Computer Communications 144 (2019) 8–17

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Identifying IoT devices and events based on packet length from encrypted traffic

Antônio J. Pinheiro [a,*], Jeandro de M. Bezerra [a,b], Caio A.P. Burgardt [a], Divanilson R. Campelo [a]

[a] Centro de Informática (CIn), Universidade Federal de Pernambuco (UFPE), Recife - PE, 50740-560, Brazil
[b] Universidade Federal do Ceará – Campus Quixadá, Quixadá - CE, 63902-580, Brazil

ARTICLE INFO

Keywords:
Traffic classification
Internet of Things
Encrypted data
Event identification

ABSTRACT

Recently, machine learning algorithms have been used to identify Internet of Things (IoT) devices and events. However, existing proposals may inspect the packet payload, what creates risks to IoT users' privacy, and may use several features, increasing the computational complexity for traffic classification. In addition, existing techniques may also use complex mechanisms for extracting traffic characteristics, including the creation of vectors containing data from the Transmission Control Protocol (TCP) sessions. This paper proposes a solution that uses packet length statistics from encrypted traffic to characterize the behavior of IoT devices and events in a smart home scenario. The solution uses only the statistical mean, the standard deviation and the number of bytes transmitted over a one-second window, which can be extracted from the encrypted traffic, making the use of TCP vectors unnecessary. The solution identifies IoT devices and events, such as voice commands to smart assistants, and also distinguishes between IoT and non-IoT devices. The solution to characterize IoT devices and events is evaluated with traffic from two real-world testbeds and five classifiers. The evaluation included the algorithms k-Nearest Neighbors (k-NN), Decision Tree, Random Forest, Support Vector Machine (SVM) and Majority Voting, some of the most popular algorithms for traffic classification. The results show that the Random Forest algorithm can achieve up to 96% of accuracy in the identification of devices, 99% of precision in distinguishing between IoT and non-IoT devices and 99% of accuracy in the identification of IoT device events. Hypothesis testing is used to validate the obtained results. Also, the results show that the Decision Tree presented the lowest latency among the five classifiers evaluated in the identification of the devices, followed by k-NN, Random Forest, SVM and Majority Voting.

1. Introduction

The Internet of Things is composed of a variety of networked devices with capabilities for sensing user presence on numerous domains, such as smart homes and cities, vehicles, industrial locations and hospitals. A smart home, which emerges as one of the main IoT domains [1], is typically equipped with devices such as smart bulbs, sleep monitors, personal assistants, motion sensors and security cameras, each one performing a different task.

Due to their quantity and variety, identifying IoT devices in a home network poses significant challenges to the administrator. The lack of visibility on the devices present in a smart home can result in security and performance issues [2]. In this way, the development of classification mechanisms is fundamental to the proper operation of a home network composed of IoT devices [2]. However, it is essential that traffic classification solutions preserve the privacy of users.

Existing proposals for traffic classification may inspect the packet payload, what creates privacy risks [3]. IoT devices generate Internet traffic with characteristic metadata patterns, which can be used in traffic classification to avoid payload inspection [4]. The metadata include, among other attributes, the packet length, the inter-arrival times, and the flow direction [3]. Existing proposals may use numerous traffic metadata, increasing their computational complexity and the consumption of resources to store the attributes [5]. Typically, home network devices (e.g., router and gateway) have fairly constrained computing resources [6]. Therefore, solutions for smart homes should avoid excessive resource utilization.

There is a variety of works on IoT device identification. The inspection of the payload of packets to extract the features used to identify them was used in [2,7–13]. The authors of [14–16] present mechanisms that analyze encrypted traffic. The mechanism presented in [14] uses a vector of traffic attributes with 276 dimensions (12 packets x 23 features), which imposes an excessive computational cost [17]. The mechanism presented in [15] requires 49 traffic attributes and 30,000 frames to identify devices, and long periods are required to capture

* Corresponding author.
E-mail addresses: ajp@cin.ufpe.br (A.J. Pinheiro), jmb@cin.ufpe.br (J.d.M. Bezerra), capb@cin.ufpe.br (C.A.P. Burgardt), dcampelo@cin.ufpe.br (D.R. Campelo).

https://doi.org/10.1016/j.comcom.2019.05.012
Received 2 January 2019; Received in revised form 17 April 2019; Accepted 17 May 2019
Available online 21 May 2019

traffic. For example, it takes at least 771 s to build the signature of a consumer device.

The authors of [16] propose the use of a time window with variable length to capture the transmission rate of consumer IoT devices. However, the authors of [16] do not inform the criteria used to set the window length. Even though, the time window values used in [16] were omitted, they are likely to be long since consumer IoT devices tend to generate small amounts of traffic and remain inactive during long periods. The solutions presented in [14–16] require too much memory to store the attributes, make the classification computationally costly, and make it challenging to the real-time device identification [17,18].

In this paper, we propose a solution to characterize consumer IoT devices present in smart homes that require low computational resources and preserve users' privacy. The proposed solution does not inspect the payload and uses only three traffic attributes. The premise is that consumer IoT devices produce traffic with packet length patterns that can be used to characterize the data source, even if they are encrypted [14].

We propose the use of a one-second window to enable real-time classification. Longer time windows increase memory consumption to store traffic attributes and make it difficult to classify traffic in real-time [19]. We investigate the existence of bias among different classifiers and also investigate the most relevant statistics for characterization of the normal behavior of consumer IoT devices. In this paper, normal behavior means the expected functioning of the device defined by its the vendor.

A prototype was implemented to identify IoT devices based on the proposed solution for traffic characterization. The prototype uses a three-stage process: (1) it distinguishes between IoT and non-IoT equipment; (2) it identifies specific IoT devices; (3) it identifies device events. A binary classifier is used to identify the device type — IoT or non-IoT. A multi-class classifier is used to determine the type of device, such as camera, motion sensor or personal assistant, when it is identified as being an IoT device by the binary algorithm. Finally, a multi-class classifier identifies the device event that generated the analyzed traffic.

The statistics used by the proposed solution are calculated from real-world traffic grouped in a one-second window. Also, we use algorithms based on Forest of Trees to analyze the contribution of each statistic to the identification of the devices. The main contributions of the paper are:

- A privacy-preserving solution for characterizing the normal behavior of consumer IoT devices;
- A consumer encrypted IoT traffic classification with low computational resource consumption.
- Quantification of the computational performance of classifiers in the identification of IoT devices.

The remainder of this paper is organized as follows. The motivations and use cases for IoT traffic characterization are described in Section 2. The description of the solution for IoT device and events identification is presented in Section 3. A description of the experiments that quantify the performance of the solution is presented in Section 4. Section 5 discusses the results obtained in the evaluation. The conclusions are presented in Section 6.

2. IoT traffic characterization: motivations and use cases

In general, solutions for consumer IoT traffic classification must meet some requirements, described as follows. Since consumer devices capture sensitive data from their users, who may wish their domestic activities not to be exposed to third parties [20], the solutions must be *privacy-friendly*. In addition, the solutions must be also *timely*, because consumer IoT traffic is dynamic and composed of short-lived sessions. Finally, since home networking devices (e.g., router or gateway) typically have limited computational resources such as memory and processing capabilities, the solutions for the characterization of consumer IoT traffic must be *constrained* in terms of the use of computational resources.

Unlike devices such as laptops and smartphones that run multiple applications in different combinations, specific-purpose IoT devices run particular applications, which induce them to generate traffic with characteristic patterns. Household devices from different types react to distinct events with distinctive traffic patterns. The set of events generated by each equipment contributes to characterize and distinguish it from others. The information type the device captures and the events it reacts to influence the patterns on the generated traffic. Moreover, consumer devices capture user information and traffic analysis can raise concerns about the privacy of individuals.

To avoid privacy violations, we investigate the use of encrypted traffic, what imposes significant challenges, but meets the requirements of user data protection. For example, it is impossible to extract traffic attributes from Transmission Control Protocol (TCP) sessions, a procedure widely used in traffic classification [21]. Because data is inaccessible due to the use of encryption, we analyze metadata for patterns that characterize IoT devices.

Network traffic exposes many metadata, but we must use the least possible to minimize the computational cost and time to characterize the devices. The most used metadata in traffic classification are the packet length and the inter-arrival time of frames. The length is more appropriate for device characterization because inter-arrival times are susceptible to network conditions such as congestion [22]. In this paper, we analyze only the packet length.

Individual packets provide insignificant contribution to characterize the devices, since different equipment can generate frames of the same length. As a result, it is necessary to analyze a set of packets for device characterization. Capture windows based on the number of packets (e.g., 25) [19] are ineffective in home networks because the devices generate different data quantities, what makes the capture process excessively complicated and requires too much memory to store the traffic attributes [23]. For example, it would take a few seconds to capture the camera-generated packets and long periods to get the same amount of frames from a motion sensor. Traffic attributes can be captured based on time windows [5].

Long time windows increase the classification delay and memory consumption to store the traffic attributes, what should make it difficult to classify traffic in real-time [5]. The authors of [9] identified that 95% of the IoT sessions lasted for up to 5 s. Too long windows can cause the device to be recognized after the end of the session. Thus, we investigated the use of short time windows to construct signatures from the packet length. Due to the presented assumptions, one-second windows were used in this paper.

Creating vectors with individual packets captured at one-second intervals lead to memory consumption to store data and complexity for classification models [5]. In this paper, we use statistics calculated from the packet length captured over one-second windows. Statistics are computationally less expensive to train and test classification solutions when compared to vectors containing individual packets [19]. To reduce the use of computational resources, we chose statistics that can be calculated through simple arithmetic operations: (1) the mean, which shows a good measure of the central tendency of the length; (2) the standard deviation, which gives the variation from the mean; and (3) the number of bytes, which shows the fluctuations in the data transmitted by the devices.

Besides the mean, we investigate the role of the median and the mode as a measure of central tendency in the characterization of consumer devices. However, due to the characteristics of consumer IoT traffic, these two measures would bring little contribution to characterize devices and their behaviors. Typically, consumer devices remain periods without interaction with users, when they only produce signaling traffic, such as TCP, Domain Name System (DNS), Network Time

Table 1
Mean, median and mode statistics of the packet length generated by the analyzed devices.

Device	Mode (*bytes*)	Mean (*bytes*)	Median (*bytes*)
Blipcare Blood Pressure meter	54	105	59
iHome plug	54	98	108
Smart Things hub	60	71	60
Samsung SmartCam	66	421	346
Amazon Echo	66	115	90
Withings Smart Baby Monitor	66	66	79
Belkin Wemo switch	66	366	118
Belkin wemo motion sensor	66	112	66
Withings Aura smart sleep sensor	66	138	66
Triby Speaker	66	104	66
PIX-STAR Photo-frame	66	111	78
HP Printer	86	134	86
Insteon Camera	102	104	90
Light Bulbs LiFX Smart Bulb	123	96	92
Dropcam	156	206	156
TP-Link Smart plug	172	107	66
Withings Smart scale	333	333	300
Netatmo weather station	350	171	113
NEST Protect smoke alarm	509	294	350
Netatmo Welcome	1510	471	86
TP-Link Day Night Cloud camera	1514	532	160

Protocol (NTP), and Internet Control Message Protocol (ICMP). These packets have standardized lengths for all devices, what makes these packets useless to differentiate the type of equipment or event that generated the traffic.

The signaling traffic can represent a significant amount of the transmitted packets of a device. As the mode shows the most frequent value, it is possible that this length comes from the signaling packets. The median has the same possibility of being affected by the signaling traffic. The impact of this traffic on the mean is small since this measure considers all packets, including the data frames that contribute to characterize the device.

From the Comma-Separated Values (CSV) files extracted from data made available by [9], packets generated by each device were separated and the mean, the median and the mode were calculated. The results of this experiment are presented in Table 1. As it can be seen from Table 1, the mean is the best measure to differentiate the devices because it varies among distinct equipment. Also, several devices share the same value for the mode and the median.

Some use cases for the characterization of consumer IoT devices are:

1. QoS (Quality of Service) policy enforcement: some devices (e.g., medical equipment) play a critical role in the well-being of users and can benefit from prioritized traffic. In a medical emergency, it is crucial that the data on healthcare devices take precedence;
2. Detection of anomalous behaviors: consumer IoT devices are susceptible to malware infections that alter their behavior. Detecting abnormal behaviors is essential to identify devices that should be sanitized or isolated;
3. Isolation of compromised devices: malware typically infects other devices on the local network from the compromised equipment. In home networks, the communication between IoT devices enhances the propagation of malware. Therefore, it is essential to isolate compromised devices to prevent malware from spreading across the home network [14].

The use cases described above require the identification of the devices present in the home network.

3. Solution for identification of devices and events

In this section, it is presented the proposed solution that uses a three-stage process for traffic classification: (1) it distinguishes between

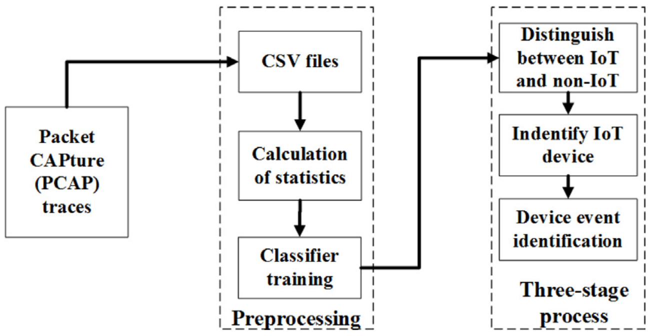

Packet CAPture (PCAP) traces
CSV files
Calculation of statistics
Classifier training
Preprocessing
Distinguish between IoT and non-IoT
Indentify IoT device
Device event identification
Three-stage process

Fig. 1. Execution flow of the proposed solution.

IoT and non-IoT devices; (2) it identifies specific IoT devices; and (3) it identifies device events. Because there are two device classes, a binary classifier trained with traffic from networked devices was incorporated to distinguish between IoT and non-IoT devices [7]. When the binary classifier identifies an IoT device, a multi-class algorithm is used to identify the specific device, such as a security camera, a smart assistant or a motion sensor. Lastly, a multi-class classifier identifies the events that induces the device to generate traffic. Fig. 1 illustrates the execution flow of the proposed solution.

The solution uses packet length statistics to identify IoT devices and events and to distinguish from non-IoT. The solution utilizes the mean packet length, standard deviation, and the number of bytes transmitted over a one-second window. These statistics were selected based on the analysis of IoT traffic provided by [9] and data from a testbed we configured. The use of only three statistics reduces the computational complexity for the IoT traffic classification, as the number of statistical features influences the cost of the identification process [3]. Also, it is unnecessary to group traffic into vectors, what simplifies the calculation of statistics.

The traffic from each device was grouped into one-second windows based on the timestamp [23,24]. Larger or variable windows do not improve the accuracy of the classifiers [16,25]. Also, the one-second interval allows faster identification, which is essential for real-time solutions [25]. Moreover, the devices analyzed in our testbed generate traffic for their events in one-second windows.

We collected and analyzed traffic from off-the-shelf devices connected to our testbed. These devices generated traffic with distinct patterns for each event. The same statistics were used to identify device events. The proposed solution can identify when users interact with their devices, e.g., an individual says the voice command “Alexa” to an Amazon Echo Dot, movements in the field of vision of a security camera, the turn on/off of devices, and so on.

Python scripts with the pandas library [26] were developed to calculate the packet length statistics. This library was also used to group the traffic generated by each device into one-second windows. This stage involved the calculation of the mean and standard deviation of the packet length, and the number of bytes in each group. After completing the above process, the data can be used to train and test the classifiers. 748,443 samples were utilized to train and test the solution abilities in IoT device identification.

The contribution of each statistic to the device identification was quantified using algorithms based on Forest of Trees [27]. These algorithms were used to build numerous decision trees, wherein each tree takes as input a subset of the 748,443 samples and a different combination of the three statistical features used by the solution. The results generated by the different decision trees allow the measurement of the importance, in percentage, of each statistic for the classification [27]. Fig. 2 illustrates the contribution of each statistic to the IoT traffic classification. This figure shows that the mean length is the most relevant statistic. However, there is no disparity about the contribution of the other statistics.

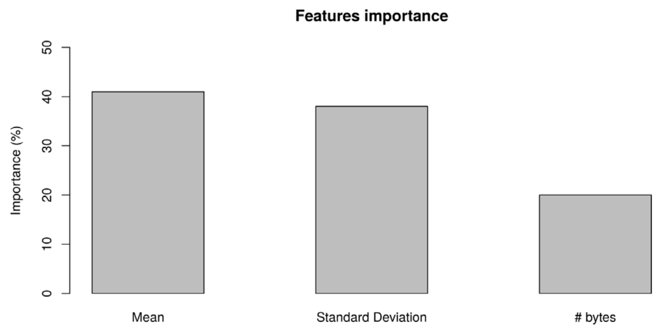


Fig. 2. Importance of the statistics in IoT device identification.

Despite the benefits offered by the proposed solution, there are some challenges to its implementation. Some of the main difficulties are firmware updates that change traffic patterns, compromised devices, need of re-training, and equipment unknown by the model. The scarcity of data for training is another limitation. However, the popularization of smart homes should make public datasets of smart home traffic more common. In future works, we intend to investigate the use of unsupervised learning to eliminate the need for re-training and human interaction to deal with unknown devices [13]. There are numerous cyber-security threats, such as a malware emulating the behavior of a device known by the models to claim some benefit. The proposed solution can be used to detect abnormal behaviors since it is capable of characterizing the expected operation of a device by its vendor.

4. Proposed solution evaluation

A prototype of the solution was evaluated in a virtual machine (VM) with 4 GB of Random Access Memory (RAM) Double Data Range (DDR), a processor Intel® Xeon® 2.20 GHz 64-bit and the operating system Ubuntu 16.04 Long Term Support (LTS). A virtual machine was used because of cost flexibility, deployment speed and easy replication.

The evaluation includes representatives of the main categories of smart home devices: smart assistant, security cameras, plug, bulb, hub and health care device. The analysis incorporated real IoT traffic collected by the authors of [9] in a testbed configured at the University of New South Wales.[1] This testbed is composed of 21 IoT devices and 7 non-IoT devices, and its configuration is detailed in [9]. In the experiments, we used three smart plugs, six security cameras and one device of each of the following types: personal assistant, smoke alarm, weather station, hub, smart scale, blood pressure meter, sleep sensor, light bulb, motion sensor, speaker, photo frame and printer. These devices come from 14 vendors. Table 2 shows the IoT devices, types of equipment and vendors present in the dataset used in the experiments.

Data generated by the non-IoT devices consist of video streaming and web traffic. IoT device data consists of autonomously generated traffic (e.g., Domain Name System (DNS) and Network Time Protocol (NTP)) and traffic from interaction with individuals (e.g., an Amazon Echo responding to voice commands). The authors of [9] made the captured traffic available in 20 Packet CAPture (PCAP) files.[2] A text file containing the Media Access Control (MAC) of the devices was made available. The MAC address distinguishes the traffic from each device.

The proposed solution was implemented in Python [28]. The Pandas library was used to compute the packet length statistics and Scikit-learn[3] [29] to implement classifiers and re-sample methods. Five classifiers were implemented: k-Nearest Neighbors (k-NN[4]), Random Forest, Decision Tree, Support Vector Machine (SVM), and the Majority Voting. The algorithms used in this paper are briefly described as follows:

- k-Nearest Neighbors (k-NN): computes the Euclidean distance of each test instance for the nearest k neighbors [17]. Advantages of k-NN relevant to our work [30]: (1) implementation simplicity, which should be useful for deployment on constrained devices (e.g., home router); (2) k-NN is suitable for classification with few features; (3) absence of parametric assumptions, rarely present in the real-world [17];
- Decision Tree: constructs a model based on a tree structure, where each node represents a feature verification, each branch the result, and the leaves are the *label* [31]. Advantages [30]: (1) implementation simplicity; (2) high accuracy in classification;
- Random Forest: model formed by a set of tree-based classifiers [27]. Advantages [30]: (1) it avoids over-fitting; (2) there is no need for selecting the features previously; (3) the variance of the model reduces as the number of trees increases, while the bias remains the same;
- Support Vector Machine (SVM): training data are divided into categories. Each sample from the test data is assigned to one of these categories [17]. Advantages [30]: (1) ability to generalize training data to identify unknown samples; (2) it minimizes the risk of classification error;
- Majority Voting: combination of multiple classifiers. Each classifier contributes with one vote, and the majority determines which label will be assigned to each test sample [31]. This algorithm combines the advantages of the previously mentioned classifiers.

We used multiple classifiers to investigate the bias in IoT traffic classification. Also, multiple classifiers contributed to minimizing the likelihood of obliquity in the results [31]. The classifiers belong to different categories – including nearest neighbors, trees, ensemble, vector machines, and voting – to elucidate the proposed solution performance with several types of algorithms. The training of the classifiers was conducted offline. The objective of the training was to assist the algorithms in identifying patterns that represent each device and events [31]. A testbed was configured to investigate IoT device events.

4.1. Testbed description and IoT traffic analysis

Three devices were used to set up an IoT testbed: one Amazon Echo Dot (2nd generation), one Amcrest security camera (IP2M-841B) and one TP-Link smart plug (HS100). These are the most challenging

Table 2
Devices used in experiments, their types, and vendors.

Type	Device	Vendor
Switches and plugs	Belkin Wemo switch	Belkin
	TP-Link Smart plug	TP-Link
	iHome plug	iHome
Camera	Netatmo Welcome	Netatmo
	TP-Link Day Night Cloud camera	TP-Link
	Samsung SmartCam	Samsung
	Dropcam	NEST
	Insteon Camera	Insteon
	Withings Smart Baby Monitor	Withings
Smart assistant	Amazon Echo	Amazon
Hub	Smart Things Hub	Samsung
Smoke alarm	NEST Protect smoke alarm	NEST
Weather station	Netatmo weather station	Netatmo
Smart scale	Withings Smart scale	Withings
Blood Pressure meter	Blipcare Blood Pressure meter	Blipcare
sleep sensor	Withings Aura smart sleep sensor	Withings
Light bulbs	LiFX Smart Bulb	LiFX
Motion sensor	Belkin wemo motion sensor	Belkin
Speaker	Triby Speaker	Imvoxia
Photo frame	PIX-STAR Photo-frame	PIX-STAR
Printer	HP Printer	HP

[1] The data was made available by the authors of the paper [9] published in IEEE INFOCOM 2017.
[2] Available at: http://149.171.189.1/.
[3] Scikit-learn is a suite of machine learning tools written in Python.
[4] Typically, 5 is the value for k (number of nearest neighbors) [17].

devices used in related works because they have the broadest set of events. More specialized devices – such as motion sensors and blood pressure devices – make the classification process easier, what discouraged us from considering them in this research. Also, the three devices revealed much more information about their users, what contributes to providing better services to individuals through classification.

Our testbed was configured based on [1,14]. One laptop with Intel Core i7 processor, 16 GB RAM, and Ubuntu 16.04 LTS was configured as an Access Point (AP) for the IoT devices [1,14]. The softwares hostapd (host access point daemon) and dnsmasq were configured on this laptop as an AP and connected it to the Internet through the wired interface. The IoT devices are connected to this laptop over the wireless interface. All IoT traffic goes to the Internet through this laptop. A second laptop equipped with a Core i3 processor, 4 GB of RAM and Windows 10 was used to access the camera's video streaming. We configured the IoT devices on a Motorola Moto G (2nd generation) smartphone. The TP-Link Kasa application was installed on this phone. The testbed was configured in a lab of Centro de Informática of Universidade Federal de Pernambuco.

Utilizing the laptop's wireless interface, a TCPdump captures the traffic generated by the IoT devices. This data was collected while interacting with the devices. The experiments were conducted for approximately 20 h, what enabled us to identify patterns generated by the devices for various events. We interact with the devices to generate the events investigated. Each event (e.g., Echo Dot listening to voice commands) was monitored in a distinct experiment. Each experiment was repeated 20 times while the traffic was captured and saved in PCAP files named in the format `device_event`.

4.2. Distinction between IoT and non-IoT devices

The objective of the experiment is to measure the performance of the solution in distinguishing between IoT and non-IoT devices. From the data available by [9], we extracted packet length statistics of non-IoT and IoT devices. We obtained 838.141 samples: 748,443 (89.30%) from IoT and 89,698 (10.70%) from non-IoT devices. The IoT traffic is the majority of the captured data because the testbed configured by [9] was used mainly for IoT traffic capture.

The captured traffic was divided into two classes, IoT and non-IoT, so that a binary classifier identified the device type. Since the traffic is highly imbalanced, we re-sampled the data by over-sampling the minority class and under-sampled the majority class [32]. Also, ensemble learning was used to independently re-sample subsets from the original dataset [32]. The following algorithms were used: Synthetic Minority Oversampling Technique (SMOTE) (over-sampling), RandomUnderSampling (under-sampling) and BalanceCascade (ensemble) [32].

Some of the files made available by [9] contain very little non-IoT data, what made testing classifiers practically unfeasible. So, we aggregated the 20 files provided by [9] and randomly divided the resulting dataset into training and test sets in the proportion of 75% and 25%, respectively. Also, we divided the aggregated dataset in chronological order: the first 75% of samples for training and the last 25% for testing. We re-sampled the training set; trained the classifiers with this data and used the test set to evaluate the classifiers [33]. Finally, we obtained the following metrics for the performance of the classifiers: precision, recall, F1-score, specificity and geometric mean [32,33]. The classifiers k-NN, Random Forest, Decision Tree, SVM and Majority Voting were applied.

4.3. IoT device identification

The objective of this experiment is to quantify the performance of the proposed solution for IoT device identification. Two 10-fold cross-validation procedures were used: (1) stratified with random partitioning, the same process used in [9,15,16]; (2) 10-fold with chronological partitioning and sliding window. The stratified k-fold cross-validation ensures that each fold preserves the proportion of the device classes and reduces the probability of over-fitting.

The chronological partitioning procedure is relevant in our experiments because we analyze the traffic attributes at fixed time intervals — one-second windows. In cross-validation with chronological partitioning, the dataset is divided for training and testing based on time sequences. For instance, in the first fold, the initial 4 h are for training and the next 2 h for testing. In the second fold, the initial 5 h are for training and the next 2 for testing.

The metrics included the accuracy, recall, F1-score, specificity and geometric mean, obtained with multi-class classifiers based on k-NN, Random Forest, Decision Tree, SVM, and Majority Voting. These classifiers were applied to the 20 files made available by [9], thereby, obtaining the average accuracy, recall, F1-score, specificity, and geometric mean from multiple subsets. This process contributes to avoiding bias in the results [34]. In total, 748,443 samples of IoT traffic were analyzed, a sample larger than the 35,000 recommended by [15]. The five classifiers were tested with each of the 20 files provided by [9].

The dataset available by [9] and used in the experiments is imbalanced. The dataset is imbalanced mainly due to the presence of traffic generated by cameras. Unlike the other devices present in the dataset, the cameras produce large volumes of traffic, capture data without interaction with users, and spend long active periods. Thus, the cameras create most of the data, which can imbalance the dataset used in the experiments.

Due to the imbalanced data, the cameras traffic was extracted from the dataset and an experiment was performed using only data from the other devices. Section 5.3 presents the results of this experiment. Since accuracy is a biased metric, recall and F1-score were used to mitigate this possible bias in the results. Also, specificity and geometric mean metrics that are appropriate for imbalanced classification were used [32,33].

4.4. Device events identification

This experiment was used to evaluate the solution in the identification of IoT events. The capture of the traffic generated by the IoT devices that compose the testbed is described in Section 4.1. The devices present in the testbed and the analyzed events are described below:

- Amazon Echo Dot: the device being in a idle state, a user says "Alexa" and, by means of a voice command, a smart plug is turned on/off;
- Amcrest security Camera: in an idle state, the camera captures audio and motion, and someone is watching what is being captured;
- TP-Link smart plug HS 100: in an idle state, the device is turned on/off by an Amazon Echo Dot or a Kasa app, a user enables/disables the away mode of the plug through the Kasa app, and when the away mode is enabled.

We captured and analyzed traffic to/from devices when performing the events mentioned above. Traffic generated by the Amazon Echo Dot and the Amcrest Security camera was captured because these devices produce data in response to events. The traffic received by the smart Plug was analyzed because this device receives commands from external entities, such as the Kasa App and Amazon Echo Dot. It was observed that the devices generate characteristic packet length patterns for the analyzed events, providing the ability to identify these patterns and map them to the corresponding events. Table 3 shows packet length patterns generated in some device events.

We have identified that all analyzed devices continuously transmit traffic to remote servers, even when they are idle. Most of this traffic is made up of DNS requests. The smart plug generates DNS queries every 10 s. The camera generates DNS requests for "config.amcrestcloud.com " every 2 s. When idle, an Amazon Echo Dot generates mainly DNS requests and TLSv1.2 packets of 95 bytes. These traffic patterns make

Table 3
Packet length patterns generated by IoT devices when performing some events.

Device	Event	Pattern (bytes/packet)
Echo Dot	Hear “Alexa”	491
Echo Dot	Turn on/off a plug	sequence: 304, 74, 74, 172
Camera	Motion detection	1514
Camera	Audio detection	258
Plug	Enable/disable away mode	sequence: 74, 66, 149, 66, 66
Plug	Kasa turn on/off plug	sequence: 74, 66, 115, 66, 66

it possible to identify when a device is idle. Much of the traffic is still transmitted in clear text. For example, packets used to alert about audio detection are transmitted in clear text and contain the string “AudioDetect” in JSON format.

The traffic produced by each event was saved in a separate PCAP file, which was later converted to CSV. From the CSV files, the packet length statistics produced by the analyzed events are calculated. Finally, the statistics train and test the five classifiers previously mentioned. Again, the accuracy, recall, F1-score, specificity, and geometric mean metrics quantify the performance of the multi-class classifiers in identifying events of the IoT devices.

4.5. Computation performance

This experiment is to measure the speed of the identification of IoT devices. The training time, the latency – time spent by device identification – and throughput – the number of identifications per second – were evaluated. The latency is a fundamental metric for the performance evaluation of the proposed solution [35]. A Python Script was implemented to evaluate the computational performance of the proposed solution. Again, we aggregate the files available in [9] and divided the resulting datasets into training and testing sets in the proportion of 75% (561,332 samples) and 25% (187,111 samples), respectively. The time spent by each classifier was quantified in the training phase. Last, an evaluation was conducted to identify the time required for each identification and throughput in the testing phase.

Besides obtaining good results in the classification, it is fundamental that the solution be fast [35]. However, no literature was found concerning the evaluation of latency within an IoT traffic classification system. Therefore, we were unable to compare these results with related works.

4.6. Hypothesis testing

We used hypothesis testing to validate the results for IoT device identification. Hypothesis testing reveals which classifier presented the best accuracy for the device identification. Results for recall and F1-score are very similar with respect to accuracy. We applied hypothesis testing on the device identification due to the multiple subsets from the 20 files available in [9]. Despite the relevance of a technique for comparing the performance of classifiers [34], no literature is available that applies hypothesis testing when validating the results of an IoT traffic classification.

Since the difference in the performance of the classifiers in the two partitioning methods is very small, the results obtained with the traditional random cross-validation were used in the hypothesis testing. Thus, the traditional random cross-validation is more appropriate for comparison with related works that apply hypothesis testing and random partitioning.

The Kolmogorov–Smirnov (KS) adherence test was applied to determine what type of hypothesis testing should be used: parametric (Gaussian) or non-parametric [36]. The KS test suggested the use of non-parametric hypothesis testing. Therefore, we use the Friedman, Nemenyi Post-Hoc and Wilcoxon Signed-Rank tests [34].

The Friedman‘s test verified the performance difference among the five classifiers. Then, the Nemenyi Post-Hoc Test was applied to identify which pairs of classifiers presented statistically different performances [34]. Finally, the Wilcoxon Signed-Rank Test (WSRT) was used to determine which algorithms obtained the best performance in device identification [34]. The confidence level was 95% for all tests.

Table 4
Performance of the classifiers in distinguishing between IoT and non-IoT devices with under-sampling.

Metric	Partitioning Method	k-NN	RF	DT	SVM	Majority Voting
Precision (%)	Random	96	97	96	96	96
	Chronological	93	95	95	91	95
Recall (%)	Random	94	95	95	93	94
	Chronological	93	95	95	91	95
F1-score (%)	Random	95	96	96	94	95
	Chronological	93	95	95	91	95
Specificity (%)	Random	94	97	96	98	98
	Chronological	93	95	95	91	95
Geometric mean (%)	Random	94	96	96	96	96
	Chronological	93	95	95	91	95

Table 5
Performance of the classifiers in distinguishing between IoT and non-IoT devices with over-sampling.

Metric	Partitioning Method	k-NN	RF	DT	SVM	Majority Voting
Precision (%)	Random	97	97	97	97	97
	Chronological	95	97	96	95	95
Recall (%)	Random	96	97	97	95	97
	Chronological	95	97	96	95	95
F1-score (%)	Random	96	97	97	95	97
	Chronological	95	97	96	95	95
Specificity (%)	Random	94	96	94	98	97
	Chronological	97	98	98	97	97
Geometric mean (%)	Random	95	96	95	97	97
	Chronological	96	97	97	96	96

5. Results

This section discusses the results obtained in the experiments used to evaluate the solution. The results were obtained in three different scenarios: (1) distinction between IoT and non-IoT devices; (2) traffic classification of 21 IoT devices; (3) identification of IoT device events. The results of these three scenarios are presented below.

5.1. Distinction between IoT and non-IoT devices

The results for the under-sampling, over-sampling and ensemble methods are presented in Tables 4, 5 and, Table 6, respectively. Each of these methods has disadvantages that can affect the results. One of the methods used to re-sample the data could benefit the proposed solution. Therefore, the importance of applying the three methods to avoid concerns about the evaluation results.

The results show that the performance of the solution is consistent among the three methods of re-sampling. Tables 4, 5 and 6 show a variation of at most five percent for all metrics, except specificity that reached 9% among the three re-sampling methods. These tables show that the performance of the solution is not favored by a specific re-sampling method or partitioning procedure.

5.2. IoT device identification

Table 7 depicts the solution performance for identifying IoT devices. The results demonstrate that the algorithms reached a performance above 90% in all the metrics when conducting an analysis with the proposed solution. These results suggest that the proposed solution can use the packet length patterns to identify the IoT devices.

Table 6
Performance of the classifiers in distinguishing between IoT and non-IoT devices with the ensemble method.

Metric	Partitioning Method	k-NN	RF	DT	SVM	Majority Voting
Precision (%)	Random	97	97	97	97	98
	Chronological	95	96	96	99	99
Recall (%)	Random	97	96	97	96	98
	Chronological	95	96	96	99	99
F1-score (%)	Random	97	96	97	97	98
	Chronological	95	96	96	99	99
Specificity (%)	Random	85	97	87	94	93
	Chronological	95	96	96	99	99
Geometric mean (%)	Random	90	96	92	95	95
	Chronological	95	96	96	99	99

Table 7
Performance of classifiers in IoT device identification on cross-validation with random and chronological partitioning.

Metric	Partitioning Method	k-NN	RF	DT	SVM	Majority Voting
Average accuracy (%)	Random	94	96	96	94	96
	Chronological	92	94	94	91	94
Average recall (%)	Random	94	96	96	94	96
	Chronological	92	94	94	91	94
Average F1-score (%)	Random	94	96	96	94	96
	Chronological	92	94	94	91	94
Average specificity (%)	Random	99	99	99	99	99
	Chronological	99	99	99	99	99
Average geometric mean (%)	Random	97	98	97	96	98
	Chronological	95	96	96	95	96

Table 8
Performance of classifiers in IoT device identification except cameras on cross-validation with random and chronological partitioning.

Metric	Partitioning Method	k-NN	RF	DT	SVM	Majority Voting
Average accuracy (%)	Random	92	94	94	92	94
	Chronological	90	93	92	90	93
Average recall (%)	Random	92	94	94	92	94
	Chronological	90	93	92	90	93
Average F1-score (%)	Random	92	94	94	92	94
	Chronological	90	93	92	90	93
Average specificity (%)	Random	99	99	99	99	99
	Chronological	99	99	99	98	99
Average geometric mean (%)	Random	95	97	96	95	97
	Chronological	94	96	96	94	96

5.3. Classification of devices except cameras

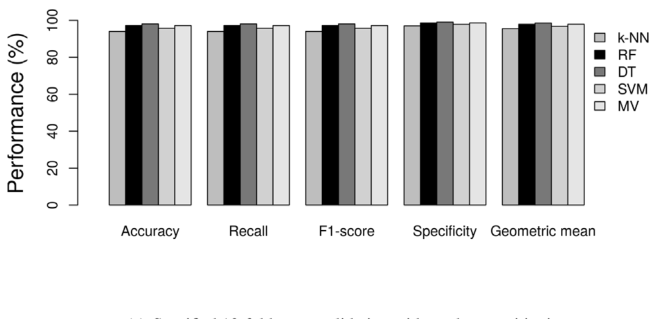
This section presents the performance of the proposed solution for the classification of all devices except cameras. This evaluation demonstrates the influence of the cameras on the performance of the solution. Table 8 presents the performance of classifiers to identify IoT devices. Table 8 shows a slight decrease in the performance of the solution compared to the data presented in Table 7. These results suggest that the packet length patterns generated by the devices, excluding the cameras, are similar enough to reduce the performance of the proposed solution. However, these results showed that the cameras have a limited influence since the performance is reduced by at most 2% on stratified random and chronological partitioning.

5.4. Device events identification

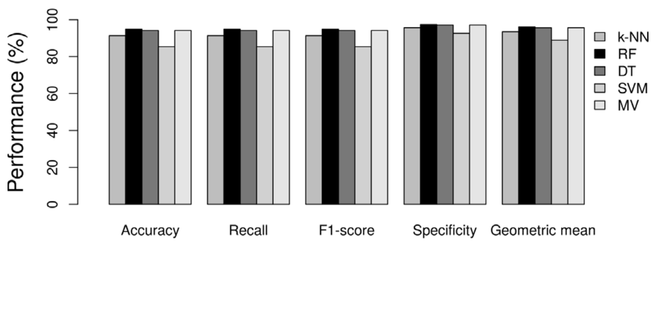
This section presents the results for the performance of the proposed solution in identifying events from an Amazon Echo Dot, an Amcrest security Camera and a TP-Link smart plug.

5.4.1. Amazon Echo Dot

An Amazon Echo Dot requires the user to say the keyword "Alexa" before a voice command. Therefore, the proposed solution identifies



(a) Stratified 10-fold cross-validation with random partitioning.



(b) 10-fold cross-validation with chronological partitioning.

Fig. 3. Solution performance in identifying when an Amazon Echo Dot hears "Alexa" or turn on/off plug commands under random and chronological partitioning, illustrated in Figs. 3(a) and 3(b), respectively.

when the Amazon Echo Dot hears this keyword or is required to turn on/off a smart plug. This step determines when an individual is interacting with the intelligent personal assistant. The solution utilizes a multi-class classifier, trained with traffic from the three classes (events): idle, "Alexa" keyword and turn on/off a smart plug. The performance of the solution is shown in Fig. 3. This figure illustrates how well the solution identifies the analyzed events.

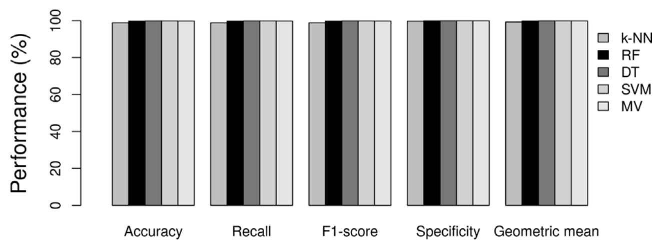
5.4.2. Amcrest security camera

The Amcrest security Camera analysis identified some exciting events: movements in the camera's field of vision; if someone is watching the camera streaming; when the camera is idle; and audio detection. Fig. 4 demonstrates the performance of the proposed solution in the identification of the various analyzed events of the Amcrest security camera. These results indicate it is possible to identify when someone is watching the video streaming from the camera.

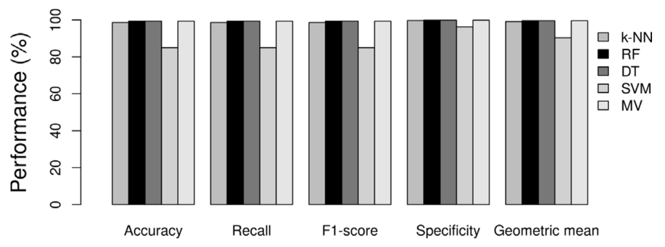
5.4.3. TP-link smart plug

The proposed solution identifies various events of the smart plug: idle; turn on/off with the Kasa app; turn on/off with the Amazon Echo Dot; enable/disable Away Mode; Away Mode in operation; and when the device was switched on/off by a Kasa/Echo/Away Mode. The performance of the solution in identifying these events is shown in Fig. 5. The TP-Link smart plug generates particular patterns for each event, which translates into the excellent performance presented by the solution.

We have identified that the smart plug produces the same traffic pattern to turn on or off, but this pattern varies depending on the source of this command. Fig. 6 displays the performance of the proposed solution in identifying when a smart plug switches on/off with the Amazon Echo Dot, Kasa app or the Away Mode. The proposed solution can identify when the On/Off command is initiated from the Amazon Echo Dot or Kasa app. With this information, you can infer that there is an Echo Dot device on the same network. The away mode of the

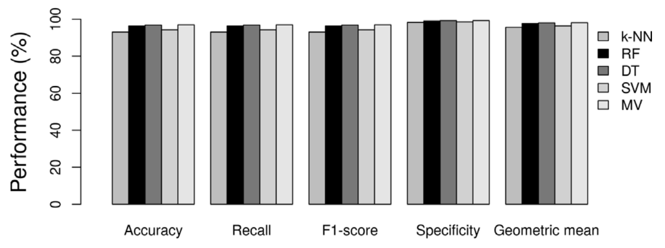


(a) Stratified 10-fold cross-validation with random partitioning.

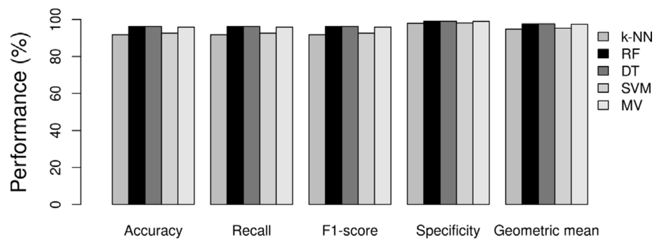


(b) 10-fold cross-validation with chronological partitioning.

Fig. 4. Solution performance in identifying camera events under random and chronological partitioning, illustrated in Figs. 4(a) and 4(b), respectively.



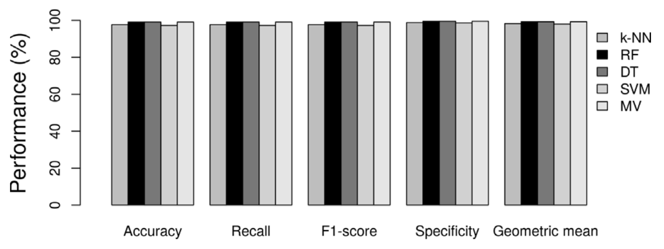
(a) Stratified 10-fold cross-validation with random partitioning.



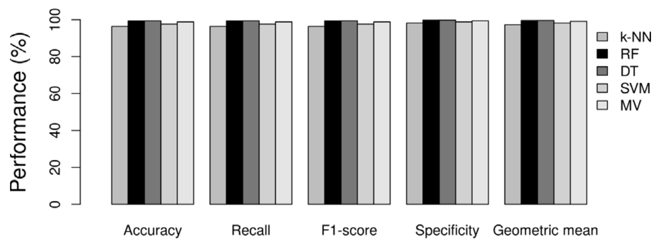
(b) 10-fold cross-validation with chronological partitioning.

Fig. 5. Solution performance in identifying smart plug events under random and chronological partitioning, illustrated in Figs. 5(a) and 5(b), respectively.

smart plug is intended to simulate the presence of individuals in a smart home. However, the proposed solution can identify when this mode is working and infer that the residence is empty.



(a) Stratified 10-fold cross-validation with random partitioning.



(b) 10-fold cross-validation with chronological partitioning.

Fig. 6. Solution performance in the distinction between traffic generated by an Amazon Echo Dot, Kasa App, and the Away Mode with random and chronological partitioning, illustrated in Figs. 6(a) and 6(b), respectively.

5.5. Statistical validation for IoT device identification

The hypothesis testing validated the results for device identification [34]. These tests are valuable for identifying the classification algorithm that presents the best performance in the two scenarios: All devices and except the cameras. The following hypotheses were formulated prior to conducting the Friedman's test:

$$H_0 : \text{There is no statistical difference among the classifiers.} \quad (1)$$

$$H_a : \text{There is a statistical difference among the classifiers.} \quad (2)$$

For the Nemenyi Test, the hypotheses are:

$$H_0 : \text{The performance of classifiers A and B does not differ significantly.} \quad (3)$$

$$H_a : \text{The performance of classifiers A and B differs significantly.} \quad (4)$$

The formulated hypotheses for the WSRT are:

$$H_0 : \text{Classifier } \mathbf{A} \text{ has a higher accuracy than } \mathbf{B}. \quad (5)$$

$$H_a : \text{Classifier } \mathbf{A} \text{ does not have better accuracy than } \mathbf{B}. \quad (6)$$

For the All Devices Scenario, we obtain a p-value = 7.203^{-15} with the Friedman's test. So, the hypothesis H_0 must be rejected because there is a statistical difference between the classifiers performance. The Nemenyi post-hoc test showed the following pairs of classifiers do not differ in their performances: Decision Tree and k-NN; SVM and k-NN; Majority Voting and Decision Tree; Majority Voting and Random Forest. The Wilcoxon Signed-Rank Test attests that the Random Forest classifier presented the best performance, followed by Majority Voting, Decision Tree, k-NN and, SVM.

For scenarios where security cameras were excluded, the Friedman's test presented a p-value = 3.311^{-13}. Therefore, we must reject the hypothesis that there is no difference in the performance of the classifier. The Nemenyi post-hoc test attested no difference in the performance of

Table 9
Computational performance of the algorithms regarding training time, latency, and throughput identification.

	k-NN	RF	DT	SVM	MV
Training time (s)	25	6.65	1.42	7665	7950
Latency (ms)	0.708	0.775	0.034	5.292	9.652
Throughput (identification/s)	1412	1290	29,411	188	103

the classifiers: Decision Tree, Random Forest, Majority Voting, k-NN, or SVM. The Wilcoxon Signed-Rank Test evidenced that the Random Forest, Majority Voting and Decision Tree classifiers showed better performance than k-NN and SVM.

5.6. Computational performance

In addition to achieving good performance in the identification of the devices, it is fundamental that the proposed solution be fast during the classification process. It is counterproductive to achieve high performance in the classification process if the identification task is very slow [35]. As a result, the proposed solution avoids packets inspection and only conducts the analysis with the three statistics: mean, standard deviation and number of bytes.

Table 9 shows the results for the computational performance of the proposed solution. Table 9 shows that the Decision Tree classifier provides the best results for all metrics. As expected, since four algorithms were combined, the majority voting showed the worst results. Unfortunately, we did not find papers wherein the computational performance of a solution for IoT traffic classification was evaluated. Therefore, it is not possible to compare these results with related works.

5.7. Interpretation of the evaluation results

The general interpretation of the results is the ability of the proposed solution to characterize the normal behavior of consumer IoT devices. This capability results in its performance in identifying different consumer devices, distinguishing between IoT and non-IoT and differentiating events generated by a home appliance.

The proposed solution demonstrated satisfactory performance in all the metrics evaluated. Results for the specificity and geometric mean suggest that the imbalance in the data available by [9] does not degrade the performance of the proposed solution. The performance of the proposed solution shows to be fine, even with the results in chronological order presenting a slight decrease compared to stratified cross-validation. Results show that the difference in the performance obtained with the two methods is practically negligible.

Since the results on device identification are relatively similar, some factors may influence the choice of a classifier: simplicity of implementation, the likelihood of bias and the speed in the identification of devices. As expected, the Random Forest classifier proved to be much faster than the Support Vector Machine.

6. Conclusion

This paper presents a solution for classifying encrypted IoT traffic based on packet length statistics. The proposed solution, incorporates three network traffic statistics to identify the devices and events: the mean and standard deviation of the packet length, and the number of bytes transmitted by each device. The statistics enabled the classification of encrypted traffic.

The solution was evaluated with the traffic of 21 IoT and 7 non-IoT devices made available by the authors of [9]. Also, we developed a testbed composed of three consumer IoT devices. The proposed solution achieved high performance in distinguishing IoT and non-IoT devices, obtaining a minimum precision of 96% on random partitioning and 93% on chronological order. The solution presented a performance above 94% of accuracy for IoT device identification on random partitioning and 91% on chronological order. Finally, realizing up to 99% of accuracy on both random and chronological partitioning, the proposed solution accurately identified the device events. The proposed solution is very fast and achieves a latency of only 0.034 ms with a Decision Tree Classifier.

Hypotheses testing based on Friedman, Nemenyi post-hoc and, Wilcoxon Signed-Rank tests attest that the Random Forest classifier is the best algorithm for IoT device identification with all evaluated scenarios. The Decision Tree is the fastest classifier. The Random Forest was the third fastest algorithm. The performance of Random Forest when compared to the other classifiers was expected, since it is an ensemble algorithm [35]. We identify a trade-off between accuracy and latency in IoT traffic classification.

The results of the proposed solution raise a number of research opportunities such as prioritization of IoT traffic, isolation of specific devices for security purposes, identification of abnormal events, and so on. The opportunities will be explored in future works.

Acknowledgments

This research is partially supported by FACEPE, Brazil, FUNCAP, Brazil and the Brazilian National Council for Scientific and Technological Development (CNPq), Brazil.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] S.M. Beyer, B.E. Mullins, S.R. Graham, J.M. Bindewald, Pattern-of-life modeling in smart homes, IEEE Internet Things J. (2018) 1–8, http://dx.doi.org/10.1109/JIOT.2018.2840451.

[2] A. Sivanathan, H.H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, V. Sivaraman, Classifying iot devices in smart environments using network traffic characteristics, IEEE Trans. Mob. Comput. (2018) 1, http://dx.doi.org/10.1109/TMC.2018.2866249.

[3] H. Kim, K. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, K. Lee, Internet traffic classification demystified: Myths, Caveats, and the best practices, in: Proceedings of the 2008 ACM CoNEXT Conference, ACM, New York, NY, USA, 2008, pp. 11:1–11:12, http://dx.doi.org/10.1145/1544012.1544023, URL http://doi.acm.org/10.1145/1544012.1544023.

[4] N. Apthorpe, D. Reisman, N. Feamster, A smart home is no Castle: Privacy vulnerabilities of encrypted iot traffic, in: Workshop on Data and Algorithmic Transparency (DAT), FTC, New York, NY, USA, 2016.

[5] W. Li, A.W. Moore, A machine learning approach for efficient traffic classification, in: 2007 15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2007, pp. 310–317, http://dx.doi.org/10.1109/MASCOTS.2007.2.

[6] M. Serror, M. Henze, S. Hack, M. Schuba, K. Wehrle, Towards in-network security for smart homes, in: Proceedings of the 13th International Conference on Availability, Reliability and Security, in: ARES 2018, 2018, pp. 18:1–18:8.

[7] Y. Meidan, M. Bohadana, A. Shabtai, J.D. Guarnizo, M. Ochoa, N.O. Tippenhauer, Y. Elovici, ProfilIoT: A machine learning approach for iot device identification based on network traffic analysis, in: Proceedings of the Symposium on Applied Computing, ACM, New York, NY, USA, 2017, pp. 506–509, http://dx.doi.org/10.1145/3019612.3019878, URL http://doi.acm.org/10.1145/3019612.3019878.

[8] Y. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N.O. Tippenhauer, J.D. Guarnizo, Y. Elovici, Detection of Unauthorized IoT Devices Using Machine Learning Techniques, arXiv, arXiv:1709.04647 URL http://arxiv.org/abs/1709.04647, 2017.

[9] A. Sivanathan, D. Sherratt, H.H. Gharakheili, A. Radford, C. Wijenayake, A. Vishwanath, V. Sivaraman, Characterizing and classifying iot traffic in smart cities and Campuses, in: IEEE INFOCOM Workshop on SmartCity: Smart Cities and Urban Computing, IEEE, Atlanta, GA, USA, 2017, pp. 1–6.

[10] M.R.P. Santos, R.M.C. Andrade, D.G. Gomes, A.C. Callado, An efficient approach for device identification and traffic classification in iot ecosystems, in: 2018 IEEE Symposium on Computers and Communications (ISCC), Natal, Rio Grande do Norte, BR, 2018, pp. 1–6.

[11] Z. Fki, B. Ammar, M.B. Ayed, Machine learning with internet of things data for risk prediction: application in ESRD, in: 2018 12th International Conference on Research Challenges in Information Science (RCIS), 2018, pp. 1–6, http://dx.doi.org/10.1109/RCIS.2018.8406669.

[12] F. Shaikh, E. Bou-Harb, J. Crichigno, N. Ghani, A machine learning model for classifying unsolicited iot devices by observing network telescopes, in: 14th International Wireless Communications and Mobile Computing Conference, IEEE, Limassol, Cyprus, 2018.

[13] V. Thangavelu, D.M. Divakaran, R. Sairam, S.S. Bhunia, M. Gurusamy, DEFT: A distributed iot fingerprinting technique, IEEE Internet Things J. 6 (1) (2019) 940–952.

[14] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A.R. Sadeghi, S. Tarkoma, Iot sentinel: Automated device-type identification for security enforcement in iot, in: 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), 2017, pp. 2177–2184, http://dx.doi.org/10.1109/ICDCS.2017.283.

[15] R.R. Maiti, S. Siby, R. Sridharan, N.O. Tippenhauer, Link-layer device type classification on encrypted wireless traffic with COTS radios, in: S.N. Foley, D. Gollmann, E. Snekkenes (Eds.), ESORICS 2017: 22nd European Symposium on Research in Computer Security, Proceedings, Springer International Publishing, Oslo, Norway, 2017, pp. 247–264, http://dx.doi.org/10.1007/978-3-319-66399-9_14.

[16] N. Apthorpe, D. Reisman, S. Sundaresan, A. Narayanan, N. Feamster, Spying on the Smart Home: Privacy Attacks and Defenses on Encrypted IoT Traffic, arXiv preprint arXiv:1708.05044, 2017.

[17] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., John Wiley & Sons, 2001.

[18] N. Williams, S. Zander, G. Armitage, A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification, ACM SIGCOMM Comput. Commun. Rev. 36 (5) (2006) 5–16.

[19] T.T. Nguyen, G.J. Armitage, A survey of techniques for internet traffic classification using machine learning, IEEE Commun. Surv. Tutor. 10 (1–4) (2008) 56–76.

[20] E.K. Choe, S. Consolvo, J. Jung, B. Harrison, J.A. Kientz, Living in a glass house: A survey of private moments in the home, in: Proceedings of the 13th International Conference on Ubiquitous Computing, in: UbiComp '11, ACM, New York, NY, USA, 2011, pp. 41–44, http://dx.doi.org/10.1145/2030112.2030118, URL http://doi.acm.org/10.1145/2030112.2030118.

[21] W. Li, M. Canini, A.W. Moore, R. Bolla, Efficient application identification and the temporal and spatial stability of classification schema, Comput. Netw. 53 (6) (2009) 790–809.

[22] A. Hajjar, J. Khalife, J. Díaz-Verdejo, Network traffic application identification based on message size analysis, J. Netw. Comput. Appl. 58 (2015) 130–143.

[23] R.R. Maiti, S. Siby, R. Sridharan, N.O. Tippenhauer, Link-layer device type classification on encrypted wireless traffic with COTS radios, in: S.N. Foley, D. Gollmann, E. Snekkenes (Eds.), Computer Security – ESORICS 2017, Springer International Publishing, Cham, 2017, pp. 247–264.

[24] R. Sridharan, R.R. Maiti, N.O. Tippenhauer, WADAC: Privacy-preserving anomaly detection and attack classification on wireless traffic, in: Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks, in: WiSec '18, ACM, New York, NY, USA, 2018, pp. 51–62, http://dx.doi.org/10.1145/3212480.3212495, URL http://doi.acm.org/10.1145/3212480.3212495.

[25] G.A. Ajaeiya, N. Adalian, I.H. Elhajj, A. Kayssi, A. Chehab, Flow-based intrusion detection system for SDN, in: 2017 IEEE Symposium on Computers and Communications (ISCC), 2017, pp. 787–793, http://dx.doi.org/10.1109/ISCC.2017.8024623.

[26] W. McKinney, Python for data analysis: Data wrangling with Pandas, NumPy, and IPython, O'Reilly Media, Inc, 2012.

[27] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.

[28] M. Lutz, Programming Python: Powerful Object-Oriented Programming, O'Reilly Media, Inc, 2010.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, J. Mach. Learn. Res. 12 (Oct) (2011) 2825–2830.

[30] A.L. Buczak, E. Guven, A survey of data mining and machine learning methods for cyber security intrusion detection, IEEE Commun. Surv. Tutor. 18 (2) (2016) 1153–1176.

[31] E. Alpaydin, Introduction to Machine Learning, second ed., in: Adaptive computation and machine learning, The MIT Press, 2010.

[32] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284, http://dx.doi.org/10.1109/TKDE.2008.239.

[33] Y. Tang, Y.-Q. Zhang, N.V. Chawla, S. Krasser, SVMs modeling for highly imbalanced classification, IEEE Trans. Syst. Man Cybern. B 39 (1) (2009) 281–288.

[34] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (Jan) (2006) 1–30.

[35] S.E. Gmez, B.C. Martnez, A.J. Snchez-Esguevillas, L.H. Callejo, Ensemble network traffic classification: Algorithm comparison and novel ensemble scheme proposal, Comput. Netw. 127 (2017) 68–80, http://dx.doi.org/10.1016/j.comnet.2017.07.018, URL http://www.sciencedirect.com/science/article/pii/S1389128617303079.

[36] D.C. Montgomery, Applied Statistics and Probability for Engineers, sixth ed., Wiley, 2014, URL http://gen.lib.rus.ec/book/index.php?md5=f20ec9ee406e4fd41ccca22082a3de52.