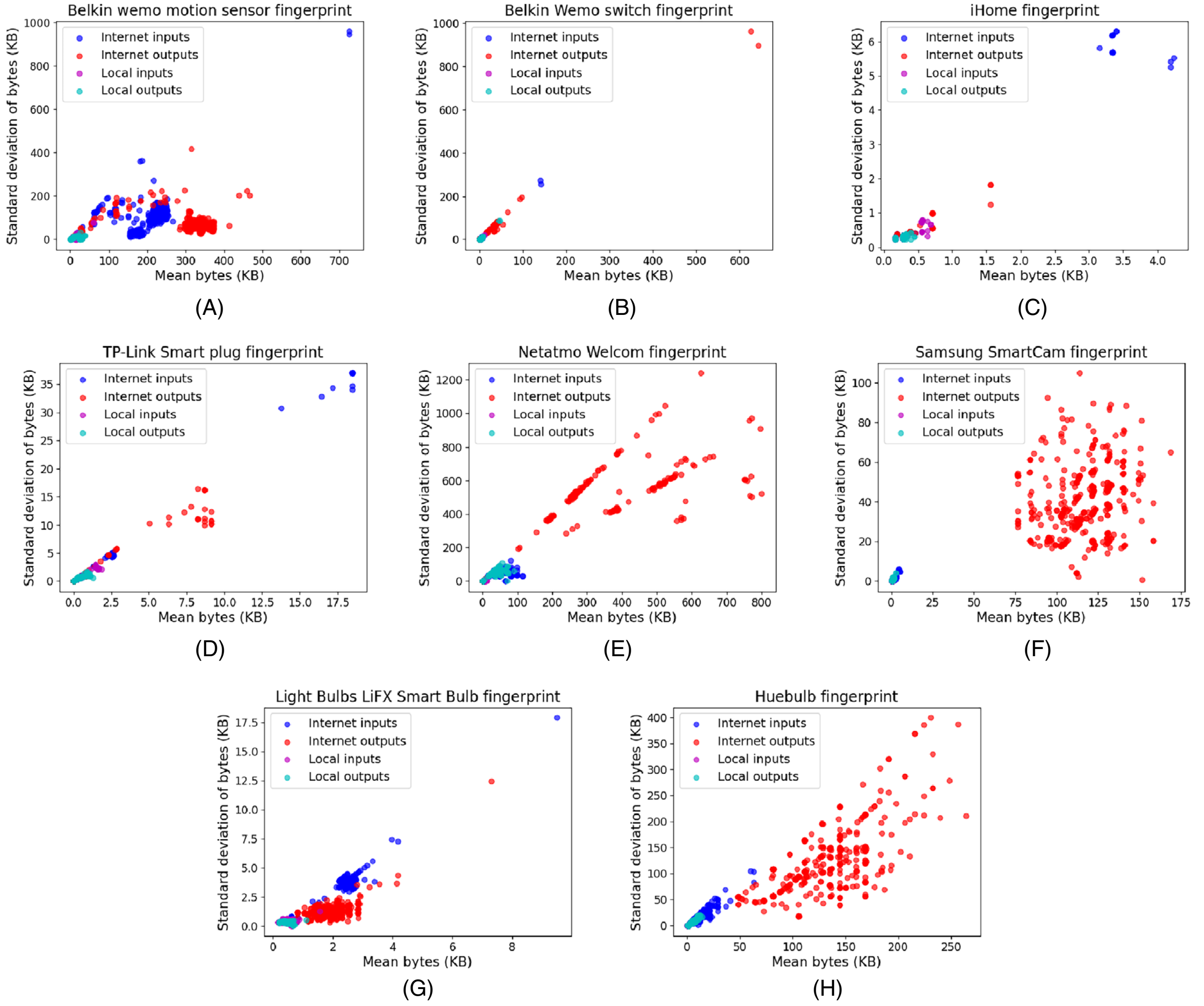

FIGURE 3 Device fingerprints: coarse-grained flow traffic features. (A) Belkin motion; (B) Belkin switch; (C) iHome; (D) TP link plug; (E) Netatmo Cam; (F) Samsung SmartCam; (G) LiFX; (H) Huebulb.

To address this issue, an ML-based traffic classification system is required to achieve high accuracy and operational viability. However, ML-based techniques incur high computational costs and the need for scalable solutions (efficient and effective) for the large-scale IoT devices present in SBs. The insights gained regarding the specific patterns exhibited on location and direction of traffic are used to build device specific classification models in Section 5.

5 | PROTECTING IOT IN SBS

This section builds on the flow tuples identified in the previous Section to extract features from key attributes for a traffic classification system. It aims to answer the research question of using low-cost attributes for the characterization of traffic and evaluates multiple feature sets to find the optimal set that balances the tradeoff between accuracy and efficiency. The challenge of modeling the traffic profiles of a heterogeneous set of devices present in the SB environment is tackled by developing device-specific K-means clustering models, to run as an SDN application. The results of feature set analysis and the low-cost nature of the attributes are then discussed to evaluate the operational viability of the models in SB networks.

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

5.1 | Unsupervised IoT traffic classification

The primary aim is to train a set of classification models, a dedicated model per device, where each model learns the traffic patterns of its device network traffic characteristics. Device specialized models have been shown to be an effective approach to better detect anomalous traffic.[3,14,20] We employ a similar approach to model legitimate device behavior in the network. This optimizes the decision boundaries/thresholds in classifying the traffic by reducing the variance in traffic only to that of the device rather than all the devices in the network. Furthermore, this enables the model to learn the legitimate network behavior of the device and detect deviations from expected boundaries, thus leading to identifying zero-day attacks. Although device-specific models enable better characterization of device traffic they incur a high computational cost in large-scale SB networks, where having device specific models in an environment consisting of hundreds of devices can be resource-intensive. Moreover, models that rely on features extracted for all flows increase feature extraction overheads and compromise the overall scalability of the solution. Consequently, device-specific models relying on large feature sets to characterize device traffic are an inefficient approach to monitoring a large-scale SB network.

We aim to leverage the distinct behavior of the 4 flow types identified in Section 4 to extract features to characterize their traffic, thereby avoid generating large feature sets, and take advantage of the merits offered by device-specific model architecture to provide a more scalable solution appropriate for large SB networks. Each model is trained on features extracted from the 4 flows (including Internet inputs, Internet outputs, local inputs, and local outputs) of benign traffic from their respective IoT device to detect whether network traffic observations belongs to the trained class or not.

5.1.1 | Data preprocessing

In order to synthesize device traffic into a specific flow type, based on the location and direction of traffic, to extract the necessary attributes from the traffic traces in the dataset, a native packet parsing tool, in Python, is developed.[22,23] The process is illustrated in Figure 2, where raw pcap files are taken as inputs to construct a flow table for each device, if device traffic is present in the file. Note that flow tables are present in SDN switches and device flow level telemetry can be extracted in a similar process. The flows are then sorted according to the location and direction of traffic by inspecting the source and destination MAC addresses to first determine whether the traffic is incoming to the device (destination address is device MAC address) or outgoing (source address is device MAC address). The flows are then further segregated depending on whether the source or destination is in the local network to create the 4 filtered flows: local inputs, local outputs, Internet inputs, and Internet outputs. Each flow represents aggregated traffic in the filter. Lastly, byte and packet counters in each flow type are computed at a configurable resolution, that is, sampling rate (e.g., 30 s). The counters are used to generate a stream of time series vectors based on a sliding window time scale of a configurable resolution (e.g., 2-min). Employing such techniques allows for attribute extraction over the 4 flow types for both training and testing instances.

The various attributes (e.g., packet size) extracted per flow type result in feature sets with high dimensional data that can be computationally expensive for the K-means algorithm to compute efficiently. In order to reduce the dimensionality of the feature set, principal component analysis (PCA) is employed to transform the originally larger feature set to a smaller subset that captures 99.99% of the variation in the benign instances. Therefore, due to PCA requirements of normalized features, for a given training instance the attributes are normalized using z-score, where the z-score scaler (mean and std) is recorded for each attribute. The recorded scalers are then applied to test instances (a mix of benign and attack traffic). By normalizing each feature independently, the risk of overweighing large value features (e.g., Internet outputs) over smaller features (local inputs) is reduced. Through experimentation and the elbow method,[24] the optimal cluster points for the K-means algorithm are deduced from the resulting principal components for each feature set outlined in Table 2.

5.1.2 | Identifying anomalies through boundary outlier detection

The K-means clustering algorithm divides its training instances into spherical clusters, when tuned optimally (the number of clusters) the instances neatly converge into a spherical shape around the cluster center point (centroid). The right

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE 2 One-class classification model metrics per device: time-series traffic data (training and testing) in the dataset, reduced dimensionality of features sets (principle components), and optimal number of clusters and cluster boundary for outlier detection in K-means clustering algorithm.

	Training	Test instances (min)		No. of principal components	Total	Cluster
Device	instances (min)	Total	Attack	FS1 FS2 FS3 FS4	clusters	boundary
Belkin motion	38 284	20 454	300	12 14 13 16	64	98%
Belkin switch	20 954	12 750	180	12 15 12 16	64	97.5%
iHome	37 906	19 342	30	11 15 12 16	64	98%
Netatmo	13 529	10 639	147	9 13 9 13	128	99%
Samsung SmartCam	38 227	36 747	357	11 14 12 15	128	99.5%
TPlink Smart plug	38 211	35 205	178	11 12 12 13	64	99%
Phillips Huebulb	11 530	9376	297	9 14 11 16	64	98%
LiFX bulb	25 903	26 181	150	11 13 14 11	64	98.5%

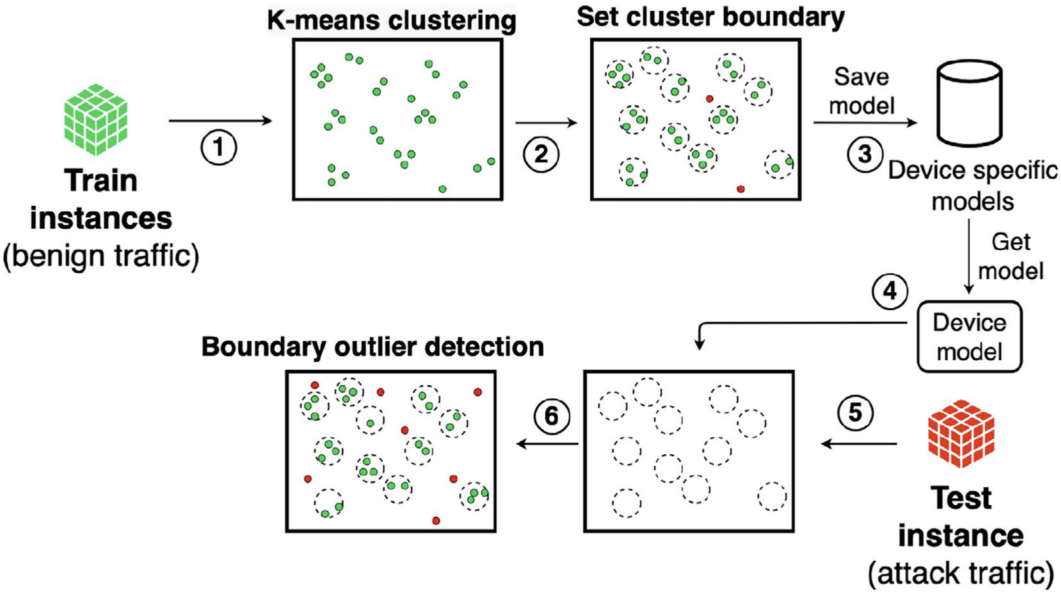

FIGURE 4 SB device model training and detection operations.

number of clusters is critical to effective classification of device traffic. A smaller number of clusters can lead to under fitting the data, as the classifier is not discriminatory enough to classify anomalies accurately. On the other hand, a large number of clusters than the optimal point over fits the data and leads to an inflexible classifier. Figure 4 outlines the training and detection process to identify anomalous traffic instances. The outlier detection process is as follows:

1. Benign device traffic features are fed to the K-means clustering algorithm to find the centroid points for the number of clusters specified.
2. The Euclidean distance between the centroid and instances belonging to the cluster point is calculated to determine the cluster boundary. The cluster boundary is defined as a percentile covering the nearest instances for each cluster in the model from the training dataset (detailed in Table 2). For example, a cluster boundary of 98% considers the closest 98% of instances of the cluster as the boundary. Instances lying beyond the boundary are classified as anomalies. Inherently, this leads to a baseline false positive rate (FPR) as benign traffic will be classified as anomalous.
3. Upon training of each device model, the centroid points and cluster boundaries are saved to have a generated device-specific model.
4. The trained device model is obtained (i.e., centroid points and respective boundaries) based on the device being tested.

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

5. The trained model is tested on a mix of benign and attack traffic, where the raw pcap files are preprocessed, features extracted and normalized, and feature dimensions reduced. The transformed test instances are fed to its device model to classify the nearest cluster centroid, from training data, for each test instance.
6. The Euclidean distance between a test instance and its nearest centroid is computed and checked to determine whether it lies inside or outside the corresponding cluster boundary.

5.1.3 | Feature engineering

Accurately characterizing long range dependent traffic requires traffic features to be computed at multiple time scales (i.e., sliding windows). Building on the fingerprint created in Section 4.2, the efficacy of features extracted from the 4 flows, via the sampling rate and sliding window technique as illustrated in Figure 2, for device traffic classification is evaluated. These attributes are selected as they are highly correlated and extracting flow-level traffic rate features are of low cost compared to packet-level features (e.g., inter packet interval time). Figure 5 illustrates the difference in feature

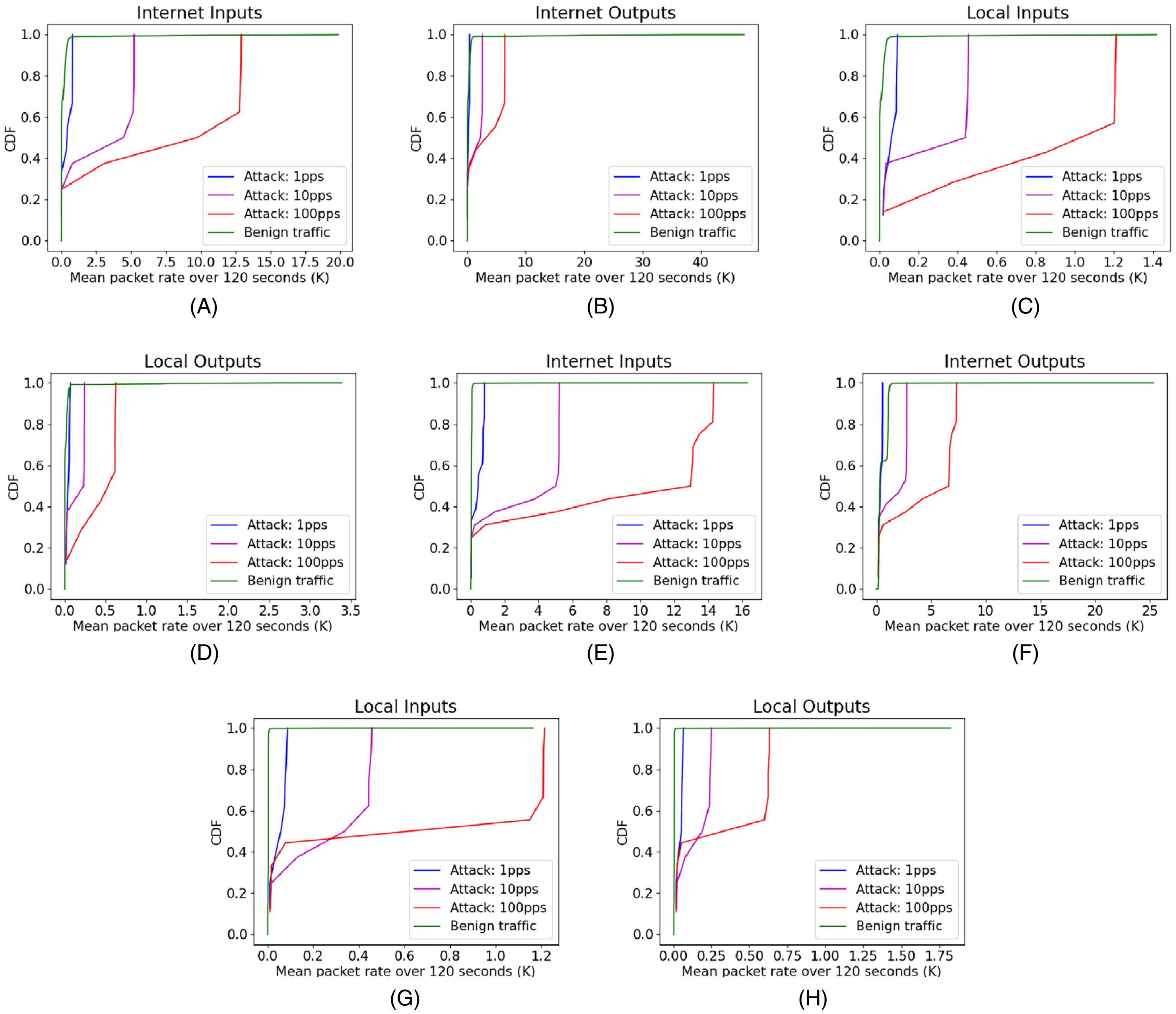

FIGURE 5 Benign and attack traffic feature characteristics of 2 sample IoT devices. A sampling rate of 30 s and a sliding window of 2-min. (A) Netatmo Welcome: Internet Inputs; (B) Netatmo Welcome: Internet Outputs; (C) Netatmo Welcome: Local Inputs; (D) Netatmo Welcome: Local Outputs; (E) Samsung SmartCam: Internet Inputs; (F) Samsung SmartCam: Internet Outputs; (G) Samsung SmartCam: Local Inputs; (H) Samsung SmartCam: Local Outputs.

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

characteristics between attack and benign traffic. The one-class classification models are able to learn and differentiate between benign and attack traffic instances. The SD of total byte and packet counts between the sampling rate windows measures the variability and burstiness of traffic rates over the coarse-grained flows, illustrated in Figure 2. The mean bytes and packets values measure the average traffic rate in the sliding window, and the total count measures the total volume of bytes/packets in the sliding window. These features combined model the traffic patterns for each device-specific classification model to learn benign traffic patterns, while showing attack traffic to exhibit different characteristics that can be detected. This is highlighted in Figure 5, where the mean packet rate in coarse grained flows is shown to exhibit distinct benign and attack traffic characteristics in two sample camera devices. The key flow attributes identified are extracted at multiple sliding windows (time granularity) to construct and evaluate various feature sets detailed in Table 3.

The need for features to be extracted at multiple sliding windows adds complexity and overhead to the feature extraction process. Additionally, the size of the largest sliding window determines the latency of the extraction process. For example, a 2-min sliding window for extracting 38 282 min of training instances in the Belkin motion sensor is slower than a 4-min sliding window. Thus, the feature sets serve to evaluate the impact of packet-based features and the use of larger sliding windows on the classification models' detection accuracy. As such, mean and SD values of packet count are not extracted in FS1 and FS2, while FS2 and FS4 use all three sliding windows. Thus, the performance difference between the feature sets will give insight into the factors that impact the scalability and accuracy of the traffic classification systems.

5.1.4 | Feature set performance analysis

A range of sampling rates are tested to explore the impact of this parameter on the extracted features (e.g., local inputs mean byte count) on all feature sets (FS1–FS4), as findings by Apthorpe et al.[25] show that sampling rates impact the accuracy of classifying IoT traffic. The performance of the feature sets, extracted under various sampling rates, is comprehensively reviewed, based on detection rate of attacks, true positive rate (TPR) and false positive rate (FPR), to find the optimal set that balances the tradeoff between accuracy and efficiency. Table 4 shows the overall performance of the feature sets in detecting various attack scenarios. Direct attack types include ARP spoofing, TCP SYN flooding, and Fraggle (UDP flooding). Reflection attacks include SNMP, SSDP, and TCP SYN. Each attack type is launched at three different rates: 1 packet per seconds (pps), 10 pps, and 100 pps. We make some key observations, detailed in Table 4, based on our experiments. First, the use of packet rate features (mean and SD) in FS3 and FS4 results in a higher average detection

TABLE 3 Combination of features present in the feature sets.

	Features									
	Count		Mean		SD		Sliding windows			
Feature set	Byte	Packet	Byte	Packet	Byte	Packet	1-min	2-min	4-min	Total attributes
FS1	✓	✓	✓		✓		✓	✓		24
FS2	✓	✓	✓		✓		✓	✓	✓	40
FS3	✓	✓	✓	✓	✓	✓	✓	✓		32
FS4	✓	✓	✓	✓	✓	✓	✓	✓	✓	56

TABLE 4 Detection rate (%) performance of feature sets.

Attack Ttpe	Reflective			Direct		
Attack	TCP SYN	SSDP	SNMP	ARP	TCP SYN	Fraggle
FS1	64.81	64.35	58.02	68.14	63.14	57.40
FS2	62.5	60.41	60.49	61.48	58.10	57.40
FS3	75.92	71.42	70.98	78.37	75.13	84.44
FS4	75.46	68.44	75.81	73.33	76.65	81.11

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

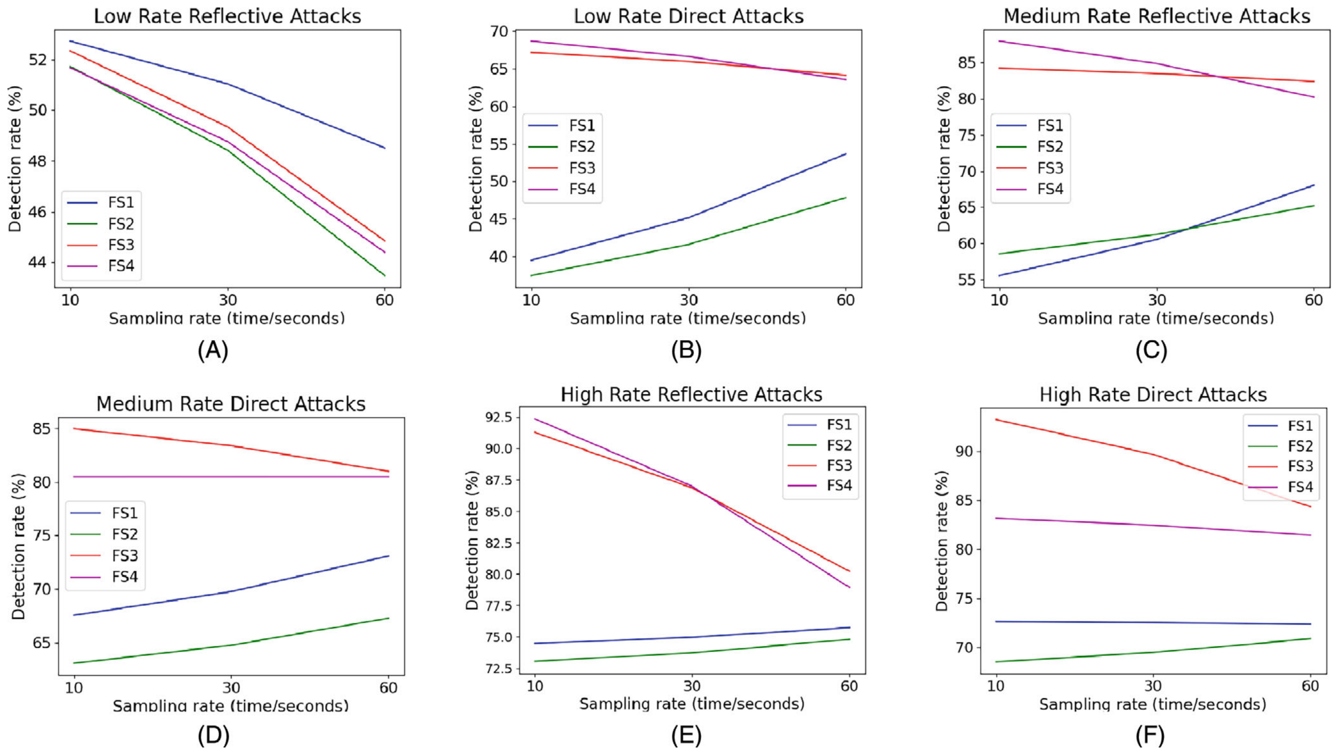

FIGURE 6 Evaluating the performance of feature sets on the detection of different attack rates and the impact of sampling rate on detection rate (instances detected / total instances) in the classification models trained on the feature sets.

rate for all device-specific classification models. This shows that packet-based features enable the classification model to capture a device's traffic patterns and more accurately distinguish between attack and benign traffic instances. This is illustrated in Figure 5 to demonstrate how reflective and direct attack traffic rates exhibit distinct packet count patterns on each coarse-grained flow compared to benign traffic.

Second, similar to findings in related works,[10] we find that larger sliding windows increases (FS2, FS4) the sensitivity of the model to detect low-rate attack scenarios, both reflective and direct, as illustrated in Figure 6A. However, larger sliding windows incur high FPR, where benign instances are incorrectly classified as anomalies, and the gains illustrated in Figure 6A,B is shown to be minimal. Moreover, larger sliding windows fail to yield any benefits in the detection of medium (10 pps) and high rate (100 pps) attacks. Figure 6 illustrates the impact of sampling rates on detection rates. The detection rates of attack types for feature sets, averaged across all sampling rates, are further detailed in Table 4 to reveal that FS3 provides the most optimal solution.

Attributes that correlate the variance in the respective rates make intuitive sense to increase detection accuracy. However, the use of a small sampling rate increases the cost of feature extraction. The comprehensive feature set analysis reveals that FS3, consisting of 32 features, is an effective approach to characterize IoT network traffic patterns. Due of the optimal performance provided by FS3 combined with a sampling rate of 10 s to extract feature attributes, we employ this feature set to evaluate the performance of device-specific models in Section 5.2.

5.2 | Performance analysis

Building on the optimal feature set, and sampling rate combination, identified in the previous section, the performance of device-specific classification models is evaluated under different attack scenarios. The scenario of each attack is varied in terms of the location of the attacker being either in the local network or Internet. Tables 5 and 6 detail the detection rate of various attack types (reflective and direct) in the device-specific models. Reflective attack scenarios are detailed in the table by the location of the attacker then the location of the victim host the IoT device is reflecting traffic to, that is, L → D → I, while direct attack scenarios detail the location of the attacker.

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE 5 The detection rate (%) of reflection attacks (columns) for each device classification model (rows).

Attack type	SNMP			TCP SYN		SSDP		
Scenario	I → D → I	L → D → I	L → D → L	L → D → L	I → D- → I	L → D → L	I → D → I	L → D → I
Packet rate (s)	1 10 100	1 10 100	1 10 100	1 10 100	1 10 100	1 10 100	1 10 100	1 10 100
Belkin motion				40 100 100	33 70 80	80 80 100	67 80 100	60 100 100
Belkin switch				67 100 100	55 80 70			
Phillips Huebulb				40 80 100	40 80 100	0 40 75	70 84 90	50 70 100
LiFX bulb								
Netatmo cam				67 100 80	50 80 90			
Samsung Cam	83 100 100	50 80 100	0 100 100	80 100 100	76 100 90			
TP-Link switch				100 100 100	75 80 100			

Note: L, local; D, device; I, Internet.

TABLE 6 The detection rate (%) of direct attack types (columns) for each device classification model(rows).

Attack type	ARP spoofing	TCP SYN		Fraggle	
Scenario	L → D	L → D	I → D	L → D	I → D
Packet rate (s)	**1 10 100**	**1 10 100**	**1 10 100**	**1 10 100**	**1 10 100**
Belkin motion	17 20 100	17 100 100	25 80 70	0 100 100	
Belkin switch	0 100 100	100 100 100	45 80 70		
Phillips Huebulb	20 40 70	50 70 80	60 80 100		
LiFX bulb	80 80 60			60 80 80	40 80 100
Netatmo cam	67 100 100	83 100 100	36 64 70		
Samsung Cam	84 80 67	100 100 100	75 82 100	80 80 100	100 100 100
TP-Link switch	100 100 100	83 83 100	58 80 80		
iHome	0 50 100				

Note: L, local; D, device; I, Internet.

Beginning from the detection rate of reflection attacks in Table 5, we find that the average detection rates for SNMP, SSDP, and TCP SYN reflection attacks in all scenarios are 79.2%, 75.4%, and 81.83%, respectively. The average detection rate is mainly reduced by the model's poor performance in detecting low-rate attacks of 1 pps. This is primarily down to the 2-min window size employed in FS3, which is less sensitive to low-rate attacks. Though the detection rate for low-rate attacks in the model is optimized by using smaller sampling rates, the overall performance remains poor. Netatmo cam exhibits poor low-rate attack detection rates perhaps due to the high variability of its traffic profile. The detection rate of direct attacks in Table 6 details the average detection rate for ARP spoofing, TCP SYN, and Fraggle attacks of all scenarios are 71.4%, 76.33%, and 84.4%, respectively. Due to the window size of the feature set, the model has relatively poor performance for detecting low-rate direct attacks. However, the model's overall accuracy demonstrates that IoT traffic behavior can be effectively captured by extracting key traffic attributes on the coarse-grained flow as identified in Section 4. Notably, Figure 5 clearly shows the distinct patterns exhibited between attack and benign traffic features among varying attack rates. The clear difference in probability distribution across benign and attack traffic patterns successfully demonstrates how the devised features from the coarse-grained flows enable the K-means clustering model to yield a high level of accuracy. Furthermore, the traffic attributes identified and selected in feature set analysis, such as byte and packet counts, are flow-based and can be deemed as relatively low cost with the use of networking hardware, such as OpenFlow capable devices.[26] OpenFlow APIs enable dynamic measuring of specific flow-level features such as byte count, packet count, and flow duration relatively easily.

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE 7 Performance and efficiency of device-specific models.

Device	TPR (%)	FPR (%)	Accuracy (%)	Model size (KB)
Belkin Wemo motion	95.3	3.82	95.37	95
Belkin Wemo switch	97.67	1.50	97.70	50
Phillips Huebulb	93.46	4.57	93.23	27
LiFX bulb	92.94	6.83	92.99	60
Netatmo Cam	97.21	2.28	97.24	38
Samsung SmartCam	96.48	2.93	96.53	80
TP-Link switch	98.49	1.02	98.50	84
iHome	96.45	3.43	96.45	80

6 | DISCUSSION

We have shown that devices exhibit specific traffic patterns in flows coarse-grained over location and direction of traffic at the source device. We show that simple traffic rate features extracted over these flows can be leveraged to build relatively small feature sets. These feature sets show promising performance for unsupervised traffic classification tasks as they effectively characterize device traffic patterns. Critically, devices are able to be fingerprinted over the mean and SD of bytes on these flows because the variation in device traffic rates, which is a reflection of device activity patterns, is captured through the time series vectors that sample the mean and SD over a sliding window. The performance analysis of feature sets reveals that FS3, consisting of 32 features provides the most optimal performance in balancing the tradeoff between efficiency and accuracy at the cost of using a relatively small sliding window of 2-mins, which can result in large overheads during the feature extraction process. The performance of FS1 and FS2 demonstrates the need for packet-level features to be extracted in the feature extraction process to accurately characterize device traffic patterns in the coarse-grained flows. Note that larger sliding windows, FS2 and FS4, suffer from higher FPR respective to FS1 and FS3.

The size of the sampling rates clearly affects the sensitivity of the models to detect low-rate attacks across all feature sets, where smaller sampling rates increase the detection rate. Similarly, smaller sampling rates increase detection rate for the detection of medium and high rate attacks in FS3 and FS4. On the other hand increasing the size of the sampling rate yields gains in the detection of medium rate attacks and minimal gains for detecting high rate attacks in FS1 and FS2. This is due to the lack of packet-based features in FS1 and FS2. However, reducing the sampling rate size adds overhead to the feature extraction process. Table 7 presents the most accurate classification models from the set of experiments: FS3 with features extracted at a sampling rate of 10 s and sliding window of 2-min.

Overall, in response to the accuracy efficiency trade-off, we demonstrate that a device-specific classification model trained on low-cost flow-based features enables a relatively lightweight approach for integrating ML-based traffic classification systems for SB environments. The high TPR (benign instances are correctly classified) of the ML models can be leveraged to ensure the authenticity and integrity of autonomous IoT communications for large-scale deployments in commercial SB environments. However, the lightweight nature of the solution is not without its tradeoff. First, the model has limited sensitivity to detect low-rate attacks, specifically in scenarios where the location of the attacker is from the Internet. This is likely due to the nature of the coarse-grained flows, where low-rate attacks in Internet input flows resemble benign traffic characteristics, such as device state commands. This is possibly due to the majority of traffic consisting of Internet connections in the devices. Finally, we demonstrate the usefulness of traffic classification systems as a tool for addressing security concerns in IoT networks, primarily their ability to provide visibility into the attack surface of IoT nodes and identify traffic that poses a threat to the availability of the node and the confidentiality and integrity of its communications.

7 | LIMITATIONS, FUTURE WORK AND CONCLUSION

Although our device-specific models focus on the detection of attack traffic patterns at each device, software updates to the devices pushed independently by the manufacturers may alter the benign traffic patterns and thus require re-training

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

of the models. Subsequently, the current implementation will lack the sensitivity to effectively adapt to new variances in benign traffic patterns of the device that could adversely impact the accuracy of the model to correctly identify malicious traffic. Moreover, implementing the classification models in a real-world scenario would require its findings to be integrated with a Security Operations Center (SOC) for security analysts to examine the outputs of the models and identify false positives manually. Such a process is typical for security tools in the real world as security incidents require human investigation before being flagged as cyberattacks. While the cost of feature extraction is discussed, the empirical cost of extraction of each feature set is not measured. Measuring the cost of feature extraction on a real SDN switch would enable an empirical evaluation of the viability of the key identified attributes (e.g., mean bytes) to scale to meet the requirements of SB networks. Similarly, the device-specific clustering models are only tested in an offline capacity and its performance in real time detection is unknown. While the fingerprint established in this work is based on coarse-grained flows that show consistent behavior of local and Internet traffic in the devices, this is largely a result of the testbed. As such, the traffic patterns will vary in different environments. For example, in scenarios where the BAS is located in the local network, it is likely the dominance of Internet input and output traffic present in the dataset will vary from the observations in Section 4. Another interesting direction could be to investigate packet-level features to capture the event-driven, back and forth nature of IoT communications. Specifically, it has been shown device event flows exhibit specific patterns in packet length and direction to form pairs of packets.[27] Using packet pairs, a.k.a.packet-level signatures, could be a way forward to model bidirectional traffic in IoT devices for the monitoring of device state triggers to ensure illegitimate event triggers are detected more accurately, that is, ARP spoofing attacks.

Deploying traffic classification systems for SB networks presents challenges when scaling to a large number of devices while achieving a high level of accuracy. We have demonstrated that device-specific models effectively manage the tradeoff between scalability and accuracy. Given our experimental results, simple traffic rate features extracted on the coarse-grained flows in the device are shown to be a low-cost technique to characterize device traffic patterns and enables classification models to achieve a relatively high degree of accuracy in classifying between benign and attack traffic instances. Finally, SB administrators may not fully realize the risks posed by and to the IoT devices in their network and further lack tools to detect malicious behavior of devices in a timely manner. The device-specific classification models provide visibility into the network operations of IoT nodes and the ability to identify breaches that can be acted upon to mitigate. As such, SB administrators seeking to protect their IoT assets can employ this solution in their security architecture to enhance their organization's overall security posture by having greater visibility into the SB network's attack surface.

ORCID

Muhammad Rizwan Asghar https://orcid.org/0000-0002-9607-376X

REFERENCES

1. Plageras AP, Psannis KE, Stergiou C, Wang H, Gupta BB. Efficient IoT-based sensor BIG Data collection–processing and analysis in smart buildings. *Futur Gener Comput Syst*. 2018;82:349-357.
2. Doshi R, Apthorpe N, Feamster N. Machine learning DDoS detection for consumer Internet of Things devices. *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE; 2018:29-35.
3. Meidan Y, Bohadana M, Mathov Y, et al. N-baIoT–network-based detection of IoT botnet attacks using deep autoencoders. *IEEE Pervasive Comput*. 2018;17(3):12-22.
4. Anthi E, Williams L, Słowińska M, Theodorakopoulos G, Burnap P. A Supervised Intrusion Detection System for Smart Home IoT Devices. *IEEE Internet Things J*. 2019;6(5):9042-9053.
5. Wong W. What's the Difference Between Consumer and Industrial IoT? 2016 https://www.electronicdesign.com/technologies/iot/article/21801856/whats-the-difference-between-consumer-and-industrial-iot
6. Gartner Inc. Gartner Says 5.8 Billion Enterprise and Automotive IoT Endpoints Will Be in Use in 2020. 2019 https://www.gartner.com/en/newsroom/press-releases/2019-08-29-gartner-says-5-8-billion-enterprise-and-automotive-io::text=Gartner%2C%20Inc.,a%2021%25%20increase%20from%202019.text=Utilities%20will%20be%20the%20highest,to%20reach%201.37%20billion%20endpoints
7. Tahaei H, Afifi F, Asemi A, Zaki F, Anuar NB. The rise of traffic classification in IoT networks: A survey. *J Netw Comput Appl*. 2020;154:102538.
8. Mirsky Y, Doitshman T, Elovici Y, Shabtai A. Kitsune: An ensemble of autoencoders for online network intrusion detection. arXiv preprint, arXiv:180209089 2018.
9. Sivanathan A, Sherratt D, Gharakheili HH, et al. Characterizing and classifying IoT traffic in smart cities and campuses. *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE; 2017:559-564.
10. Lopez-Martin M, Carro B, Sanchez-Esguevillas A, Lloret J. Network traffic classifier with convolutional and recurrent neural networks for Internet of Things. *IEEE Access*. 2017;5:18042-18050.

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

11. McDermott CD, Majdani F, Petrovski AV. Botnet detection in the Internet of Things using deep learning approaches. *2018 international joint conference on neural networks (IJCNN)*. IEEE; 2018:1-8.
12. Chawathe SS. Monitoring IoT networks for botnet activity. *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*. IEEE; 2018:1-8.
13. Sagirlar G, Carminati B, Ferrari E. AutoBotCatcher: Blockchain-based P2P botnet detection for the Internet of Things. *In: 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*. IEEE; 2018:1-8.
14. Hamza A, Gharakheili HH, Benson TA, Sivaraman V. Detecting volumetric attacks on IoT devices via SDN-based monitoring of MUD activity. *Proceedings of the 2019 ACM Symposium on SDN Research*. ACM; 2019:36-48. https://dl.acm.org/doi/abs/10.1145/3314148.3314352
15. Yassin W, Abdullah R, Abdollah MF, Mas'ud Z, Bakhari FA. An IoT botnet prediction model using frequency based dependency graph: proof-of-concept. *Proceedings of the 2019 7th International Conference on Information Technology*. IoT and Smart City; 2019:344-352.
16. Vinayakumar R, Alazab M, Srinivasan S, Pham QV, Padannayil SK, Simran K. A visualized botnet detection system based deep learning for the Internet of Things networks of smart cities. *IEEE Trans Ind Appl*. 2020;56(4):4436-4456. https://ieeexplore.ieee.org/abstract/document/8985278
17. Joshi HP, Bennison M, Dutta R. Collaborative botnet detection with partial communication graph information. *2017 IEEE 38th Sarnoff Symposium*. IEEE; 2017:1-6.
18. Lyu M, Sherratt D, Sivanathan A, Gharakheili HH, Radford A, Sivaraman V. Quantifying the reflective DDoS attack capability of household IoT devices. *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. ACM; 2017:46-51. https://dl.acm.org/doi/abs/10.1145/3098243.3098264
19. Antonakakis M, April T, Bailey M, et al. Understanding the Mirai botnet. *26th USENIX Security Symposium USENIX Security 17*. USENIX; 2017:1093-1110. https://www.usenix.org/system/files/conference/usenixsecurity17/sec17-antonakakis.pdf
20. Sivanathan A, Gharakheili HH, Sivaraman V. Detecting behavioral change of IoT devices using clustering-based network traffic modeling. *IEEE Internet Things J*. 2020;7(8):7295-7309.
21. Sivanathan A, Gharakheili HH, Loi F, et al. Classifying IoT devices in smart environments using network traffic characteristics. *IEEE Trans Mob Comput*. 2018;18(8):1745-1759.
22. Murthy A. Implementation. GitHub. 2021 https://github.com/amithmurthy/MSc-Thesis-A-Lightweight-IDS-for-IoT-based-Smart-Building
23. Murthy A. Detecting IoT device compromise using Python. https://www.youtube.com/watch?v=qJAxYC4uu88list=PLBGl1tVyiWQT0Gq9NwrA2Rb7m16Sb_CRuindex=19t=4s
24. Kodinariya TM, Makwana PR. Review on determining number of cluster in K-means clustering. *Int J*. 2013;1(6):90-95.
25. Apthorpe N, Reisman D, Sundaresan S, Narayanan A, Feamster N. Spying on the smart home: Privacy attacks and defenses on encrypted IoT traffic. arXiv preprint, arXiv:170805044 2017.
26. Van Adrichem NL, Doerr C, Kuipers FA. Opennetmon: Network monitoring in OpenFlow software-defined networks. *2014 IEEE Network Operations and Management Symposium (NOMS)*. IEEE; 2014:1-8.
27. Trimananda R, Varmarken J, Markopoulou A, Demsky B. Packet-level signatures for smart home devices. *Signature*. 2020;10(13):54.

How to cite this article: Murthy A, Asghar MR, Tu W. A lightweight Intrusion Detection for Internet of Things-based smart buildings. *Security and Privacy*. 2024;e386. doi: 10.1002/spy2.386

24756725, 0, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/spy2.386 by Durham University - University, Wiley Online Library on [14/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License