Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2022) 30: 1941 – 1961

doi:10.55730/1300-0632.3915

Packet-level and IEEE 802.11 MAC frame-level analysis for IoT device identification

Rajarshi Roy Chowdhury*, Azam Che Idris, Pg Emeroylariffion Abas
Faculty of Integrated Technologies, Universiti Brunei Darussalam, Jalan Tungku Link, Gadong, Brunei Darussalam

Received: 19.12.2021 • **Accepted/Published Online:** 30.05.2022 • **Final Version:** 22.07.2022

Abstract: In cyberspace, a large number of Internet of Things (IoT) devices from different manufacturers with heterogeneous functionalities are connected together. It is challenging to identify all these devices in an IoT ecosystem. The situation becomes even more complicated when the devices come from the same manufacturer and of similar types due to their analogous network communication behaviour. In this paper, a device fingerprinting (DFP) approach based on a set of combined features from packet-level and frame-level has been proposed. A large number of features has been studied, and consequently, a suitable subset of features has been selected according to gain-ratio and device-specific features for DFP. Furthermore, experiments with different types of IoT devices in a laboratory environment to collect network traffic traces have been conducted and used to evaluate the performance of the proposed approach using the J48 (C4.5) algorithm. It has been shown that the proposed model is able to identify individual device types with 99.0% precision and 98.9% recall with the approach capable of classifying IoT devices coming from the same manufacturer and of similar types, with higher accuracy. These results are significant as it can be used as a security reinforcement tool towards increasing the security and resilience of IoT networks.

Key words: Network packet analysis, MAC frame analysis, device fingerprinting (DFP), probe request frame, Internet of Things (IoT)

1. Introduction

The rapid proliferation of heterogeneous IoT devices with distinct functionalities has imposed new security and privacy challenges in cyberspace, such as device management [1–3], anomaly detection [4, 5], security rules enforcement [6, 7], authentication [8, 9], attack detection [10, 11], faulty device identification [12], and location tracking [13]. To mitigate these issues, device identification plays a key role in IoT networks. Devices can be identified explicitly; using user-defined identifiers, such as internet protocol (IP) and media access control (MAC) addresses [14, 15], or implicitly; based on network traffic analysis [6, 16–18], and using radio frequency [19], which are used for their communication.

In a conventional wired or wireless network, devices are commonly identified using either their IP/MAC addresses to allow communication over the network or the Internet [20, 21], with these addresses ensuring logical and physical addresses of a particular device, respectively [22]. However, from the security perspective, both addressing schemes have some constraints and limitations as device identifiers [23], due to their vulnerability to different attacks, including spoofing attacks [24, 25], MAC address randomization [25, 26] and network-

*Correspondence: 19h0901@ubd.edu.bn or rajarshiry@gmail.com

dependent IP addressing scheme (device mobility or dynamic host configuration protocol (DHCP) protocol) [23]. Spoofing is a type of cyber-attacks, which allows a device (or a user) to access illegitimate services or resources on the network by masquerading as a legitimate entity. Some companies, including Apple Inc., have incorporated MAC address randomization technique into their devices to maintain anonymity and privacy [27]. There do exist some software packages; including Cisco-Snmp-Tool [28], and Mac-Changer [29], that have been made available on Windows and Linux based operating systems (OS) to perform penetration testing on such vulnerabilities, these solutions are not fool-proof against all vulnerabilities.

Device fingerprinting (DFP) is one of the techniques that may be used to identify devices without any explicit identifiers, either using active or passive fingerprinting approaches [23]. Different types of features, including network traffic packet header information [1, 6, 16, 17, 30, 31], payload [1, 32], traffic flows [1, 17, 31, 33], hash value [4, 34], interarrival time (IAT) [27, 33, 35], statistical measurements [27, 33, 36, 37], and raw radio signal [8, 38, 39] features, may be used to generate device-specific signatures. Some features extraction processes require expensive hardware and software for sophisticated signal processing. An example of this is the extraction of raw radio signal features (in-phase and quadrature (I/Q)). On the other hand, some set of features can be easily extracted using commodity hardware and software tools, such as the use of commercial-of-the-shelf USB WiFi adapter used to capture network traffic traces. These also need to be taken into consideration when designing a device identification method.

IoT devices identification is a challenging task, especially if the devices come from the same manufacturer and of similar types [4, 6, 16, 30, 40]. A DFP model based on a combination of feature set from wireless local area network (WLAN) communication traffic traces (packet-level and frame-level), has been proposed as identifiers in this paper. To test the efficiency of the proposed feature set, they are used as input for training and testing a machine learning (ML) classification algorithm, i.e. J48. It has been shown that the proposed DFP scheme is able to improve the classification accuracy of individual IoT devices in IoT networks, despite the devices coming from the same manufacturer and of similar types. An experimental set-up using 8 D-Link IoT devices was established to collect a total of 2.53 GB of network traffic traces from both packet and frame levels, which were then extracted and used to analyze a set of features from these captured traffic traces for evaluating the proposed DFP model. Experimental results have signified that the proposed scheme can achieve 99.0% precision and 98.9% recall, respectively. In brief, the key contributions of this research work are:

- A DFP model for similar types of IoT devices from the same manufacturer based on a combination of two levels (packet and frame levels) of feature set has been proposed to improve individual IoT device classification performance. The scheme improves IoT device identification precision up to 99%.
- Characteristics of key feature set from both packet-level and frame-level features, including *udp.srcport, ip.ttl, tcp.window_size, wlan.fcs*, etc., have been analyzed to ascertain features, which are device-specific.
- Deployment of an experimental testbed with a total of 8 D-Link IoT devices from the same manufacturer and of similar types, i.e. camera, to collect and synthesize data. The dataset [41] has been made publicly available for the research community, which includes both packet and frame levels communication traffic traces.

The remainder of this paper is structured as follows: Section 2 describes related works and existing DFP approaches. Experimental design, datasets, data collection process, and packet-level and frame-level features analysis along with the proposed device fingerprinting model are given in Section 3. Section 4 discusses

evaluation results with different datasets. Finally, the conclusion and future direction of work of the paper is presented in Section 5.

2. Related work

Most of the schemes have been developed based on analysis of network traffic traces, by utilizing either passive or active measurement of traffic traces. In the context of traffic profiling, features are used for DFP, which may be broadly categorized into two forms: packet-level [16, 30, 42] and frame-level (IEEE 802.11 MAC frame) [25, 43] features. For frame-level analysis, probe request frame is a preferable choice to be used for device identification. There are two reasons for this: i) probe request frames are transmitted by only WiFi-enabled devices, which are in abundance especially when considering IoT networks, and ii) the information carried in a probe request frame is commonly in the form of plain text [25].

Miettinen et al. [6] developed an automated IoT device identification scheme to enforce security in an IoT network by using network traffic analysis. Twelve consecutive packets' information from different layers of the communication model: link, network, transport, and application layers, have been used to construct a 276-dimensional feature vector, in order to generate a unique set of fingerprints for the individual device type. The scheme then uses the set of fingerprints to train a random forest (RF) classification model per device type, however, it is necessary to train a new model for every new device type. Overall, this approach achieves 81.50% accuracy using the testbed dataset of 27 out of 31 IoT devices. The relatively low accuracy of the proposed technique is due to the presence of multiple IoT devices coming from the same manufacturer. Sivanathan et al. [17] proposed an IoT device identification framework based on a statistical analysis of network traffic characteristics. All the captured network traffic traces are converted into flows by using the Joy tool on an hourly basis to compute statistical attributes, such as device activity and signalling patterns. The proposed framework is designed in two stages, with each multivalued attributes processed in Stage-0 in the form of bag of words (BoW) to generate two outputs using naive Bayes (NB) classifiers. Then, these outputs are fed onto the next stage (Stage-1) along with quantitative attributes from the traffic flows, to train the RF classifier model to be used for device identification. The reported accuracy of the scheme was over 99% from the testbed dataset of 28 IoT devices. On the other hand, Radhakrishnan et al. [35] utilized statistical assessment of IAT values for fingerprinting to identify wireless devices and their types from wire-side observations of network traffic traces. The method incorporates both active and passive analysis of application-specific network traffic. Statistical assessment for counting IAT values frequency distribution falling in the range of 300 equally spaced time bins is performed for signature (histogram) generation. These signatures are then used to train multilayer feed-forward artificial neural networks (ANNs) for classification, with the scheme evaluated on an isolated testbed and a live campus network using a total of 37 devices. Typically, the method achieves higher classification accuracy in the known device (seen) analysis mode as compared to the unknown (unseen) device analysis mode.

Robyns et al. [43] presented a mobile device fingerprinting approach based on probe request frame per bit entropy analysis using three metrics: variability, stability, and suitability of a bit, to compute suitable bitmask (bit patterns) for device fingerprinting. The scheme has achieved accuracy between 67.6% and 80.0% for a small dataset in the range of 50 to 100 devices, while for a large dataset of 1000 to 10,000 devices, precision varies from 15.1% to 33.0%. In reference [44], a total of 19 fields (as features) among 53 fields in the IEEE 802.11 probe request frame is used to generate device fingerprints. These feature set are selected based on the correlation and integrity score of each field. Statistical analysis of the selected set of features is performed to construct a unique signature pattern for a device type. Then, a similarity score is measured between stored

devices fingerprints and observed fingerprints to identify a device type. The scheme has been evaluated using two datasets: the Sapienza/probe-requests dataset, and a testbed of 300 devices types. Experimental result has shown that 95% accuracy is achievable for identifying individual device type using this model. Similarly, Neumann et al. [45] measured five distinct network parameters including transmission rate, frame size, medium access time, transmission time, and frame IAT for identification of 802.11 standard device types. A histogram per frame type based on the frequency distribution of the five parameters are generated, with each histogram weight used as a signature for the individual device type. Two types of tests: similarity and identification, are performed to evaluate these selected parameters by using four different wireless traffic traces collected from the Sigcomm conference (2008) and an experimental testbed. It has been observed that transmission time and frame IAT parameters performed better as compared to other parameters.

It has been observed that most of the existing DFP approaches have been designed to classify devices either using packet-level or MAC frame-level features. Researchers [4, 6, 16, 30, 40] in the same domain have identified that their proposed DFP models performances decline significantly when multiple devices come from the same manufacturer. Therefore, deep analysis of communication traffic traces from both levels is required to improve classification accuracy even when devices come from the same manufacturer and of similar types. Additionally, a dataset with similar types of IoT devices from the same manufacturer is required for analyzing to generate unique fingerprints, which includes both packet and IEEE 802.11 MAC frame levels communication traffic traces. No dataset, which includes both levels of traffic traces, has been found publicly available online.

3. Experimental methodology

3.1. Datasets and data collection

Several datasets have been used for the analysis of network traffic traces, from both packet and frame levels. For packet-level features analysis, two publicly available datasets: IoT Sentinel [6] and UNSW [17] datasets have been used. On the other hand, a publicly available Glimps [43] dataset, which was collected from a music festival in Ghent, Belgium, has been used for the frame-level (probe request frame) features analysis. In the Glimps dataset, only devices (unique MAC address) which have a minimum of two instances (probe request frames) are considered; giving a total of 26,648 unique MAC addresses with multiple frames. Additionally, an experimental dataset, D-Link IoT [41] dataset, has also been collected and made publicly available online, for both packet-level and frame-level feature analysis. Specifications of the different datasets are given in Table 1. The D-Link IoT dataset has been captured from an experimental laboratory testbed at Universiti Brunei Darussalam (UBD); with only 8 D-Link IoT devices (2 HD WiFi Camera model no. DCS-936L, and 6 Wireless N Network Camera model no. DCS-930L) from the same manufacturer and of similar types.

Table 1. List of datasets.

Dataset	Devices	Packets	Frames	Source
IoT Sentinel	31	1,02,347	–	[6]
UNSW	22	68,45,378	–	[17]
Glimps (hasselt/glimps2015)	26,648	–	1,22,989	[43]
D-Link IoT	14	46,64,130	1,22,173	[41]

Experimental design for the network traffic collection process of the IoT devices is depicted in Figure 1. The devices have been configured to connect to a specific access point (AP), i.e. AP-1 (interface mode:

Master), using their WiFi interface for communication over the network. A laptop, running Ubuntu-v.18.04 as host operating system (OS) and Kali Linux-v.2019.3 as guest OS over VMware workstation player-v.15.5.2, functions as the access point. An external USB WiFi adapter attached to the system has been used as a WiFi interface for the Kali Linux. The laptop gets its internet services using the Ethernet interface through the UBD network. A local server is connected to the same network for data storing and processing. The *tcpdump* (open-source packets analyzer) package is utilized to capture wireless network traffic traces (inbound and outbound communication traffic) passively. During the data capture process, traffic traces have been filtered according to their MAC addresses and recorded in separate packet capture (pcap) files. To automate data collection and storing processes, shell scripts that automatically execute daily at midnight (local time at 12:00 AM) have been utilized, by using the *cron job* (time-based job scheduling).

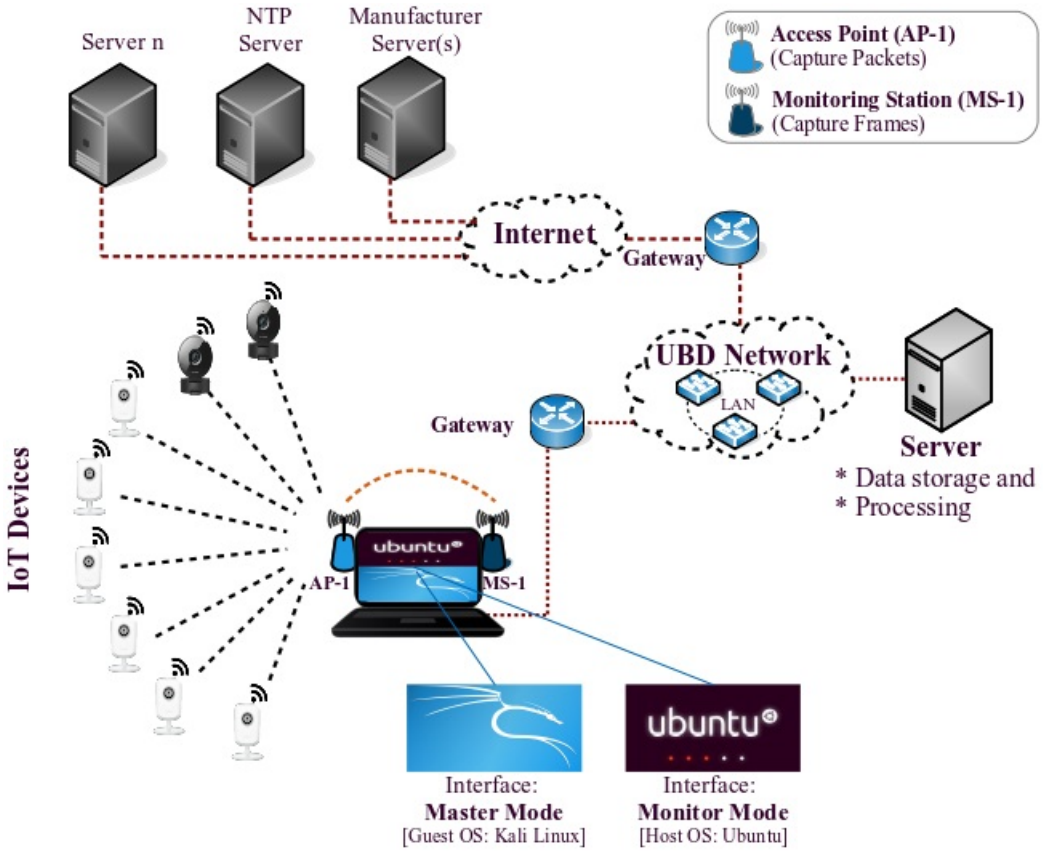


Figure 1. Experimental design of the IoT devices network traffic collection process.

3.2. Data preprocessing and feature extraction

Both network packets and frames have been filtered according to the individual device MAC addresses and then, feature values extracted using the *Tshark* [46] utility, to give a comma-separated values (CSV) file, with device name used as a label. The *pcapfix* utility has been employed to repair any possible damage or corrupted pcap files in the dataset. For frame analysis, the *TShark* utility has been used to derive feature values from individual probe request frame features. To prepare the dataset for training and testing of an ML classification model, inconsistent data have been removed from both datasets (packet and frame datasets).

All the selected packet and frame levels features are extracted from the typical behaviour of the IoT devices during network communication, with these feature set categorized into two levels: packet header features (network, transport, and application layers protocols header information), and IEEE 802.11 MAC frame features (probe request frame). There are a total of 212 [30] and 380 features (unsigned integer) extracted from packet headers and probe request frames, respectively, with brief descriptions of these features presented in reference [46]. Since the D-Link IoT devices are specific-purpose devices, these devices generally use less number of tagged parameters in their probe request frame as compared to mobile devices [43]. Consequently, there are only a total of 107 features that have values in the probe request frames of the D-Link IoT dataset.

3.3. Feature analysis

Both packet-level and frame-level features from the passively observed network traffic traces of the eight D-Link IoT devices listed in Table 2 have been analyzed. Only network traffic features that can be extracted using a standard wireless card, without resorting to any expensive equipment, have been considered. A wide range of feature set has been studied on both levels, in order to identify a suitable subset of features that vary significantly from device to device, and hence, may be used to uniquely identify IoT devices.

Analysis of packet header features:- Figure 2 provides a visual representation on the total number of packets transmitted by devices in the D-Link IoT dataset, over a 24 days period. It is observed that packets distribution among the devices varies, despite the devices coming from the same manufacturer and of similar type. The top 20 features among the feature set, in terms of the number of unique feature values from all the devices, are shown in Figure 3, with *tcp.time_relative* giving the most number of unique values.

Table 2. List of D-Link IoT devices.

IoT device	Model	MAC address	IoT device	Model	MAC address
D-LinkCam1_a4	DCS-936L	b2:c5:54:44:0f:a4	D-LinkDayCam3_8f	DCS-930L	b0:c5:54:3d:3f:8f
D-LinkCam2_11	DCS-936L	b2:c5:54:44:0f:11	D-LinkDayCam4_a6	DCS-930L	b0:c5:54:42:8f:a6
D-LinkDayCam1_5d	DCS-930L	b0:c5:54:46:48:5d	D-LinkDayCam5_e5	DCS-930L	b0:c5:54:42:8f:e5
D-LinkDayCam2_93	DCS-930L	b0:c5:54:3d:3e:93	D-LinkDayCam6_88	DCS-930L	b0:c5:54:42:8f:88

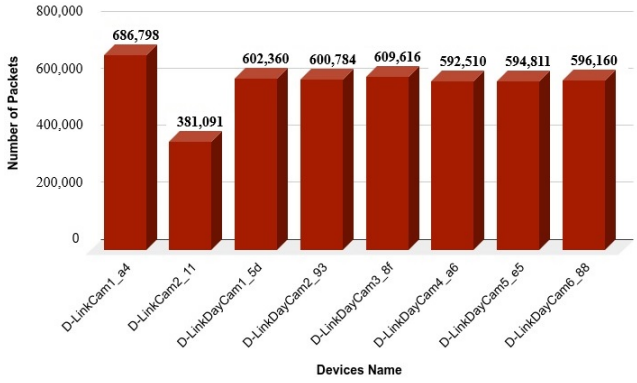


Figure 2. Packet distributions pattern of the D-Link IoT devices.

The top twenty features among the same feature set from the IoT Sentinel [6] and UNSW [17] datasets have also been identified in Figure 4, where it can be observed that most of the features in the top 20 list from

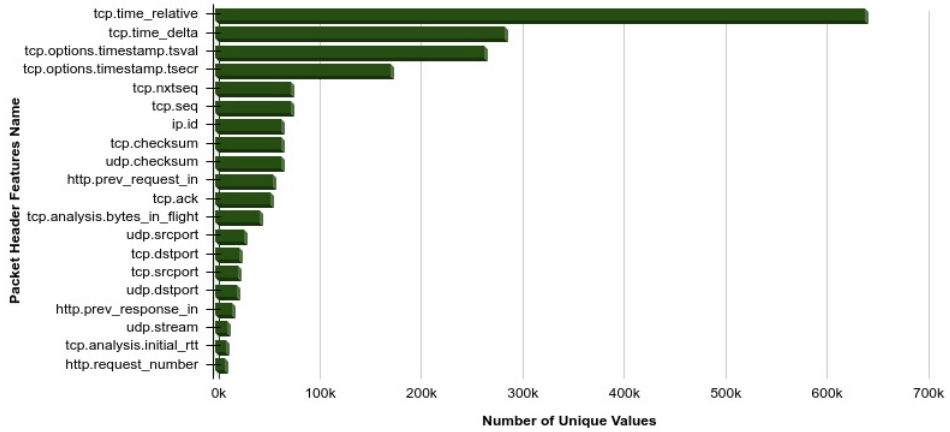


Figure 3. Total number of unique values per feature.

the IoT Sentinel and UNSW datasets, are similar to the feature list from the D-Link IoT dataset. The top 16 features from the IoT Sentinel dataset and the top 17 features from the UNSW dataset, are in the top 20 features list from the D-Link IoT dataset, despite differences in features rankings among the datasets. It is further noted that *udp.dstport* and *http.request_number* features, are only present in the D-Link IoT dataset top twenty feature list. Subsequently, the selected set of features has been further investigated, to identify the total number of unique values of a particular feature for a particular device as compared to similar features of other devices. It is noted that each device has a group of distinct feature values within the same set of features, as presented in Table 3, which may be used to uniquely identify the device. Some of the significant set of packet header features are as follows:

UDP source port numbers (*udp.srcport*): Each IoT device uses a handful of user datagram protocol (UDP) source port numbers in the range of 0 to 65,535 for acquiring different services over the network. Figure 5 shows that D-LinkCam1_a4 used the maximum number of unique source port numbers (3666) as compared to all other devices from the same manufacturer, with D-LinkDayCam3_8f and D-LinkDayCam5_e5 individually utilizing only six particular port numbers.

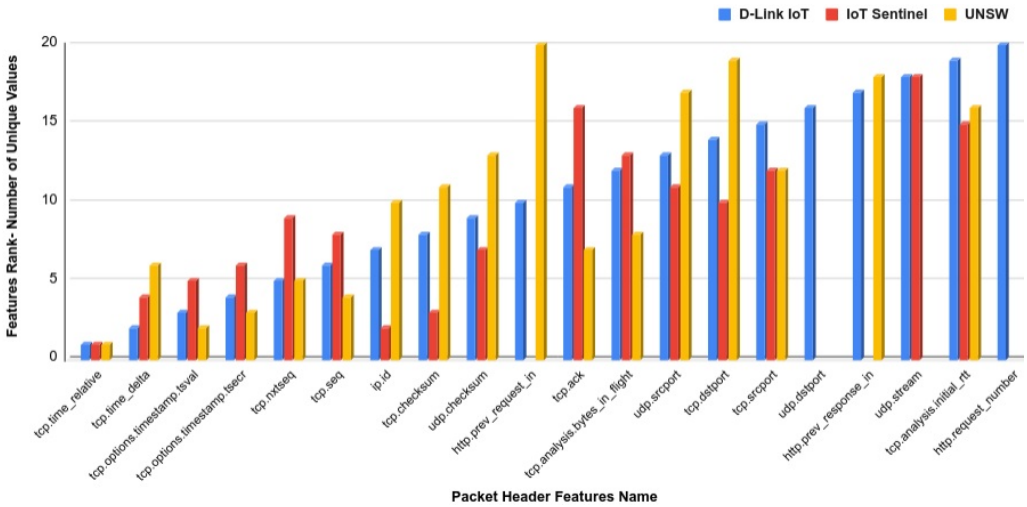
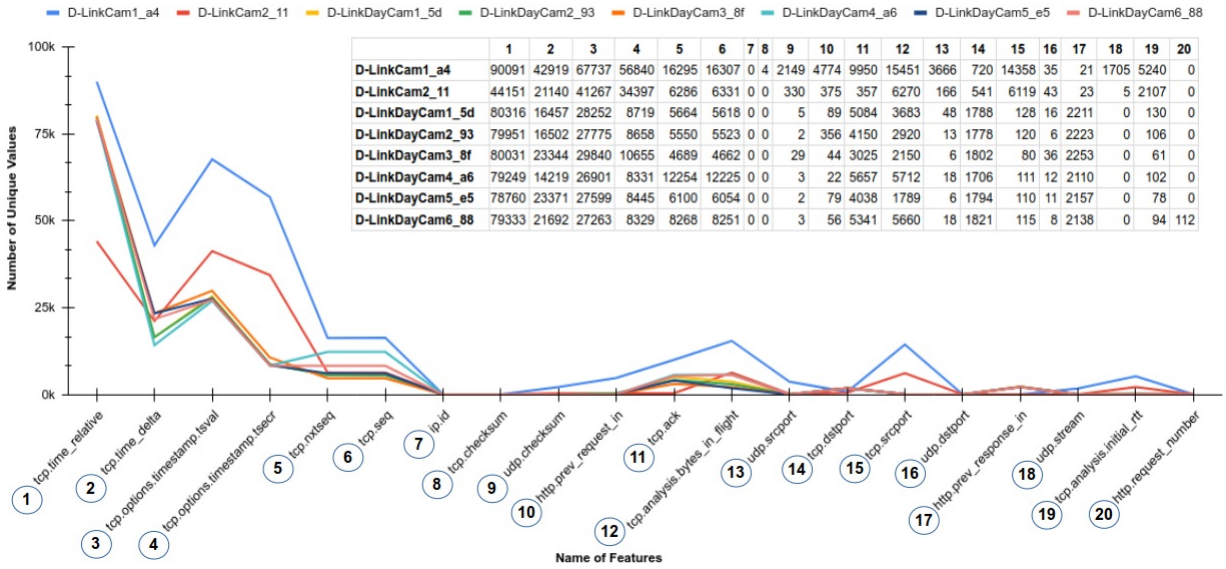


Figure 4. Top 20 features of experimental datasets.

Table 3. A set of unique number of feature values.

Features	D-LinkCam		D-LinkDayCam					
	1_a4	2_11	1_5d	2_93	3_8f	4_a6	5_e5	6_88
udp.srcport	3666	166	48	13	6	18	6	18
udp.checksum	2149	330	5	2	29	3	2	3
udp.dstport	35	43	16	6	36	12	11	8
ip.id	0	0	0	0	0	0	0	0
tcp.seq	16,307	6331	5618	5523	4662	12,225	6054	8251
tcp.ack	9950	357	5084	4150	3025	5657	4038	5341
tcp.time_delta	42,919	21,140	16,457	16,502	23,344	14,219	23,371	21,692
http.prev_response_in	21	23	2211	2223	2253	2110	2157	2138
tcp.analysis.bytes_in_flight	15,451	6270	3683	2920	2150	5712	1789	5660



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
D-LinkCam1_a4	90091	42919	67737	56840	16295	16307	0	4	2149	4774	9950	15451	3666	720	14358	35	21	1705	5240	0
D-LinkCam2_11	44151	21140	41267	34397	6286	6331	0	0	330	375	357	6270	166	541	6119	43	23	5	2107	0
D-LinkDayCam1_5d	80316	16457	28252	8719	5664	5618	0	0	5	89	5084	3683	48	1788	128	16	2211	0	130	0
D-LinkDayCam2_93	79951	16502	27775	8658	5550	5523	0	0	2	356	4150	2920	13	1778	120	6	2223	0	106	0
D-LinkDayCam3_8f	80031	23344	29840	10655	4689	4662	0	0	29	44	3025	2150	6	1802	80	36	2253	0	61	0
D-LinkDayCam4_a6	79249	14219	26901	8331	12254	12225	0	0	3	22	5657	5712	18	1706	111	12	2110	0	102	0
D-LinkDayCam5_e5	78760	23371	27599	8445	6100	6054	0	0	2	79	4038	1789	6	1794	110	11	2157	0	78	0
D-LinkDayCam6_88	79333	21692	27263	8329	8268	8251	0	0	3	56	5341	5660	18	1821	115	8	2138	0	94	112

Figure 5. Number of unique values among all the devices.

UDP checksum (*udp.checksum*): Figure 5 shows that D-Link IoT devices use distinct numbers of unique UDP checksum values. From the dataset, it can be seen that the DCS-936L model devices have more unique values as compared to the DCS-930L model devices. For instance, the D-LinkCam1_a4 (DCS-936L) used a total 2149 specific values for its *udp.checksum*, whereas the D-LinkDayCam2_93 and D-LinkDayCam5_e5 used only two distinct values individually.

UDP destination port numbers (*udp.dstport*): IoT devices from the same manufacturer share a certain number of destination or server ports [17], to acquire their required services. It has been observed that some standard port numbers, such as port number 53 (for DNS protocol), port number 123 (for network time protocol (NTP) time synchronization protocol), and port number 1900 (for simple service discovery protocol (SSDP) advertising and network services discovery protocol), were shared among the D-Link IoT devices. However, these devices also shared a list of nonstandard port numbers, including port numbers 3088 and 3091. Figure 5 shows that each device utilized a limited number of unique destination port numbers; for instance,

D-LinkCam2_11 exploited a maximum of 43 discrete port numbers for its communication.

IP identification (*ip.id*): The IP identification field mainly assists the network layer fragmentation and reassembly of datagrams, to allow reduction of datagrams duplication. Generally, *ip.id* feature has a set of unique values, although these values may be replicated among the devices. Hence, unique values are rare for particular individual devices, especially when dealing with a large number of packets, as depicted in Figure 5.

TCP sequence number (*tcp.seq*): The transmission control protocol (TCP) is a connection-oriented protocol, whereby all the bytes or segments are numbered sequentially for each connection. Whilst individual device chooses initial sequence number (ISN) in the range of 0 to (2^{32} – 1) during connection establishment, it then follows a successive procedure for communication. It has been observed that the D-Link IoT devices have several distinct sequence numbers when compared to one other, as shown in Figure 5. For instance, D-LinkCam1_a4 used 16,307 unique sequence numbers in its communication, while 4662 unique sequence numbers were used by the D-LinkDayCam3_8f device.

TCP acknowledgment (*tcp.ack*): In the TCP header fields, the TCP acknowledgement number (*tcp.ack*) defines the next sequence number (*tcp.nxtseq*) of an expected packet to be received from the intended client (or server). These two features are interconnected with each other. From the dataset, each D-Link IoT device has a significant number of unique values for this feature, as shown in Figure 5. For instance, D-LinkCam1_a4, D-LinkDayCam1_5d, and D-LinkDayCam6_88 devices had a total number of 9950, 5084, and 5341 unique values, respectively.

TCP Time delta (*tcp.time_delta*): This timestamp is measured from the time elapsed between two successive packets, i.e. *tcp.time_delta* (current packet) = *tcp.time_relative* (current packet) - *tcp.time_relative* (previous packet), within the same TCP stream. From Figure 5, it is noted that all the devices had a significant number of distinct values for this feature, which consequently may be used as distinguishable features for the device.

Previous HTTP response packet number (*http.prev_response_in*): This feature determines the prior response start packet number of hypertext transfer protocol (HTTP) protocol, a commonly used protocol in IoT networks. Figure 5 shows that according to the device model, devices had approximately similar number of unique values for this feature. The DCS-930L model devices had a unique number of values range between 2110 and 2253. However, this feature is not a suitable feature to be used for device identification, as the feature follows a fixed pattern for all the devices and is not probabilistic.

Bytes in flight (*tcp.analysis.bytes_in_flight*): This feature represents the amount of data (or bytes) that has been sent by a TCP stream (or bytes) sender without receiving an acknowledgement (ACK) [47]. The total size of bytes is less than or equal to the recipients receive window size (or TCP window size); with the TCP stream receivers managing the window size by calculating the available buffer size and informing accessible window size to the corresponding transmitter in an ACK packet [48]. It has been observed that this feature relates to the devices, and individual devices that have a significant number of unique values can be used for distinguishing devices, as shown in Figure 5. For instance, D-LinkCam1_a4 had a total of 15,451 specific values for this feature whilst the rest of the devices had between 1789 and 6270 unique values.

Analysis of MAC frame features: Frame analysis presents an intensive view of wireless network infrastructure by scanning access points (or a specific access point) within range. It can be performed without associating with any particular AP. In this study, probe request frames have been analyzed, where in a WLAN, the client station utilizes this frame for scanning available networks in its range, in order to initiate a connection

for communication. The probe request frame is a subtype of management frames, which is sent intermittently (either to a specific access point or broadcast) by most of the WiFi-enabled devices for scanning available networks to associate with. Additionally, IoT device also transmits probe request frames even after it has associated with an AP, with distinct patterns in associate and unassociate states [49]. From Figure 6, it is shown that the devices have distinct network scanning patterns.

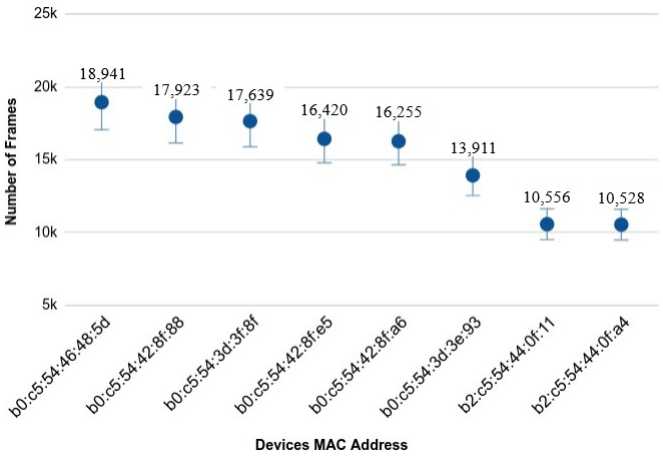


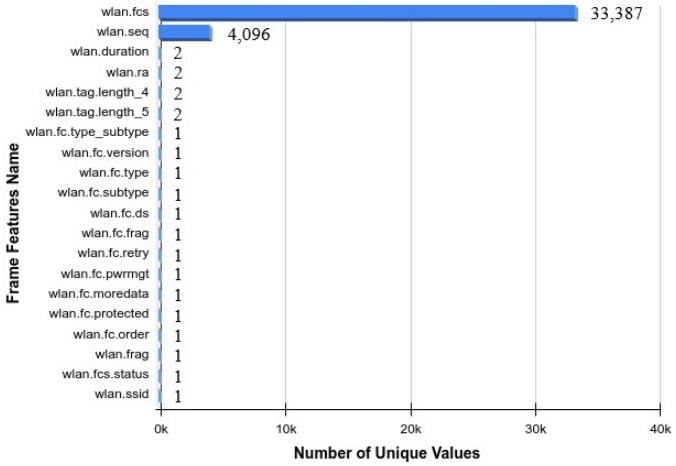
Figure 6. Frames distribution pattern of the devices.

It is observed that the DCS-930L model devices periodically broadcasted probe request frames to request information. On the other hand, the DCS-936L model devices infrequently transmitted probe request frames after they had completed association to a specific AP. In the probe request frame, information elements (IEs) field values are embedded explicitly in its body, containing information about wireless client stations. IEs contain various tagged parameters, such as service set identifier (SSID) parameter set, supported rates, extended supported rates, high throughput (HT) capabilities, vendor specific, interworking, robust security network (RSN) information, AP channel report, direct sequence (DS) parameter set, extended capabilities, and very high throughput (VHT) capabilities [50].

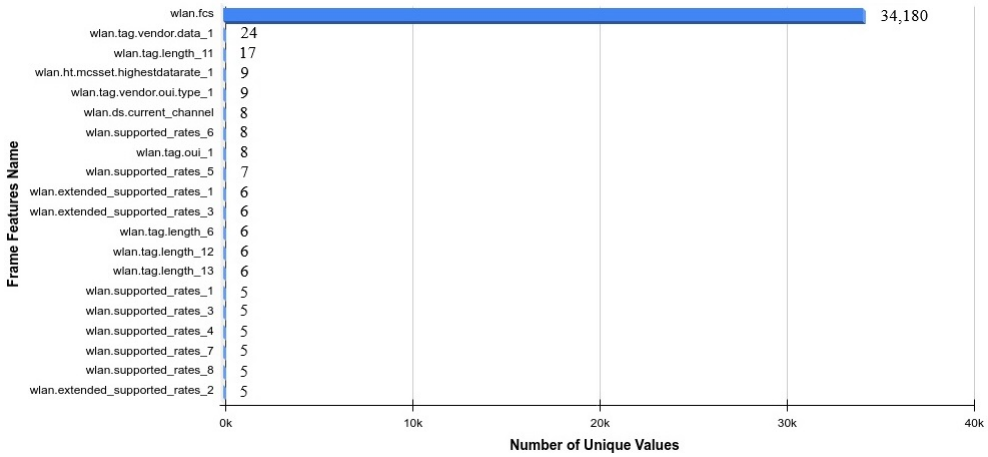
Since these IEs are optional and vary from device to device, the set of tagged parameters also varies according to the device configuration and capabilities. In the testbed dataset, it is noted that the DCS-930L devices had used a total of 5 tagged in their probe request frame, including SSID parameter set, supported rates, extended supported rates, HT capabilities and vendor specific, whilst the DCS-936L model devices used only 4 tagged parameters (except HT capabilities). In Figures 7a and 7b, it can be observed that the top 20 frame features vary significantly among two datasets: D-Link IoT and Glimps, however, the *wlan.fcs* feature is ranked top in both datasets. The reason is that the D-Link IoT frame dataset consists of data from only 8 devices of the same manufacturer, whilst the Glimps dataset comprises data from a few thousand mobile devices coming from multiple manufacturers. Key feature set from the D-Link IoT devices probe request frames have been further explored:

Wireless LAN frame check sequence (*wlan.fcs*): An IEEE 802.11 MAC frame comprises a set of three fields: MAC header, frame body, and frame check sequence (FCS). The FCS field is an error-detecting code, which uses a cyclic redundancy check (CRC) algorithm to generate a 32-bits value. Whenever a device receives a frame, it compares the calculated FCS value to the received FCS value in the frame to ensure the integrity of the frame [51]. In the dataset, it has been observed that *wlan.fcs* had a maximum number of unique

values as compared to any other feature values, as shown in Figure 7a, which suggest that it may be utilized for device identification.



(a) D-Link IoT dataset.



(b) Glimps dataset.

Figure 7. Top 20 frame features: (a) D-Link IoT and (b) Glimps datasets.

Wireless LAN sequence number (*wlan.seq*): The sequence number field in a MAC frame consists of 2 bytes and it is a sequential counter in the range of 0 to 4095 (each counter number is the modulo of 4096 [52]), with value increasing incrementally for each nonfragmented frame (or whenever a frame is sent out). By default, the fragment number is zero except if the frame is a fragment of a large packet [53]. This field is similar to the IP identification field in an IP header, which is used for reassembling fragments of a MAC frame [53]. From Figure 7a, *wlan.seq* had the second highest unique number of values from the D-Link IoT dataset.

Duration (*wlan.duration*): In a MAC frame, *wlan.duration* indicates the amount of time (in microseconds) required for successfully transmitting a frame over a specific channel. Additionally, this feature is used as the association identifier during the association process [52]. Its value varies according to the type of frames transmitted during communication. In the D-Link IoT dataset, this feature only had a limited number of unique values.

Receiver address (*wlan.ra*): The IEEE 802.11 MAC frame header comprises different address fields according to the types of frames, with *wlan.ra* defining the MAC address of the next instantaneous recipient of the frame. In the frame dataset, a probe request frame contains a similar MAC address in the following three features: *wlan.ra, wlan.da*, and *wlan.bssid*. As the experimental dataset had been captured in a laboratory setting using a dedicated AP, *wlan.ra* feature had a minimal number of unique values.

Analysis based on gain-ratio:- In addition to the feature analysis above, the features are also analyzed in terms of gain-ratio $GR(S, Fx)$ of the x^{th} feature F_x.

$$GR(S, F_x) = \frac{H(S) - \sum_{j=1}^{f_{nx}} \frac{|F_{xj}|}{|S|} \times H(F_{xj})}{-\sum_{j=1}^{f_{nx}} \frac{|F_{xj}|}{|S|} \times log_2 \frac{|F_{xj}|}{|S|}}, \tag{1}$$

where f_{nx} is the total number of distinct values of a feature F_x, $|F_{xj}|$ defines the number of instances corresponding to devices in the current split of the dataset. $H(S)$ and $H(F_{xj})$ are the entropy of dataset S, and feature unique value F_{xj}, defined as

$$H(S) = -\sum_{i=1}^{C_n} \frac{fr(cls_i, S)}{|S|} \times log_2 \frac{fr(cls_i, S)}{|S|}, \tag{2}$$

where $fr(cls_i, S)/|S|$ gives the probability of each device i and unique feature values, according to cls_i frequency in the given dataset S. C_n represents the number of unique devices, where C_n is 8 for the D-Link IoT dataset, and $|S|$ is the total number of instances in the dataset S.

Gain-ratio rectifies the biases of a feature F_x with the maximum number of unique values and measures the significance of the feature according to proportional information helpful for classification [54]. Device-specific features, which may be suitable for distinguishing devices, would have a high gain-ratio. On the other hand, gain-ratio is zero or close to zero, if a feature carries insignificant device-specific information. Table 4 lists some features with higher and lower gain-ratio values from the packet-level features, with the feature set sorted based on their computed gain-ratio.

Table 4. Samples of gain-ratio values with some selected feature set from the packet-level.

Feature	Gain-ratio	Feature	Gain-ratio
udp.srcport	0.4350897504966603	*ip.len*	0.2338027494292048
udp.checksum	0.1848400109047611	*udp.stream*	0.1777247226346733
tcp.analysis.initial_rtt	2.9367282965867854E-4	*http.time*	0.0

3.4. Classification model

Packet-level and frame-level features may then be used to train an ML classifier for device identification. A J48 (C4.5) classifier, which is an extended version of the iterative dichotomiser 3 (ID3) classifier and has been widely used for data mining [54], has been chosen for this purpose. It produces a decision tree based on information theory for the classification approach [55], and uses the tree pruning technique for reducing the misclassification error, as well as the greedy divide and conquer approach for making decision trees recursively for classification [56]. The J48 (C4.5) algorithm is the most widely used ML algorithm by the research community in data mining

[57], and has been used for various purposes, including for IoT devices classification [16, 30], intrusion detection [58, 59], and tumour classification in the medical science. References [16, 30] have shown that J48 performs better as compared to other classification algorithms.

3.5. Device fingerprinting framework

Our previous analysis has suggested that not all features are equally significant to be used as DFP. Some features are more device-specific and unique, such that when used as input for classification purpose, would allow more accurate identification of devices in an IoT network. However, as can be seen, features need to be extracted and processed before they can be used as input to a classifier model. Furthermore, the more features are used as input to a classification model, complexity would also increase. As such, a balance needs to be struck, such that the model would give an acceptable accuracy in identifying devices, whilst at the same time, only limited and importantly, carefully selected features are used as input to the classification model.

A device fingerprinting framework has been proposed in this paper, as depicted in Figure 8. Intuitively, similar types of IoT devices from the same manufacturer have analogous network behaviour most of the time, and as such, are more challenging to classify. Traffic traces are filtered from packet-level and frame-level, which practically increase the feature space, and allow the extraction of a set of features that can be used for DFP. Packet-level features comprise of the network layer (IP and internet control message protocol (ICMP) protocols), transport layer (transmission control protocol (TCP) and user datagram protocol (UDP) protocols), and application layer (DNS, HTTP, and transport layer security (TLS) protocols) protocols information (packet header), while frame-level features are extracted from individual tagged parameters of the probe request frames.

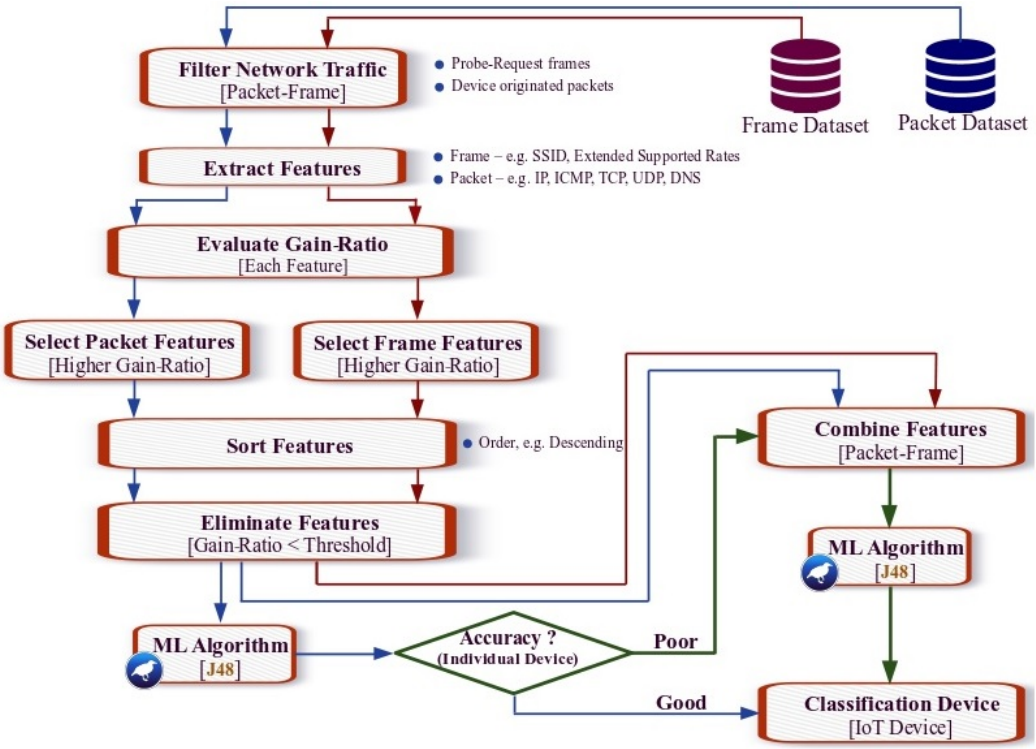


Figure 8. IoT devices identification framework.

m packet-level features are empirically selected based on their gain-ratio and used as device fingerprints. These features are then used as input to an ML algorithm, i.e. J48, for classification during training and testing, to achieve optimal accuracy for the identification of individual devices. Additionally, n frame-level features are also empirically selected based on their gain-ratio and used as device fingerprints. Then, a combined set of features, with m packet-level and n frame-level features, is used input to a J48 ML algorithm for classification, during both training and testing. A 5-fold cross-validation method has been used for analyzing the proposed model performances. Accuracy of the classification algorithm during testing is used as a performance measure for different input features to the classification model, including input feature set obtained using the proposed DFP method.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

where true-positive (TP) represents the total number of positive instances correctly classified, true-negative (TN) represents the total number of negative instances correctly classified, false-positive (FP) represents the total number of negative instances incorrectly classified, and false-negative (FN) represents the total number of positive instances incorrectly classified.

4. Results and discussion

4.1. System hardware and software details

The performance of the proposed device fingerprinting model has been evaluated on a Linux machine, using a Java-based benchmark data mining software, i.e. waikato environment for knowledge analysis (WEKA) [59]. This state-of-the-art software can be used for different purposes, including data preprocessing, clustering, data visualization, classification, and regression tasks. The hardware and software (H/S) details of the machine are presented in Table 5.

Table 5. Hardware and software devices.

Item	H/S details	Item	H/S details
Laptop	HP 2000 Notebook	Central processing unit (CPU)	Intel Core i5 (3rd Gen.)
Data mining tool	Weka–v.3.8.5	Hard disk drive (HDD)	500GB HDD
Operating system (OS)	Ubuntu–v.18.04.1	Random access memory (RAM)	2 × 8GB DDR3

4.2. Performance evaluations

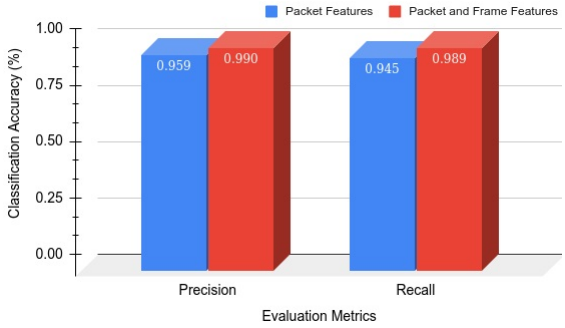
To assess the performance of the proposed DFP model on the different datasets, the feature set is fed as input to a J48 ML algorithm by utilizing the Weka tool [59]. $m = 24$ packet-level features and $n = 4$ frame-level features have been empirically obtained as the optimum. The IoT Sentinel dataset is a relatively challenging set of data since it comprises of devices set up network traffic only, and the authors [6] have not captured

the inclusive behaviour of IoT devices for their DFP approach. It consists of 31 IoT devices from different manufacturers. The proposed scheme in reference [6] achieves 81.5% accuracy over 27 devices. In reference [30], SysID achieves an average classification accuracy up to 82% using 23 devices from the IoT Sentinel dataset, whilst the proposed scheme in reference [4] gives better classification precision. On the other hand, the presented method in reference [60] attains identification accuracy of 90.3% over the 27 devices from the IoT Sentinel dataset, using 67 statistical features extracted from the header and payload information of 20–21 consecutive packets from the communication traffic of a particular device. In this paper, 212 packet header features have been extracted from the IoT Sentinel dataset according to individual packet header information, to generate individual device fingerprints for identifying devices. Initially, a suitable subset of features for DFP has been selected using the proposed method in reference [16], to reduce complexity, and improve classification accuracy. From the experimental result, it has been shown that individual device classification performance achieves 81.4% accuracy (with device originated 1,02,347 packets from 31 devices) based on 160 selected subset of features. Furthermore, the proposed DFP model (utilizing only packet-level features) has been evaluated utilizing the same dataset [6], using the J48 ML algorithm for classification. It achieves 83.9% accuracy by using only twenty-four packet header features from a single TCP/IP packet. This is in contrast to reference [60], which requires at least 21 packets before it can extract features. Additionally, the proposed DFP model preserves users' data privacy, since payload information is not considered a feature.

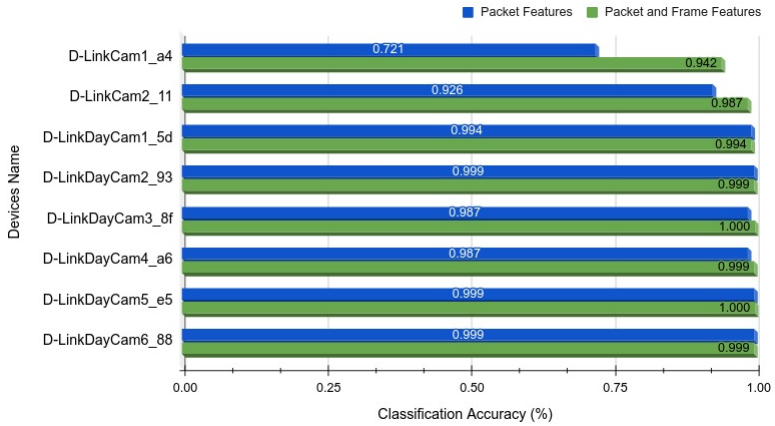
However, using the combined IoT Sentinel and D-Link IoT datasets (a total of 39 (31 + 8) IoT devices) the performance decreases significantly based on the selected 24 features, due to the presence of IoT devices from the same manufacturer, i.e. D-Link, and of similar types in both datasets. One-day network traffic of 8 D-Link IoT devices from the experimental testbed dataset has been integrated according to the number of instances in the IoT Sentinel dataset, to assess the proposed DFP method. The performance of the D-LinkCam device declines from 98.2% using the IoT Sentinel dataset only, to 84.5% for the integrated dataset, whilst performance of the D-LinkDayCam reduces from 84.5% to 79.8%. Similarly, the D-Link IoT dataset has also been incorporated with the UNSW dataset [17]. The proposed DFP model achieves an average classification accuracy over 97%, due to the unavailability of D-Link IoT devices in the UNSW dataset, whilst over 97% accuracy is obtained using the same UNSW dataset with 21 devices. The higher accuracy on the combined D-Link IoT and UNSW datasets can be attributed to the IoT devices coming from different manufacturers. Similarly, the Glimps [43] dataset has been used to assess the performance using entirely frame-level features. By using a subset of only 100 mobile devices from Glimps dataset, the J48 ML algorithm can classify over 83% of instances correctly, while on the incorporated dataset (Glimps and D-Link IoT datasets) with 108 devices, accuracy increases slightly to around 87% using a total of 818 (738 frames from the Glimps dataset + 80 frames from the D-Link dataset = 818 frames) probe request frames. This may be because of the set of features varies significantly between these two datasets. In the Glimps dataset mobile devices have a maximum of 12 tagged parameters (380 features) as compared to only 5 tagged parameters (107 features) in the D-Link IoT dataset. Furthermore, it has also been observed that mostly D-Link devices from the D-Link IoT dataset have been incorrectly classified with the combined Glimps and D-Link IoT dataset.

To evaluate the proposed device identification model, the D-Link IoT dataset (26,80,220 labelled instances or packets from eight IoT devices) is randomly split into two groups: 70% of the dataset for training, and the remaining 30% of the dataset for testing. The J48 ML algorithm has been used on two levels for classification. Figure 9a shows the overall classification performance in terms of precision and recall, of the proposed DFP

model using the selected packet-level features, i.e. 24 features, and the combination of packet-level and frame-level features, i.e. 28 features, according to gain-ratio and device-specific features. It can be observed that, on average, the proposed DFP method achieves over 95% precision and 94.5% recall utilizing only packet-level features. Using both levels of features, the proposed DFP model reaches up to 99% precision and 98.9% recall, which is almost 4.4% higher than the recall obtained using packet-level features only.



(a) Performance of the proposed IoT device classification model on the D-Link IoT dataset.



(b) Individual device classification accuracy on the D-Link IoT dataset.

Figure 9. Classification performances of the proposed DFP model.

Figure 9b depicts individual IoT device classification performance on the D-Link IoT dataset. Inspecting individual device classification performance by using only packet-level features, it can be observed that the DCS-903L model devices achieved higher classification accuracy; with D-LinkDayCam1_5d and D-LinkDayCam2_93 (DCS-903L model devices) obtaining average classification precision of over 99%, in contrast to D-LinkCam1_a4 (DCS-936L model device) with only 72.1% accuracy. The proposed DFP model attained over 94.2% of accuracy for all devices by using a combination of packet-level and frame-level features, as compared to using the packet-level feature set only, with a minimum device accuracy of 72.1%. Using the combined feature set, the precision of the D-LinkCam1_a4 device increased a whopping from 22.1% to 94.2%, whilst precision increases by 6.1% for the D-LinkCam2_11 device to reach nearly 99%. Additionally, using frame-level features only (*wlan.ra, wlan.duration, wlan.fcs,* and *wlan.tag.length_4*) as input to the classification model gave classification accuracy of over 92%. Classification performance of the proposed DFP model has also been evaluated using a blind-fold validation method, with 7 IoT devices considered as known devices (2,14,961 instances), and the remaining 1 IoT device considered as unknown (or malicious) device (22,846 instances). The unknown device has been

incorporated into the network without prior knowledge of the network administrator. Over 98% accuracy in distinguishing between known and unknown devices with only packet-level features has been demonstrated. Subsequently, the proposed DFP model has been evaluated in a real deployment environment; with network traffic traces from ten consecutive days utilized for training, and the following one-day traffic traces used for assessing the model classification performance. Overall, 94.46% and 98.89% accuracies were obtained using the packet-level and combined packet-level and frame-level features, respectively.

Table 6 presents a comparative summary of the existing device fingerprinting approaches along with the proposed DFP model. Prior DFP approaches have utilized either packet-level or frame-level features to generate unique fingerprints, whilst none of the DFP models have combined both packet-level and frame-level features as device fingerprints. Additionally, general DFP approaches require a large amount of packets information with a large number of features in most cases. It can be seen that the proposed DFP model provides a high accuracy of 99% by using a combined but limited number of packet-level and frame-level features (a total of 24 packet-level and 4 frame-level features only) from a single packet and frame only.

Table 6. Comparison of the proposed DFP scheme with the existing approaches.

Source	Dataset	Devices	Packets	Frames	Features	Accuracy
[6]	IoT Sentinel	27	12^{Packets}	...	23	$^{\text{RF}}$ 81.5%
[17]	UNSW	28	$n^{\text{Packets/Hour}}$	...	8	$^{\text{RF}}$ 99%
[35]	GTID	23	n^{Packets}	...	IAT	$^{\text{ANN}}$ 95%
[43]	Glimps	100	...	n^{Frames}	Bit patterns	80%
[44]	Sapienza	300	...	1^{Frame}	19	95%
[45]	^{2008}Sigcomm	158	...	50^{Frames}	Transmission Time	95%
[30]	IoT Sentinel	23	1^{Packet}	...	212	$^{\text{PART}}$ 82%
[60]	IoT Sentinel	27	21^{Packets}	...	67	$^{\text{RF}}$ 90.3%
[16]	IoT Sentinel	27	1^{Packet}	...	161	$^{\text{J48}}$ 83.35%
The proposed DFP model	IoT Sentinel	31	1^{Packet}	...	24	$^{\text{J48}}$ 83.9%
	UNSW	21	1^{Packet}	...	24	$^{\text{J48}}$ 97.2%
	D-Link IoT	8	1^{Packet}	...	24	$^{\text{J48}}$ 95%
	D-Link IoT	8	1^{Packet}	1^{Frame}	24 + 4	$^{\text{J48}}$ 99%

The proposed DFP model also provides high accuracy using only packet-level features. Comparison of works on the IoT Sentinel dataset, reference [6] obtains 81.5% accuracy with a 276-dimensional feature vector, whilst references [16, 30] achieve 82% (212 features) and 83.35% (160 features) accuracies using a single TCP/IP information, respectively. Reference [60] demonstrates maximum accuracy of 90%; however, it requires 67 statistical features from consecutive 20–21 packets information, as compared to the proposed DFP model, which requires only 24 packet-level features from a single TCP/IP packet, to give over 83.9% accuracy. Similarly, although reference [17] demonstrates over 99% accuracy on the UNSW dataset, it requires the computation of 8 statistical features using n number of packets information from hourly captured traffic traces. On the other hand, the proposed DFP model only utilizes a single TCP/IP packet information for classifying individual IoT devices to give over 97% accuracy on the UNSW dataset. The proposed DFP model provides the highest classification performances on the D-Link IoT dataset, with 95% and 99% of accuracies using packet-level (24 features) and combined features (28 features), respectively.

5. Conclusion and future work

Relying on user-defined identifiers, such as IP/MAC addresses, has been shown to be prone to security breaches. In this paper, a novel DFP approach based on a set of features from both packet-level and frame-level has been described. Individually, from the packet-level and frame-level, a large number of features space has been explored to identify device-specific features that can be used for device fingerprints. This feature set was then used as input to a supervised ML algorithm i.e. J48, using online datasets, and it has been shown that the proposed approach is able to improve classification performance. Additionally, an experimental testbed has been set up in a laboratory for data collection to evaluate the proposed DFP model performance. The results have shown that this approach achieves higher classification performance, even when devices come from the same manufacturer and of similar types. Using combined packet-level and frame-level features gives improvement in accuracy than using packet-level features only. Whilst frame-level features investigated in this study are specific for WiFi-enabled devices, packet-level features can be extracted from different communication technologies, which uses the widely used IP packets for communication. The outcome of this research is significant as the proposed DFP method can be implemented as an extra security tool, to ensure the integrity of an IoT network against security breaches. However, in this study, a limited number of IoT devices have been used for the experiments. A larger number of IoT devices traffic traces from both packet and frame levels may be needed to mimic a larger IoT network for further investigation. Additionally, different ML classification algorithms may be considered in the evaluation process for evaluating classification performances and anomaly detection with the same features.

Acknowledgment

The authors are profoundly grateful to the Faculty of Integrated Technologies (FIT), Universiti Brunei Darussalam (UBD), for supporting this research work, as well as to UBD for awarding the UBD Graduate Scholarship (UGS) to the first author.

References

[1] Ammar N, Noirie L, Tixeuil S. Network-Protocol-Based IoT Device Identification. In: 2019 Fourth International Conference on Fog and Mobile Edge Computing; Rome, Italy; 2019: 204-209. doi: 10.1109/FMEC.2019.8795318

[2] Lal S, Prathap J. An energy-efficient lightweight security protocol for optimal resource provenance in wireless sensor networks. Turkish Journal of Electrical Engineering & Computer Sciences 2020; 28 (6): 3208–3218. doi: 10.3906/elk-2003-6

[3] Chowdhury RR, Ansary M. A secured mutual authentication protocol for RFID system. International journal of scientific & technology research 2014; 3 (5): 52-56.

[4] Charyyev B, Gunes MH. IoT Traffic Flow Identification using Locality Sensitive Hashes. In: ICC 2020 - 2020 IEEE International Conference on Communications; Dublin, Ireland; 2020. doi: 10.1109/ICC40277.2020.9148743

[5] Alrashdi I, Alqazzaz A, Aloufi E, Alharthi R, Zohdy M et al. Ad-iot: Anomaly detection of iot cyberattacks in smart city using machine learning. In: 2019 IEEE 9th Annual Computing and Communication Workshop and Conference; Las Vegas, NV, USA; 2019:305-310. doi: 10.1109/CCWC.2019.8666450

[6] Miettinen M, Marchal S, Hafeez I, Asokan N, Sadeghi AR et al. IoT SENTINEL: Automated Device-Type Identification for Security Enforcement in IoT. In: 2017 IEEE 37th International Conference on Distributed Computing Systems; Atlanta, GA, USA; 2017: 2177-2184. doi: 10.1109/ICDCS.2017.283

[7] Tong Y, Tian L, Li J. Novel node deployment scheme and reliability quantitative analysis for an IoT-based monitoring system. Turkish Journal of Electrical Engineering & Computer Sciences 2019; 27 (3): 2052-2067. doi: 10.3906/elk-1802-61

[8] Jafari H, Omotere O, Adesina D, Wu HH, Qian L. IoT Devices Fingerprinting Using Deep Learning. In: MILCOM 2018 - 2018 IEEE Military Communications Conference; Los Angeles, CA, USA; 2018. pp. 1-9. doi: 10.1109/MILCOM.2018.8599826

[9] ÇAVDAR T, Ebrahimpour N. Decision-making for small industrial Internet of Things using decision fusion. Turkish Journal of Electrical Engineering & Computer Sciences 2019; 27 (6): 4134-4150. doi: 10.3906/elk-1809-60

[10] Hasan M, Islam MM, Zarif MII, Hashem MMA. Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. Internet of Things 2019; 7: 100059. doi: 10.1016/j.iot.2019.100059

[11] Hamdani SWA, Khan AW, Iltaf N, Bangash JI, Bangash YA et al. Dynamic distributed trust management scheme for the Internet of Things. Turkish Journal of Electrical Engineering & Computer Sciences 2021; 29 (2): 796-815. doi: 10.3906/elk-2003-5

[12] Choi J, Jeoung H, Kim J, Ko Y, Jung W et al. Detecting and identifying faulty IoT devices in smart home with context extraction. In: 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks; Luxembourg, Luxembourg; 2018: 610-621. doi: 10.1109/DSN.2018.00068

[13] Ramnath S, Javali A, Narang B, Mishra P, Routray SK. IoT based localization and tracking. In: 2017 International Conference on IoT and Application; Nagapattinam, India; 2017: 1-4. doi: 10.1109/ICIOTA.2017.8073629

[14] Guo H, Heidemann J. IP-Based IoT Device Detection. In: Proceedings of the 2018 Workshop on IoT Security and Privacy; Budapest, Hungary; 2018: 36-42. doi: 10.1145/3229565.3229572

[15] Mavrogiorgou A, Kiourtis A, Touloupou M, Kyriazis D. Identification of Bluetooth-Enabled IoT Devices Through Syntactic Similarity Techniques. In: 2019 Eleventh International Conference on Ubiquitous and Future Networks; Zagreb, Croatia; 2019: 200-205. doi: 10.1109/ICUFN.2019.8806153

[16] Chowdhury RR, Aneja S, Aneja N, Abas PE. Network Traffic Analysis based IoT Device Identification. In: BDIOT 2020: Proceedings of the 2020 the 4th International Conference on Big Data and Internet of Things; Singapore; 2020: 79–89. doi: 10.1145/3421537.3421545

[17] Sivanathan A, Gharakheili HH, Loi F, Radford A, Wijenayake C et al. Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics. IEEE Transactions on Mobile Computing 2018; 18 (8): 1745-1759. doi: 10.1109/TMC.2018.2866249

[18] Aneja S, Aneja N, Bhargava B, Chowdhury RR. Device fingerprinting using deep convolutional neural networks. International Journal of Communication Networks and Distributed Systems 2022; 28 (2): 171-198. doi: 10.1504/IJCNDS.2022.121197

[19] Qing G, Wang H, Zhang T. Radio frequency fingerprinting identification for Zigbee via lightweight CNN. Physical Communication 2021; 44: 101250. doi: 10.1016/j.phycom.2020.101250

[20] Garg U, Verma P, Moudgil YS, Sharma S. MAC and Logical addressing (A Review Study). Journal of Engineering Research and Applications 2012; 2 (3): 474-480.

[21] Chowdhury RR. Security in cloud computing. International Journal of Computer Applications 2014; 96 (15). doi: 10.5120/16870-6767

[22] Sari RD, Supiyandi APU, Siahaan MM, Ginting RB. A Review of IP and MAC Address Filtering in Wireless Network Security. International Journal of Scientific Research in Science and Technology 2017; 3 (6): 470-473. doi: 10.31227/osf.io/g6emr

[23] Xu Q, Zheng R, Saad W, Han Z. Device Fingerprinting in Wireless Networks: Challenges and Opportunities. IEEE Communications Surveys & Tutorials 2015; 18 (1): 94-104. doi: 10.1109/COMST.2015.2476338

[24] Ali F. IP Spoofing. The Internet Protocol Journal 2007; 10 (4): 2-9.

[25] Gu X, Wu W, Gu X, Ling Z, Yang M et al. Probe Request Based Device Identification Attack and Defense. Sensors 2020; 20 (16): 4620. doi: 10.3390/s20164620

[26] Matte C. Wi-Fi tracking: Fingerprinting attacks and counter-measures. PhD, Université de Lyon, France, 2018.

[27] Kawai H, Ata S, Nakamura N, Oka I. Identification of communication devices from analysis of traffic patterns. In: 2017 13th International Conference on Network and Service Management; Tokyo, Japan; 2017: 1-5. doi: 10.23919/CNSM.2017.8256018

[28] Linux B. cisco-snmp-slap. BlackArch Linux 2013.

[29] Reynolds L. Change mac address with macchanger Linux command. LinuxConfig 2021.

[30] Aksoy A, Gunes MH. Automated IoT Device Identification using Network Traffic. In: ICC 2019 - 2019 IEEE International Conference on Communications; Shanghai, China; 2019. pp. 1-7. doi: 10.1109/ICC.2019.8761559.

[31] Sivanathan A, Sherratt D, Gharakheili HH, Radford A, Wijenayake C et al. Characterizing and classifying IoT traffic in smart cities and campuses. In: 2017 IEEE Conference on Computer Communications Workshops; Atlanta, GA, USA; 2017. pp. 559-564. doi: 10.1109/INFCOMW.2017.8116438

[32] Bezawada B, Bachani M, Peterson J, Shirazi H, Ray I. et al. Behavioral Fingerprinting of IoT Devices. In: Proceedings of the 2018 Workshop on Attacks and Solutions in Hardware Security; Toronto, Canada; 2018. pp. 41–50. doi: 10.1145/3266444.3266452

[33] Yousefnezhad N, Madhikermi M, Främling K. MeDI: Measurement-based Device Identification Framework for Internet of Things. In: 2018 IEEE 16th International Conference on Industrial Informatics; Porto, Portugal; 2018. pp. 95-100. doi: 10.1109/INDIN.2018.8472080

[34] Charyyev B, Gunes MH. Locality-Sensitive IoT Network Traffic Fingerprinting for Device Identification. IEEE Internet of Things Journal 2020; 8 (3): 1272-1281. doi: 10.1109/JIOT.2020.3035087.

[35] Radhakrishnan SV, Uluagac AS, Beyah R. GTID: A Technique for Physical Device and Device Type Fingerprinting. IEEE Transactions on Dependable and Secure Computing 2015; 12 (5): 519-532. doi: 10.1109/TDSC.2014.2369033

[36] Thangavelu V, Divakaran DM, Sairam R, Bhunia SS, Gurusamy M. DEFT: A Distributed IoT Fingerprinting Technique. IEEE Internet of Things Journal 2018; 6 (1): 940-952. doi: 10.1109/JIOT.2018.2865604.

[37] Hui S, Wang H, Xu D, Wu J, Li Y et al. Distinguishing Between Smartphones and IoT Devices via Network Traffic. IEEE Internet of Things Journal 2021; 9 (2): 1182-1196. doi: 10.1109/JIOT.2021.3078879

[38] Köse M, Taşcioğlu S, Telatar Z. RF Fingerprinting of IoT Devices Based on Transient Energy Spectrum. IEEE Access 2019; 7: 18715-18726. doi: 10.1109/ACCESS.2019.2896696.

[39] Soltanieh N, Norouzi Y, Yang Y, Karmakar NC. A Review of Radio Frequency Fingerprinting Techniques. IEEE Journal of Radio Frequency Identification 2020; 4 (3): 222-233. doi: 10.1109/JRFID.2020.2968369

[40] Gu T, Mohapatra P. BF-IoT: Securing the IoT Networks via Fingerprinting-Based Device Authentication. In: 2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems; Chengdu, China; 2018. pp. 254-262. doi: 10.1109/MASS.2018.00047

[41] Chowdhury RR, Aneja S, Aneja N, Abas PE. Packet-level and IEEE 802.11 MAC frame-level network traffic traces data of the D-Link IoT devices. Data in Brief 2021; 37: 107208. doi: 10.1016/j.dib.2021.107208

[42] Yousefnezhad N, Malhi A, Främling K. Automated IoT Device Identification Based on Full Packet Information Using Real-Time Network Traffic. Sensors 2021; 21 (8): 2660. doi: 10.3390/s21082660

[43] Robyns P, Bonné B, Quax P, Lamotte W. Noncooperative 802.11 MAC Layer Fingerprinting and Tracking of Mobile Devices. Security and Communication Networks 2017; 2017. doi: 10.1155/2017/6235484

[44] Dalai AK, Jena SK. WDTF: A Technique for Wireless Device Type Fingerprinting. Wireless Personal Communications: An International Journal 2017; 97 (2): 1911–1928. doi: 10.1007/s11277-017-4652-y

[45] Neumann C, Heen O, Onno S. An Empirical Study of Passive 802.11 Device Fingerprinting. In: 2012 32nd International Conference on Distributed Computing Systems Workshops; Macau, China; 2012. pp. 593-602. doi: 10.1109/ICDCSW.2012.8

[46] Combs G, Ramirez G, Bottom T, Pane C, Boehm HR et al. tshark - Dump and analyze network traffic. Wireshark 2021.

[47] Kary. Understanding Throughput and TCP Windows. PacketBomb 2014.

[48] AskF5. K35612380: Troubleshooting Latency by Capturing Traffic. AsF5 2019.

[49] Desmond LCC, Yuan CC, Pheng TC, Lee RS. Identifying unique devices through wireless fingerprinting. In: Proceedings of the first ACM conference on Wireless network security; Alexandria, VA, USA; 2008. pp. 46–55. doi: 10.1145/1352533.1352542.

[50] Vanhoef M, Matte C, Cunche M, Cardoso LS, Piessens F. Why MAC Address Randomization is not Enough: An Analysis of Wi-Fi Network Discovery Mechanisms. In: Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security; Xi'an, China; 2016. pp. 413–424. doi: 10.1145/2897845.2897883.

[51] Raynel SM, McGregor AJ, Jorgensen MA. Using the IEEE 802.11 frame check sequence as a pseudo random number for packet sampling in wireless networks. In: Proceedings of WiOPT 2009: 7th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks; Seoul, Korea; 2009. pp. 552-557. doi: 10.1109/WIOPT.2009.5291575

[52] Orebaugh A, Ramirez G, Burke J, Pesce L, Wright J et al. Chapter 6 - Wireless Sniffing with Wireshark. In: Wireshark & Ethereal Network Protocol Analyzer Toolkit, Syngress, 2006; 267-370.

[53] Guo F, Chiueh T. Sequence Number-Based MAC Address Spoof Detection. In: International Workshop on Recent Advances in Intrusion Detection 2005; 309–329. doi: 10.1007/11663812_16

[54] Quinlan JR. C4.5: programs for machine learning. In: Morgan Kaufmann Publishers Inc; San Francisco, CA, United States; 1993.

[55] Witten IH, Frank E, Hall MA. Data Mining, Fourth Edition: Practical Machine Learning Tools and Techniques. In: Morgan Kaufmann Publishers Inc; San Francisco, CA, United States; 2016.

[56] Aljawarneh S, Yassein MB, Aljundi M. An enhanced J48 classification algorithm for the anomaly intrusion detection systems. Cluster Computing 2017; 22 (5): 10549–10565. doi: 10.1007/s10586-017-1109-8.

[57] Wu X, Kumar V, Quinlan JR, Ghosh J, Yang Q et al. Top 10 algorithms in data mining. Knowledge and Information Systems 2008; 14 (1): 1–37. doi: 10.1007/s10115-007-0114-2.

[58] Kurniabudi, Harris A, Mintaria AE, Darmawijoyo, Stiawan D et al. Improving the Anomaly Detection by Combining PSO Search Methods and J48 Algorithm. In: 2020 7th International Conference on Electrical Engineering, Computer Sciences and Informatics; Yogyakarta, Indonesia 2020; 119-126. doi: 10.23919/EECSI50503.2020.9251872

[59] Ngo T. Data mining: practical machine learning tools and technique, third edition by Ian H. Witten, Eibe Frank, Mark A. Hell. ACM SIGSOFT Software Engineering Notes 2011; 36 (5): 51–52. doi: 10.1145/2020976.2021004

[60] Hamad SA, Zhang WE, Sheng QZ, Nepal S. IoT Device Identification via Network-Flow Based Fingerprinting and Learning. In: 2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering; Rotorua, New Zealand; 2019: 103–111. doi: 10.1109/TrustCom/BigDataSE.2019.00023