Network-Protocol-Based IoT Device Identification

Nesrine Ammar
Nokia Bell Labs
Nokia Paris-Saclay
Nozay, France
nesrine.ammar@nokia-bell-labs.com

Ludovic Noirie
Nokia Bell Labs
Nokia Paris-Saclay
Nozay, France
ludovic.noirie@nokia-bell-labs.com

Sébastien Tixeuil
Sorbonne Université
CNRS, LIP6
Paris, France
sebastien.tixeuil@lip6.fr

***Abstract*—To empower end-users in the management of their IoT devices and related services, a natural solution is to design and implement a digital assistant whose role is to facilitate use of IoT devices, *e.g.* by recommending available services for the given set of existing IoT devices. This digital assistant must be able to identify the core capabilities of the IoT devices that are connected to home networks. In turn, this requires to identify the nature of the IoT devices connected to these home networks (*e.g.*, category of the device, but also manufacturer and exact model of the device). In this article, we address this issue of IoT device identification. We propose a solution based on several existing network protocols. The key idea of our solution is to analyze the packets sent by the device to extract relevant information for device identification purpose. We show that our solution is effective by uniquely identifying 31 devices among 33 of the tested devices: each of these devices is identified by a unique feature vector using the Bag Of Words representation.**

***Index Terms*—Internet of things, recommendation systems, connected device identification, network protocols**

I. Introduction

With the proliferation of smart objects, more and more people buy IP devices and home appliances to benefit from new services enabling them to be informed about and control their houses anytime and anywhere [1], [2]. With the rise of IoT devices, an IoT device management system to help end users to benefit from several functionalities such as service recommendation, devices management, devices security, quality of service and so on is primordial. In order to be able to propose and implement such systems, one of the important missing mechanisms is home devices capabilities identification. By giving an IoT system more information about the targeted devices such as its type, manufacturer and model in best case, they are able to perform smarter and deliver more relevant information to the user about their devices to ensure easier management and reinforce home security.

Noirie et al. [3] proposed a recommendation system to help end users improve their IoT experience. The aforementioned recommendation system is integrated as part of the *Majord'Home* solution [4]. The *Majord'Home* system is a software-based approach allowing users to better control and manage smart devices in their homes. This system permits to visualize, control and manage home connected devices based on software-defined LAN (SD-LAN) techniques [5]. An SD-LAN is a network slice that isolates devices from other devices, easing the discovery of devices in a limited area.

An important aspect in the aforementioned recommendation system is the required knowledge about the connected devices to be proposed for service recommendation to the users. The system must know the available connected devices and their capabilities (their physical interfaces) in order to be able to propose them for the end user to deploy a specific service. This is why the device identification is required. Network connected device identification is a challenging task in heterogeneous environment such as smart homes. In this work, we address the device identification problem within a home network by analyzing device information that are modeled by a feature vector using the Bag Of Words (BoW) technique [6]. This technique shows a good results among the set of real-world devices used to evaluate our solution. Nevertheless, this approach based on devices information extracted from network protocols from the upper layers of the protocol stack allows our system to distinguish between the different types of connected devices. This solution allows us to identify heterogeneous devices coming from several vendors, models and types.

Goals and contributions:

In this paper, we tackle this device identification issue by presenting a new assistant designed and implemented within the *Majord'Home* platform. Our device identification assistant is capable of identifying the type of devices connected to a network while minimizing user involvement and enhancing her IoT experience. In more details, our assistant extracts relevant information from several layers of the network protocol stack to identify a given device that the users have deployed in their smart environment.

The contributions of this paper are the following:

- We first present the different techniques and information used for device identification that we identified as relevant to address our problem (Section III);
- We introduce our methodology for device identification and the design of our device identification assistant (Section IV);
- We evaluate the coverage rate of our solution using a data set of various real-world devices (Section V).

II. Related work

The process of gathering device information to form a device fingerprint and using them to identify individual devices

This work was partly carried out at the LINCS (Laboratory of Information, Networking and Communication Science, http://www.lincs.fr/).

is a promising solution to address several research issues, such as improving user experience and help them better manage their connected devices in smart home environment or reducing the communications of vulnerable devices by identifying the types of devices being connected to an IoT network [7]. The basic idea is to passively or actively collect information about devices from traffic patterns based on a set of predefined features from the target devices.

Previous network communication fingerprinting works focus on identifying hardware and driver specifications for each device. Device identification based on characteristics of communication features such as signals was discussed by Talbot et al. [8], which demonstrates efforts since 1960s for addressing this research problem. Cache [9] used 802.11 frames duration fields to identify Wifi drivers. Franklin et al. [10] were able to classify 17 Wifi drivers with a high accuracy from 77% to 96% using a Bayesian classification approach with the frame inter-arrival time from devices traffic as the input features of the classification algorithm.

Furthermore, the idea of using network characteristics to identify different nodes connected to the network has been applied in several context. Network features such as packet length, packet destination, MAC protocol fields and features found at lower layers of the protocol stack were used to address several problems such as traffic classification, fingerprint specific users and fingerprint devices. Hardware specific features such as clock skew were used to identify a unique network interface card [11], [12].

However, these solutions are based on hardware and drivers specifications that are not sufficient for our purpose, because the same hardware components and drivers may be deployed in large number of heterogeneous devices. Our objective is to identify the exact device type. Recent work [7] tries to discriminate smart device types using machine learning. A fingerprint is presented by n packets and 23 features such as packet length, port number and protocol used by the packet presented as binary features. However, the performance evaluation shows a precision over 0.95 for 17 devices and a precision around 0.5 for 10 devices. In our work, we parse information shared by the network protocols to identify devices as this payload contain key textual description of the device itself. This device description is presented by a bag of words model. Our results shows a good identification of 31 devices among 33.

III. Features extracted for device identification

According to our state-of-the-art, several features could be used for device identification nature based on existing approaches. In our work we classify our features into three categories, whether they are extracted passively or actively.

- The first set of features is extracted from exchanged packets used by service discovery protocols. To advertise services provided by a device connected to a home network, some devices advertise information about their capabilities, location, name and the description of their services. They use broadcast messages containing information about themselves. For this work, we selected mDNS (Bonjour) and SSDP (UPnP) as they are the most used protocols by connected devices. However, more features extracted from others discovery protocols such as WS-Discovery could be added in future as our solution is modular, see section IV-B.
- The second set of features is based on the DHCP fingerprinting, which allows us to know the device type, its Operating System and vendor from messages exchanged between the device and the DHCP server. This can be completed with information about the MAC address.
- Another type of features is extracted from the user-agent in the HTTP header.

A. mDNS with Bonjour

Bonjour supports services discovering and advertising using multicast DNS instead of unicast DNS messages. Bonjour is an Apple implementation of a suite of zero-configuration networking protocols. Zeroconf is designed to make network configuration easier for users. Bonjour is based on DNS Service Discovery (DNS-SD, [13]) protocol over mDNS. The mDNS provides the ability to perform DNS operation on a local network in the absence of a conventional unicast DNS server. DNS-SD is widely used in devices discovery and resolution of services and names on the local link. In case an mDNS client wants to discover an endpoint by resolving its host-name to an IP address, it has to send an IP multicast query message over the network. Once the host receives the message, it replies via multicast message that contains its IP address. All nodes in the network receiving that multicast message update their mDNS caches.

Bonjour is mainly used to discover services, not devices. Each service is provided by a specific device. Thus, by discovering advertised services, we are able to identify the service provisioning device. DNS-SD is a standardized protocol for describing and resolving services by using DNS RRs. DNS-SD define how a client can discover service instances using a service type as selection criteria. Service are described using different DNS records such as SRV, PTR, TXT, A, AAAA. The SRV and TXT have the same structured name $\langle name \rangle . \langle type \rangle . \langle domain \rangle$. The $\langle name \rangle$ part is the unique identifier of the service instance in the same $\langle domain \rangle$. Then, the service type is formed by concatenating the application protocol and transport protocol used for accessing the service. Lastly, the domain defines the scope of the service instance. In addition to service instance name, the SRV RR contains the port number and the host-name of the service provider. The host name is resolved to and IP address thanks to A records.

B. SSDP with UPnP

Universal Plug and Play (UPnP) [14] is an architecture for peer to peer network connectivity from different types of devices such as PCs, wireless devices, and applications. It simplifies the integration of new devices within a home network by allowing auto-configuration rather than asking end users to manually configure their new connected devices and

with whom they cooperate. UPnP is used by most of the multimedia devices.

When a device joins the network, it obtains an IP address from the DHCP server or assigns an IP address from a specific range of addresses if the DHCP server is not available. Then it starts to discover other UPnP devices on the same network and what services these devices are offering. UPnP uses SSDP protocol for device discovery, which uses two types of messages, SEARCH and NOTIFY, for device search and advertisement. SEARCH messages are used by the control points to know about the devices connected to the same LAN. In response, the UPnP-enabled devices send NOTIFY messages to advertise about their capabilities. They contain an URL pointing to the XML description of each device. This description indicates the device vendor, type, model, OS and the different services provided by the device. Such information allows us to identify the connected device type.

C. DHCP fingerprinting and MAC addresses

DHCP fingerprinting is used to identify the device type, name, vendor and OS. Example of devices that can be identified by DHCP fingerprinting: mobiles devices, tablets, printers, desktops. A device interacts with a DHCP server to obtain an IP address. During the DHCP protocol exchange, there are options for the DHCP server to query information on the host name, the type of device, the manufacturer name and the OS of the client device. This is defined in [15] and is called as DHCP Fingerprinting.

This can be also completed with the identification of the manufacturer with the IEEE MAC address [16]. But this last information is more or less relevant as it is related to the network interface whose manufacturer may be different from the manufacturer of the whole device. Furthermore, it can be changed randomly to avoid device tracking [17].

D. User-agent in HTTP requests

Another approach consists of extracting the user-agent from the header of a HTTP request sent by the device, which contains information about the software agent that sent this request (e.g., web browser), the OS used by the device, etc [18]. This technique is limited to particular types of devices, namely mobile devices and computers.

Some discovery protocols are based on the user agent of the HTTP packets that are considered as static approaches based on device profile. The first approach is Composite Capability/Preferences Profile CC/PP defined by the World Wide Web Consortium W3C. It uses specific vocabularies based on XML for device capabilities description about the device software, hardware, network and browser. The second approach is User Agent Profile UAProf that is based on CC/PP. A mobile device sends a header within the HTTP request that contains the URL to its UAProf. The UAProf is stored in a profile repository maintained by a mobile device manufacturer. The URL is found in the HTTP header x-wap-profile. These approaches are mainly used in content delivery applications to optimize users' mobile experience [19].

IV. IoT DEVICE IDENTIFICATION

We discussed in the above section about information shared by several protocols that could be relevant to address the device identification issue. In this section, we first introduce our methodology to identify the types of IoT devices, with the list of features we used and the usage of the Bag of Words technique. Then we describe the architecture of our device identification assistant applying the aforementioned technique to determine the nature of new connected devices.

A. Device identification methodology

Our features are extracted passively and actively, it depends on the protocol used. For example, features based on mDNS and SSDP could be extracted actively as passively whereas the user agent is extracted passively. The extracted features are the following:

- Manufacturer name from MAC address;
- Device name from DHCP information;
- Manufacturer name, model, friendly name and type from XML description shared within UPnP messages during the discovery process;
- Device local name, services names and types offered by the device from mDNS records;
- Device OS, model and in some cases type from the user-agent of the HTTP header, mainly for mobile devices.

Besides, each device is initially represented by a set of textual information. The textual information could be presented on the form of a word or a set of words. Each device is represented by n extracted words from protocols packets. The number of words n is different from a device to another. Then, these textual information are modeled using the Bag of Words (BoW) representation for device identification. The BoW technique is mainly used for document classification where document are represented by a set of unordered words and is widely used for image classification too [6].

Thus, our textual information is modeled as a binary data using the BoW technique. First, we make a list L of all the unique words extracted of all devices. The next step consists to turn each device description into a feature vector using L of size m. A device is represented by a feature vector of m words (features), set to 1 if the word is present in the device description and to 0 if not.

These vectors are used to build a database in order to to determine the nature of new connected device. Furthermore, our final data is a $p \times m$ matrix M with each column representing a word and each row representing a device. The data could be labeled by the device model for a fine-granularity device identification. Moreover, the labels could be the device type, e.g. camera, tablet, scale, etc. In this case, the aggregation of the feature vector of devices of the same type is necessary. Besides, a set of feature vectors of several devices models is aggregated to a unique vector representing a device type.

The next step focuses on decreasing the size of our vector to make sure to keep only relevant words. To do so, simple cleaning techniques are adopted: ignore symbols, digits and remove common words between different devices.

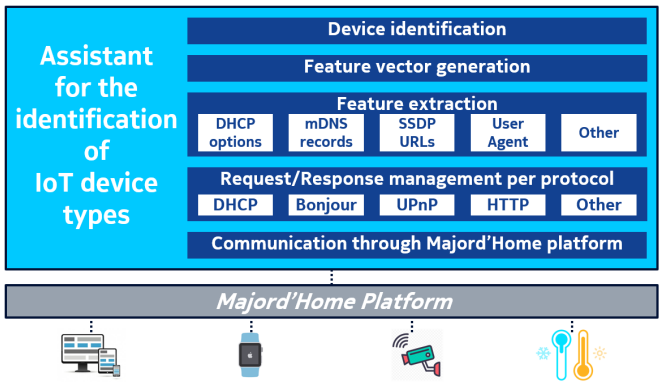

Fig. 1. Architecture of the IoT device identification assistant

After the feature extraction and feature vectors generation, the following step consists in developing our device identification method to identify new connected devices based on prior devices and be able to discriminate between them. According to our technique, the more devices we have, the better it is. The textual description of a new devices is represented by a feature vector used in our data taking into account only words in the data vector. Then, a traditional method is implemented to compare the obtained feature vector with all the inputs in the database and select the most similar one. The inputs in our database are considered as a set of rules. This method allows us to reduce the human involvement and make the device identification task autonomous.

B. Device identification assistant

Our methodology for IoT device identification is generic and could be deployed in any home network platform. For our implementation, we propose a software assistant on top of the *Majord'Home* platform, which is based on Software-Defined-Networks techniques and aims at isolating devices from each other to enforce network security [9].

This software assistant is made of several modules as shown in Fig. 1:

- The *Communication* module manages the communication with the *Majord'Home* and with the IoT devices. The *Majord'Home* informs it when a new device is connected to the home network. This module creates a SD-LAN between itself and the device to process the identification of this device. This is the only module that must be modified if another home network platform is used.
- The *Request/Response management* module is responsible to detect if the approaches listed in section III can be used to identify the device. For the DHCP + MAC address approach, it receives the information from the *Majord'Home* platform that handles DHCP operations. For the SDP-based approaches (mDNS or SSDP), it sends a request on the corresponding multicast addresses to newly connected device. The device being in the same SD-LAN, if it can to listen to this SDP-specific request, then it sends back a response advertising its capabilities. For the User Agent approach, it listens to HTTP requests from the device. For each approach, the resulting information is forwarded to the upper module;
- The *Feature extraction* module is responsible to parse the resulting information corresponding to the newly connected devices and to extract the relevant information to form the fingerprint for the device identification.
- The module *Feature vector generation* is used to represent our data as a BoW.
- Finally, the module *Device identification* is used to identify new connected devices based on prior devices description.

Our solution is modular: we can add any other features that we could think to be pertinent in the future, as all modules of feature extraction work in parallel.

V. Evaluation and Results

In this section, we evaluate our solution for IoT device identification with real devices in our laboratory as well as with publicly available IoT device traffic traces.

A. Tested devices

To evaluate the IoT device identification approaches and their effectiveness, we implemented the different modules of IV-B. We test them in our IoT laboratory that uses the *Majord'Home* platform and that is representative of a typical smart home environment with 12 connected devices [20].

To evaluate more devices, we also used the data set shared by Miettinen et al. [7]. It represents collected traffic measurements about the packets sent by the device to the gateway during the setup process. When a new device is identified thanks to its MAC address, n packets are recorded during this process. The end of this process is identified by a decrease in the rate of packets sent by the devices.

B. Analysis of IoT device identification results

To evaluate our device identification solution, we compute the number of devices uniquely identified among the tested device set of each approach with the proportion of devices identified by each of them, and the overall coverage by the proportion of devices identified by none of them. The results are presented in table I and II. The results in table I concerns the IoT devices captures published by Miettinen et al. [7], whereas the table II presents the results of the IoT devices in our laboratory [20].

A first remark is that in some cases the information from different approaches is redundant. We note twofold benefits of having duplicated information. First, the results obtained are consolidated by verifying the correlation between these information. If the information does not correspond to each other, then the device may be malicious and not be what it pretends to be. Thus, the second benefit is that these information could be used to verify the legitimacy of devices connected to the home network by measuring the correlation between the redundant information in order to identify malicious devices.

TABLE I
ANALYSIS OF THE 21 DEVICES FROM THE DATASET OF MIETTINEN ET AL. [7] USING OUR PROPOSED DIFFERENT APPROACHES

Type	Nb	Manufacturer	mDNS	SSDP	DHCP	User-agent	None
Camera	6	D-Link, Edimax, Ednet	2	4	2	0	2/6
Light	2	Philips, Lightify	1	1	1	0	0/2
Home appliance	2	iKettle2, Espressif	0	0	2	0	0/2
Sensor	3	D-Link	3	0	2	0	0/3
Mobile	3	Apple, Sumsung	0	0	0	3	0/3
Switch	3	D-Link, Philips, WeMo	1	3	1	0	0/3
Alarm	1	D-Link	1	0	1	0	0/1
Hub	1	D-Link	1	1	1	0	0/1
Total	**21**		**9**	**9**	**10**	**3**	**2/21**

TABLE II
ANALYSIS OF THE 12 DEVICES FROM OUR LAB [20] USING OUR PROPOSED DIFFERENT APPROACHES

Type	Nb	Manufacturer	mDNS	SSDP	DHCP	User-agent	None
Camera	4	D-Link, Panasonic, TRENDnet	1	4	2	0	0/4
Light	1	Philips	0	0	1	0	0/1
Tablet	2	Asus, Pixel C	0	0	2	2	0/2
PC	3	HP	0	1	3	3	0/3
Speaker	1	Chromecast	0	0	1	0	0/1
TV	1	Chromecast	0	1	1	0	0/1
Total	**12**		**1**	**6**	**10**	**5**	**0/12**

TABLE III
FEATURES FOR THE 12 DEVICES FROM OUR LAB [20]

Device	Feature vector
Camera 1	TRENDnet, Inc, TV, IP422W, Pan, Tilt, Wireless, Network, Camera
Camera 2	D, Link, International, DCS, 5020L, Wireless, Pan, Tilt, Internet, Camera
Camera 3	D, Link, International, 942LB, DLINK, Internet, Camera, DCS, 942LB1
Camera 4	Panasonic, Communications, Co, Ltd, Network, Camera, BL, C230
Light	Philips, Lighting, BV, hue
Tablet 1	AzureWave, Technology, Inc, Android, Mozilla, Gecko, Firefox,Tablet, rv
Tablet 2	ASUSTek, COMPUTER, INC, Android, Mozilla, Gecko, Firefox, Tablet, rv
PC 1	Hewlett, Packard, FRVILN0H478020, Mozilla, Windows, NT, Win64, x64, rv, Gecko, Firefox
PC 2	Hewlett, Packard, DESKTOP, 9P7SHGI, Mozilla, Windows, NT, WOW64, rv, Gecko, Firefox, Media, Player, Sharing, ...
PC 3	Hewlett, Packard, DESKTOP, C0T5OI9, Mozilla, Windows, NT, Win64, x64, rv, Gecko, Firefox
Speaker	Google, Inc, Audio, Chromecast
TV	Liteon, Technology, Inc, Google, TV, Corporation, Home, Dongle, Chromecast, Eureka

As seen in Table I and II, the DHCP protocol shows the highest coverage by extracted information of 10 devices among 21 of the first set of devices and 10 devices among 12 of second set of devices. Examples of DHCP names used by the vendors: “Philips-hue” for Philips LED in our lab, “Chromecast-Audio” for a loud speaker. These information is considered as relevant features to identify connected devices in the home network. The UPnP protocol covered 9 and 6 among 21 and 12 of the sets of devices used, respectively. For example, we were able to extract the following information about the Panasonic camera based on several fields of the UPnP description: “Network Camera BL-C230, Panasonic, Panasonic Network Camera, Network Camera, BL-C230”. Nevertheless, for the mDNS protocol, we extracted device information from mDNS record of 9 devices among 21 from the devices captures. For the second set of devices the mDNS approach shows a very low coverage because only one device responded to our multi-cast request in the laboratory. The results based on the user-agent shows that this approach succeed to identify PCs and tablets. The final results show an identification of 30 devices among 33. Besides, only 2 devices were not identified. Both are Edimax cameras that do not use any of the discovery protocols and did not share any information except for MAC addresses giving the manufacturer name. The results show that we get devices description using different protocols that covers the lack of one of them. Moreover, extracting information from several protocols make the devices description richer.

All this information was used to generate the feature vector for the characterization of each device following the technique described in the section IV-A. Removing the 2 Edimax cameras that did not give enough information, we obtained a 31×98 matrix M where each row represents a device and each column is set to 1 if the word is present in the device description and to 0 if not. The device identification was evaluated through a similarity verification process. Our results show that each device is identified by a unique feature vector. The table III shows the keywords (features) of the set of

our laboratory devices (only features set to 1 in the matrix M are presented). The results are similar for both of the devices sets. We remark that each device is identified by a set of unique features. This shows that, as expected, our method is able to discriminate between device types based on network protocols shared information.

VI. Conclusion and future work

In this paper, we have presented a software assistant designed and implemented above the *Majord'Home* architecture for identifying connected devices in a home network able to discriminate between home connected devices. To do so, our methodology consists in extracting device textual description from packets shared by network protocols. We consider that messages shared between devices and the gateway are rich enough to discriminate between home devices. Nevertheless, extracted information allow us to generate a unique feature vector for each device connected to the home network using BoW representation to build our database. Furthermore, this database is used to identify new connected devices based on prior identified devices by checking the similarity between the vectors. In this case, the identification of the new connected device happens when some packets are exchanged between this latter and the home gateway. Thus, the extracted information are based on the first set of packets sent by the device. Our solution shows a good coverage rate among the set of tested devices. Nevertheless, the results show that each device is identified by a unique feature vector.

In order to make our solution autonomous, we plan to investigate machine learning algorithms for IoT device identification based on information shared by network protocols. For this objective, we will use the collected information in this article as initial features to construct our training data set. In order to improve the accuracy of our method by covering devices that do not share any information, we intend to investigate additional features coming from the network traffic patterns of the connected devices.

Acknowledgment

The authors thanks the teams in Nokia Bell Labs Paris-Saclay and Antwerp for their support on the *Majord'Home* platform, more particularly Nicolas Le Sauze, Dinh Thai Bui, Werner Liekens, Michel le Pallec, Pierre Peloso and Frederik Vandeputte.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787 – 2805, 2010. [Online]. Available: https://doi.org/10.1016/j.comnet.2010.05.010

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347–2376, Fourth quarter 2015. [Online]. Available: https://doi.org/10.1109/COMST.2015.2444095

[3] L. Noirie, M. L. Pallec, and N. Ammar, "Towards automated IoT service recommendation," in *2017 20^{th} Conference on Innovations in Clouds, Internet and Networks (ICIN)*, Mar. 2017, pp. 103–106. [Online]. Available: https://doi.org/10.1109/ICIN.2017.7899397

[4] M. Boussard, D. T. Bui, R. Douville, N. L. Sauze, L. Noirie, P. Peloso, R. Varloot, and M. Vigoureux, "The Majord'Home: a SDN approach to let ISPs manage and extend their customers' home networks," in *10^{th} International Conference on Network and Service Management (CNSM) and Workshop*, Nov. 2014, pp. 430–433. [Online]. Available: https://doi.org/10.1109/CNSM.2014.7014207

[5] M. Boussard, D. T. Bui, L. Ciavaglia, R. Douville, M. L. Pallec, N. L. Sauze, L. Noirie, S. Papillon, P. Peloso, and F. Santoro, "Software-Defined LANs for interconnected smart environment," in *2015 27^{th} International Teletraffic Congress*, Sep. 2015, pp. 219–227. [Online]. Available: https://doi.org/10.1109/ITC.2015.33

[6] H. Nguyen-Duc, T. Do-Hong, T. Le-Tien, and C. Bui-Thu, "A survey of classification accuracy using multifeatures and multi-kernels," in *2013 International Conference on Advanced Technologies for Communications (ATC 2013)*, Oct 2013, pp. 661–666.

[7] M. Miettinen, S. Marchal, I. Hafeez, T. Frassetto, N. Asokan, A. R. Sadeghi, and S. Tarkoma, "IoT Sentinel demo: Automated device-type identification for security enforcement in IoT," in *2017 IEEE 37^{th} International Conference on Distributed Computing Systems (ICDCS)*, Jun. 2017, pp. 2511–2514. [Online]. Available: https://doi.org/10.1109/ICDCS.2017.284

[8] K. I Talbot, P. R Duley, and M. H Hyatt, "Specific emitter identification and verification," *Technology Review Journal*, vol. 11, pp. 113–133, Spring-Summer 2003.

[9] J. Cache, "Fingerprinting 802.11 Implementations via Statistical Analysis of the Duration Field," *uninformed.org*, vol. 5, Sep. 2006. [Online]. Available: http://www.uninformed.org/?v=5&a=1&t=sumry

[10] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J. Van Randwyk, and D. Sicker, "Passive data link layer 802.11 wireless device driver fingerprinting," in *Proceedings of the 15^{th} Conference on USENIX Security Symposium - Volume 15*, ser. USENIX-SS'06. Berkeley, CA, USA: USENIX Association, 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1267336.1267348

[11] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, pp. 93–108, Apr. 2005. [Online]. Available: https://doi.org/10.1109/TDSC.2005.26

[12] C. Arackaparambil, S. Bratus, A. Shubina, and D. Kotz, "On the reliability of wireless fingerprinting using clock skews," in *Proceedings of the Third ACM Conference on Wireless Network Security*, ser. WiSec '10. New York, NY, USA: ACM, 2010, pp. 169–174. [Online]. Available: https://doi.org/10.1145/1741866.1741894

[13] S. Cheshire and M. Krochmal, "DNS-Based Service Discovery," Internet Requests for Comments, RFC Editor, RFC 6763, Feb. 2013. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6763.txt

[14] UPnP forum, "UPnP Device Architecture 2.0," UPnP forum, Specification, Feb. 2015. [Online]. Available: http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v2.0.pdf

[15] S. Alexander and R. Droms, "DHCP Options and BOOTP Vendor Extensions," Internet Requests for Comments, RFC Editor, RFC 2132, Mar. 1997. [Online]. Available: http://www.rfc-editor.org/rfc/rfc2132.txt

[16] IEEE.org, "Public listing for IEEE standards registration authority," 2018. [Online]. Available: https://regauth.standards.ieee.org/standards-ra-web/pub/view.html#registries

[17] A. Mamiit, "Apple implements random MAC address on iOS 8. Goodbye, marketers," Jun. 2014, Tech Times. [Online]. Available: http://www.techtimes.com/articles/8233/20140612/apple-implements-random-mac-address-on-ios-8-goodbye-marketers.htm

[18] T. Berners-Lee, R. T. Fielding, and H. F. Nielsen, "Hypertext Transfer Protocol – HTTP/1.0," Internet Requests for Comments, RFC Editor, RFC 1945, May 1996. [Online]. Available: http://www.rfc-editor.org/rfc/rfc1945.txt

[19] D. Zhang, "Web content adaptation for mobile handheld devices," *Commun. ACM*, vol. 50, no. 2, pp. 75–79, Feb. 2007. [Online]. Available: https://doi.org/10.1145/1216016.1216024

[20] M. L. Pallec, L. Noirie, P. Peloso, D. T. Bui, and N. L. Sauze, "Digital assistance for the automated discovery and deployment of IoT services," in *21^{st} Conference on Innovations in Clouds, Internet and Networks (ICIN)*, Feb. 2018.