Saving Private Addresses: An Analysis of Privacy Issues in the Bluetooth-Low-Energy Advertising Mechanism

Guillaume Celosia, Mathieu Cunche

Saving Private Addresses: An Analysis of Privacy Issues in the Bluetooth-Low-Energy Advertising Mechanism

Guillaume Celosia
Univ Lyon, INSA Lyon, Inria, CITI
F-69621 Villeurbanne, France
guillaume.celosia@insa-lyon.fr

Mathieu Cunche
Univ Lyon, INSA Lyon, Inria, CITI
F-69621 Villeurbanne, France
mathieu.cunche@insa-lyon.fr

ABSTRACT

The Bluetooth Low Energy (BLE) protocol is being included in a growing number of connected objects such as fitness trackers and headphones. As part of the service discovery mechanism of BLE, devices announce themselves by broadcasting radio signals called advertisement packets that can be collected with off-the-shelf hardware and software. To avoid the risk of tracking based on those messages, BLE features an address randomization mechanism that substitutes the device address with random temporary pseudonyms, called Private addresses.

In this paper, we analyze the privacy issues associated with the advertising mechanism of BLE, leveraging a large dataset of advertisement packets collected in the wild. First, we identified that some implementations fail at following the BLE specifications on the maximum lifetime and the uniform distribution of random identifiers. Furthermore, we found that the payload of the advertisement packet can hamper the randomization mechanism by exposing counters and static identifiers. In particular, we discovered that advertising data of *Apple* and *Microsoft* proximity protocols can be used to defeat the address randomization scheme. Finally, we discuss how some elements of advertising data can be leveraged to identify the type of device, exposing the owner to inventory attacks.

CCS CONCEPTS

• **Networks → Network privacy and anonymity**; *Mobile and wireless security*; Wireless personal area networks.

KEYWORDS

Bluetooth Low Energy; Privacy; Tracking; Address randomization.

ACM Reference Format:
Guillaume Celosia and Mathieu Cunche. 2019. Saving Private Addresses: An Analysis of Privacy Issues in the Bluetooth-Low-Energy Advertising Mechanism. In *16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous), November 12–14, 2019, Houston, TX, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3360774.3360777

1 INTRODUCTION

Bluetooth Low Energy (BLE) is a 2.4 GHz ISM radio communication standard developed by the Bluetooth Special Interest Group (SIG). Compared to Bluetooth, BLE provides a reduced power consumption while maintaining a similar communication range that makes it suitable for devices with low energy resources such as fitness trackers, headphones, etc. According to the Bluetooth SIG, more than 2 billion devices supporting BLE have been shipped in 2018 [25].

While wireless technologies such as Bluetooth/BLE or Wi-Fi bring hands-on facilities, they also have the potential to expose users to physical tracking because of the identifiers found within the radio signals emitted by their devices [8, 10, 27]. To protect users against this threat, the LE Privacy [24, Vol 3, Part C, sec. 10.7] feature defines the use of temporary link layer identifiers that periodically change for a random value. However, it has been shown [7, 8] that some information can still leak from BLE devices.

As other wireless technologies, BLE features a discovery mechanism called *advertisement* [24, Vol 3, Part C, sec. 11]. To enable this mechanism, devices periodically broadcast advertisement packets that are populated with a variety of information announcing their characteristics and available services. Those packets are transmitted in clear, opening new possibilities for passively tracking users in the physical world [11, 20].

In this paper, we present an analysis of the privacy provisions provided by the LE Privacy feature as well as its limitations. Our study is based on a dataset of real-world observations of BLE advertisement traffic containing about 8 million packets associated to more than 53500 different device addresses collected over a period of 5 months.

Our contributions are threefold:

- We analyze the deployment of the address randomization mechanism in the wild. In particular, we found that the LE Privacy feature is largely adopted and that most implementations follow the specifications. However, we also found that a significant fraction of devices are not exhibiting those properties, exposing their users to tracking;
- We show that the content of advertisement packets can undermine the protections provided by the LE Privacy feature. Indeed, we found that some devices include unique identifiers and counters that are not reset upon an address change. We also identified fields containing temporary identifiers, but for which the renewal is not done at the same time as the random address change;
- We identify that advertisement packets include data that can reveal the manufacturer, the model and the type of the device, exposing the user to inventory attacks and inference of sensitive attributes. For instance, some medical devices are broadcasting their types (hearing aid, glucometer, insulin pen, etc.) trivially exposing a medical condition of the owner.

The paper is organized as follows. First, a BLE background is provided in Section 2. Then, Section 3 defines our experimental methodology. We present observations on our dataset in Section 4. Section 5 gives a detailed inspection of the BLE device address randomization mechanism and we show how it can be defeated in Section 6. Section 7 shows the additional information leaked by the

advertising data. Finally, we highlight the related work in Section 8 and Section 9 concludes the paper.

2 BACKGROUND

2.1 BLE protocol

BLE has been introduced in 2010 as part of the Bluetooth Core Specification version 4.0 [22, Vol 6]. In this protocol, 3 channels are dedicated to the discovery mechanism called *advertisement* while the 37 remaining channels are used for data exchanges between two connected devices.

A BLE device can endorse two main roles, Peripheral and Central, following a client-server model. A Peripheral device periodically broadcasts advertisement packets and, if connectable, accepts incoming connection requests from Central. A Central device listens to advertisement packets and, when applicable, initiates a connection with a Peripheral. Note that, a BLE device cannot endorse both roles at the same time.

At the physical layer [24, Vol 6, Part B, sec. 2.1], BLE packets are split into 4 fields: the *Preamble*, the *Access Address*, the *Protocol Data Unit* (PDU) and the *Cyclic Redundancy Check* (CRC). In the case of advertisement packets, the *Access Address* shall be set to `0x8e89bed6` [24, Vol 6, Part B, sec. 2.1.2]. The PDU itself is divided into two sections: a 2-byte header and a variable size payload. The header describes the type of the PDU and the class of the BLE device address (Public or random). The payload part contains the device address followed by the advertising data.

2.2 Device addressing

During communications, devices are identified by a Bluetooth device address (BD_ADDR), a 48-bit identifier found within the Advertising Address (`AdvA`) field of the advertising payload. As part of the LE Privacy feature, BLE supports 3 types of random addresses in addition to the globally unique MAC address (Public):

- Public device address: the address uniquely allocated to the device by the manufacturer in accordance to the IEEE specifications of MAC addresses [1, sec. 8.2];
- Random Static device address: a randomly generated device address that can be renewed after each power cycle and that shall not change during the use of the device;
- Random Non-resolvable device address: a randomly generated address that can be renewed at any time;
- Random Resolvable device address: an address composed of a 22-bit random number *prand* and a 24-bit hash produced by the hashing of *prand* with a 128-bit secret Identity Resolution Key (IRK). This IRK is shared between two BLE devices at the time of pairing and allows the Central device to resolve the Peripheral address that would otherwise look random.

The Bluetooth Core Specification [24, Vol 3, Part C, App. A] recommends to renew Random Non-resolvable and Random Resolvable addresses every 15 minutes.

In addition to this address classification, we define two other categories based on the temporal persistence of the device address:

- Stable addresses: device addresses that are used by a device indefinitely or for an extended period of time, i.e. Public and Random Static addresses.
- Private addresses: device addresses that are supposed to change frequently, i.e. Random Non-resolvable and Random Resolvable addresses.

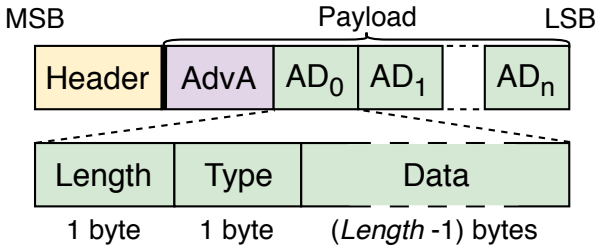

Figure 1: Representation of the BLE advertising PDU's header and payload. The payload part contains the Advertising Address (`AdvA`) field followed by Advertising Data (AD) structures. The *Length* field represents the length of the AD structure (excluding itself); *Type* specifies the nature of the following data and *Data* are the advertising data.

2.3 BLE advertisement

Advertisement is the name of the BLE discovery mechanism that allows Central devices to discover Peripherals in range prior to set up a connection. It is used to broadcast connectionless data such as device characteristics and supported services, but also data coming from services and custom applications. In addition to the frame header, advertisement packets can contain up to 31 bytes of cleartext data that can be collected and processed by any BLE device in range. Moreover, Central devices can query a Peripheral by sending a directed scan request that triggers back a scan response. The content of a scan response follows the same format as advertisement packets but will include different elements of information.

The advertising payload consists of the `AdvA` field (equal to the BD_ADDR of the Peripheral) followed by a sequence of one or more *Advertising Data* (AD) structures used to carry items of information. An AD structure is composed of a 1-byte field indicating the length of the AD structure (excluding itself), followed by a 1-byte field specifying the type[1] of the AD and finally, a sequence of up to 29 bytes of cleartext data (see Figure 1). A subset of common AD types is presented in Table 2.

3 METHODOLOGY

3.1 Data collection protocol

This work is based on a dataset of advertisement packets and scan responses that have been collected over 5 months by the authors during commute, work and leisure times. To collect the data, we used a Raspberry Pi single-board computer equipped with a CSR v4.0 Bluetooth USB dongle. In order to collect advertisement packets on BLE advertising channels, we developed a C software based on BlueZ libraries, the official Linux Bluetooth stack. Furthermore, as advertising data can be obtained from both advertisement packets and

[1] https://www.bluetooth.com/specifications/assigned-numbers/generic-access-profile

scan responses (see Section 2.3), we designed our C software to collect advertisement packets and automatically send scan requests to obtain potential scan responses from discovered Peripherals.

3.2 Structure of the data

Our dataset contains a set of records, each record corresponding to an advertising frame (advertisement packet or scan response). A record includes a timestamp along with a header and a payload. The data obtained from the data collection are stored as raw records that are organized as follows:

- Metadata: timestamp;
- Advertising PDU header: PDU type, BD_ADDR type, length of the advertising payload;
- Advertising payload: BD_ADDR, advertising data;

The advertising payload of each record is then parsed to extract AD structures and their various fields. To perform this task, we implemented our own Python parser based on online official Bluetooth SIG resources. The resulting data are then stored in a relational database comprising 179 attributes corresponding to the metadata, fields of the header and parsed AD structures.

3.3 Dataset preprocessing

Before performing our study, the dataset went through steps of anonymization and sanitization to minimize the privacy risks associated with the collected data and remove unwanted records.

Dataset anonymization. We focused on attributes corresponding to identifiers (Stable device addresses, device names, etc.) and on temporal data (timestamps). The following transformations were applied:

- Device addresses: the Stable BD_ADDR (Public and Random Static) have been pseudonymized through keyed-hashing[2] of the 24-bit lowest part (NIC), thus leaving the 24-bit highest part (OUI) unmodified.
- Names of persons: fields potentially containing names were sanitized by searching and removing substrings that were matching the pattern of a name ("`*.'s .*|*. 's .*`", "`*. de .*`") or strings from a dictionary[3] of names. Among the 1300 distinct device names of our dataset, we identified and redacted 142 names related to physical persons from 4287 advertisement records accounting for 0.05% of the dataset.
- Timestamps: the temporal information has been transformed from absolute (date and time) to relative (time elapsed since the beginning of the collection campaign).

Nevertheless, in order to be able to conduct the analysis of the distribution of addresses (Section 5.2), we kept the list of raw Random Static addresses that were stripped from any other information.

Dataset sanitization. BLE beacons are static devices deployed in the physical space to enable localization services. Thus, they are of little interest in the context of this study as they are not associated to an individual. The records associated to beacons have been filtered out the dataset based on regular expressions used to identify the main beacons standards[4]. Using this approach, we identified and removed a total of 50439 advertisement records accounting for 0.63% of our original dataset. In addition to the sanitization of beacons, we also removed all the records that have at least one malformed[5] AD structure leading to the suppression of 1958 additional advertisement records representing 0.02% of our original dataset.

[2]The key used during this process has been erased.

[3]We used the online french lastnames (https://www.insee.fr/fr/statistiques/3536630) and firstnames (https://www.insee.fr/fr/statistiques/2540004) databases of INSEE.

Table 1: Distribution of BD_ADDR and records among the device address types.

	Stable		Private	
	Public	Static	Non-res.	Res.
#BD_ADDR	4.4k	1.1k	9.5k	38.5k
#Records	2.9M	220k	740k	4M
#adv. packets	2.2M	130k	700k	2.6M
#scan responses	700k	90k	40k	1.4M

4 OBSERVATIONS ON THE DATASET

In this section, we present observations on the dataset used in this study. This dataset, collected over 5 months, contains about 8 million records and includes more than 53500 distinct BD_ADDR. Note that, since some of those addresses are random pseudonyms, the number of actual devices is expected to be smaller than the number of distinct device addresses.

4.1 Adoption of the LE Privacy feature

As described in Section 2.2, device addresses can be of 4 types: Public, Random Static (Stable category), Random Non-resolvable and Random Resolvable (Private category). Table 1 shows the distribution of BD_ADDR and records among the device address types in our dataset.

In term of volume, we observed that Private addresses (Random Resolvable and Random Non-resolvable) account for about 60% of the advertisement packets found in our dataset. This shows that the privacy-preserving mechanism of BLE is widely adopted by the industry. This also highlights that a number of devices are still using Stable addresses. In fact, our dataset includes more than 5500 Stable device addresses that can be trivially used to track the corresponding users [7, 8, 11]. Moreover, we observed that 80% of those Stable identifiers are of type Public, thus revealing the globally unique MAC address assigned by the manufacturer to the device.

Among the Private category, we computed that a large part of advertisement packets belongs to the Random Resolvable type (4M packets) representing more than 84% of the traffic generated by Private addresses. This suggests that the *resolvable* feature of LE Privacy is being endorsed by vendors over the *non-resolvable* option.

[4]*Apple* iBeacon (https://developer.apple.com/ibeacon/), *Google* Eddystone (https://github.com/google/eddystone) and *Radius Networks* AltBeacon (https://github.com/AltBeacon/spec).

[5]An AD structure that does not follow the format shown in Figure 1 and thus cannot be correctly parsed.

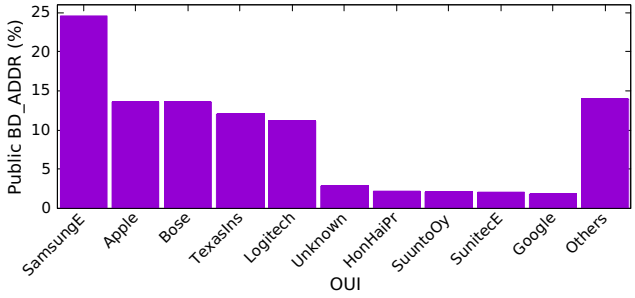

Figure 2: Distribution of OUI among Public device addresses.

4.2 Manufacturers OUI distribution

By definition, for device addresses falling in the Public type, it is possible to identify the manufacturer via the OUI (Organizationally Unique Identifier) of the address. The distribution of the top 10 OUI found within Public addresses is presented in Figure 2. We can observe the prevalence of 4 companies producing electronic devices (*Samsung*, *Apple*, *Bose* and *Logitech*) along with a chipset manufacturer (*Texas Instruments*). As they configure their products to use Public addresses, those companies are exposing their customers to tracking.

4.3 AD types distribution

Advertising data can contain a variety of AD types. In our dataset, we found that advertising payloads contain 2 distinct AD structures in average. The distribution of those AD types is highly concentrated as the six most common types (presented in Table 2) that are included in more than 80% of the advertisement packets in our dataset. Indeed, we computed that 6.3M advertisement records include at least one of those six types.

Among those most common values, we can observe types used to identify (`Complete Local Name`) or to describe the device and its capabilities (`Appearance` and `Flags`). Other types are dedicated to the advertisement of services (`Complete List of 16-bit Service Class UUIDs`) and transmission of data (`Manufacturer Specific Data` and `Service Data-16-bit UUID`).

Also, we can observe that the distribution of those AD types varies according to the address type. For instance, the `Complete Local Name` type, which is respectively found in 36.2% and 82.2% of Public and Random Static addresses, is less observed with Private addresses, which makes sense as those devices are supposed to remain anonymous.

5 DEVICE ADDRESS RANDOMIZATION SCHEME ANALYSIS

In this section, we leverage our dataset to analyze current implementations of the LE Privacy feature, and more particularly the lifetime and the uniform distribution of the random addresses.

5.1 Lifetimes of device addresses

The Bluetooth Core Specification [24, Vol 6, Part B, sec. 6.1] recommends a maximum duration of 15 minutes for both Random

Table 2: Presence of AD types in advertising payloads. Numerical values are fractions of addresses associated with each type.

AD type	Description	Stable		Private	
		Public	Static	Non-res.	Res.
`0x01`	`Flags`	87.6	88.6	10.9	77.2
`0xff`	`Manufacturer Specific Data`	84.7	51	99.3	93
`0x09`	`Complete Local Name`	36.2	82.2	0	1.7
`0x03`	`Complete List of 16-bit Service Class UUIDs`	30.1	11.7	0.1	18.7
`0x19`	`Appearance`	5.3	13.3	0	0.01
`0x16`	`Service Data-16-bit UUID`	4.6	54.4	0	17.8
Others	Other AD types	47.5	59.7	0.004	0.02
Overall	Devices that advertise at least one AD structure	99.8	99.2	100	99.9

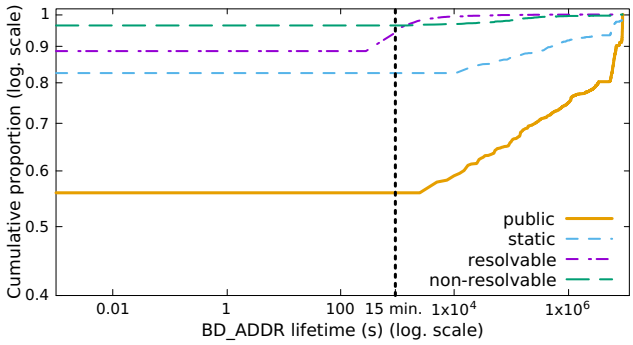

Figure 3: Empirical cumulative distribution function for lifetimes of device addresses.

Resolvable and Random Non-resolvable addresses to prevent tracking. On the opposite, Random Static (the third type of random addresses) and Public addresses are not supposed to change and are thus expected to be used for an extended duration. Due to our method of data collection, the duration for which a device stays in range during the collection cannot be controlled nor measured. As a consequence, lifetimes measured here should be considered as a conservative estimation. Indeed, a device may continue to use its BD_ADDR after it goes out of our collecting device range. We computed the lifetime of each device address and Figure 3 presents the cumulative distribution of those lifetimes for the 4 types of BD_ADDR.

First, we can observe that 41% of Public and 17% of Random Static BD_ADDR have a lifetime of more than 3 hours. The very short lifetime of the remaining Stable addresses is likely due to the fact that the device was only observed during a single and short period of time. On the other hand, we can observe that a vast majority of Private addresses have a very short lifetime as 89% of Random Resolvable and 96% of Random Non-resolvable addresses are observed for a duration below 1 second. We can also observe that 6% of Random Resolvable and 4% of Random Non-resolvable addresses

Table 3: Results of the *Kolmogorov-Smirnov* test on the random part of random addresses against a uniform distribution.

BD_ADDR type	Statistic	p-value
Random Static	0.049615	3.851678×10^{-6}
Random Non-resolvable	0.006928	0.153595
Random Resolvable	0.002908	0.636685

have a lifetime larger than 15 minutes. Some Private addresses have even been observed for more than 69 days.

Those observations show that the recommendation of the lifetime of a device address is enforced by most implementations. However, some implementations of Private addresses are not following the specifications, keeping the same address for an extended duration and thus exposing the users to tracking.

5.2 Uniformity of random addresses distribution

According to the Bluetooth Core Specification [24, Vol 2, Part H, sec. 2], random addresses should be generated using a FIPS compliant PRNG and thus should exhibit some characteristics expected from those PRNGs. In particular, it is expected that the output of those PRNGs follows a uniform distribution. We used the random addresses found within our dataset and tested whether this set of random values followed a uniform distribution[6].

For each random address found within our dataset, we extracted the part that has been generated with a PRNG[7] and submitted those values to the *Kolmogorov-Smirnov* (KS) statistical test [5]. The KS test is used to test whether one distribution of values differs substantially from theoretical expectations. It outputs the distance (the *statistic*) between the empirical distribution and the reference distribution as well as a significance score (the *p-value*) that gives the probability of obtaining this distance if the values were drawn from the reference distribution [13]. The null hypothesis is rejected when the p-value is below a certain threshold (values of 0.1, 0.05, and 0.01 are typically used).

This test was run on each type of random addresses and produced the results reported in Table 3. The p-values obtained for Random Resolvable and Random Non-resolvable addresses are high (> 0.15) while the p-value for Random Static is very small ($< 10^{-5}$). This means that the hypothesis that BD_ADDR are uniformly generated can be rejected for the Random Static addresses and kept for both Random Resolvable and Random Non-resolvable addresses. The non-uniform distribution of Random Static addresses can be clearly observed on Figure 4 that shows that some ranges of values are more common than others. In fact, we have identified that some categories of devices are using specific values for the higher part of the device address (the leftmost 24 bits). For instance, *Sony* SRS

[6]Since those random values have been produced by different instances and implementations, it is thus not possible to test most of the properties required by FIPS.

[7]Not all the bits of random BD_ADDR are supposed to be random. First, the 2 most significant bits are fixed to specify the type of the random address and are thus not random. Then, for Random Non-resolvable and Random Static addresses, the 46 remaining bits are random. Actually, for the Random Resolvable addresses, only 22 bits are random because the 24 least significant bits are computed from the random part *prand* and the IRK (see Section 2.2).

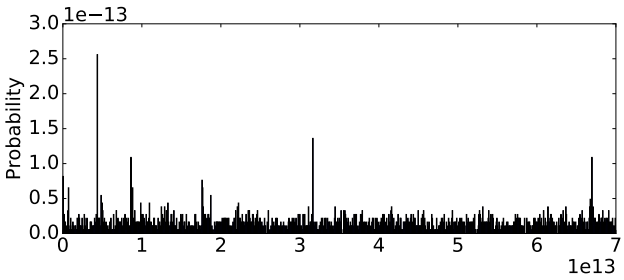

Figure 4: Distribution of Random Static addresses using a resolution of 1024 bins (size of a bin = 2^{36}).

Figure 5: Representation of a successful attack. The BLE device is randomizing its BD_ADDR (in italic) over time while keeping a static identifier (in bold) in the advertising data. Such a 5-byte identifier can be used by a passive attacker to link together the packets generated with the three different BD_ADDR.

portable speakers and *Samsung* Gear smartwatches respectively use `c7:d5:0b` (13 devices) and `dc:c1:c6` (10 devices) as a prefix. This suggests that some Random Static addresses are allocated by range, the same way MAC addresses are allocated by OUI, potentially revealing information on the type and the manufacturer of the device (see Section 7).

Although the set of Random Non-resolvable and Random Resolvable addresses appear to be uniformly distributed, this does not guarantee that all those addresses are generated with a FIPS compliant PRNG, and that there are no other flaws in the generation of those random addresses.

6 DEFEATING DEVICE ADDRESS RANDOMIZATION

According to the results of the previous section, it appears that globally, implementations of the address randomization scheme avoid common pitfalls. However, we will show in the following that data included in the rest of the advertising payload can contain information that could negate the protection provided by the address randomization.

6.1 Attacker model

In this section, we consider an external attacker which continuously monitors the traffic on BLE advertising channels. This attacker is thus able to capture the advertisement packets generated by BLE devices in range. Furthermore, we assume that the attacker

is passive: he only captures traffic and does not interact with the wireless channel through injection or jamming.

On the victim side, we only make the assumption that his device has its Bluetooth interface turned on and broadcasts advertisement packets.

The goal of the attacker is to track a device beyond the address randomization scheme based on the captured advertising traffic. More specifically, a successful attack is defined as the attacker being able to link two sets of advertisement packets generated by a single device but with two distinct BD_ADDR. Consequently, a successful attack links together two *pseudonyms* of a device.

Figure 5 shows an example where a device generates traffic with three different BD_ADDR, but it is possible to link together the three sets of advertisement packets based on a static identifier found within the advertising data.

6.2 Static advertising identifiers

6.2.1 Device names. `Complete Local Name` and `Shortened Local Name` are two AD types that are respectively used to advertise a complete and a shortened version of local names assigned to devices. From our dataset, we found that more than 1.7% of Random Resolvable and 0.06% of Random Non-resolvable addresses include either a `Complete Local Name` or a `Shortened Local Name` AD structure in their advertising payloads.

Moreover, we discovered that a number of device names advertised over Private addresses include a part corresponding to a unique identifier. For instance, device names of *Xiaomi* Amazfit fitness trackers match the following pattern `Amazfit-XXXX`, where `XXXX` are hexadecimal digits. We have identified several other families of devices such as *Boosted Boards* skateboards and *Nokia/Withings* smartwatches that also include an identifier in their device names. In practice, the size of such an identifier varies from 1 to 4 bytes and could be used to track the device over time.

From our observations on the Public addresses, we found that the identifier included in the device name is often the lowest part of the device address. This suggests that, for devices using Private addresses, the digital identifiers found within the device names could be a part of the device Public BD_ADDR.

6.2.2 Service UUIDs. Service UUIDs are identifiers that can be found in several AD structures such as the `Complete List of 16-bit Service Class UUIDs` or the `Service Data-16-bit UUID` structures. They can be advertised in a full (128 bits) or in a shortened version (16 or 32 bits).

In this context, a UUID is tailored to identify a service rather than a device. However, we found that some 128-bit UUIDs are customized by manufacturers to include a complete Public address in the least significant bits (LSB), thus forming a device unique identifier. We found a total of 180 distinct UUIDs within advertisement packets using Private addresses. Leveraging the registered OUI of the included Public addresses along with the advertised device names, we found that (1) some of those UUIDs belong to the *Nokia/Withings* manufacturer and (2) they are broadcasted by Steel smartwatches (9 devices). This practice introduces a secondary unique identifier that can be used to track the device over time.

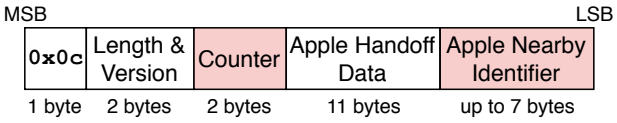

Figure 6: Format of the *Apple* Handoff `Manufacturer Specific Data` AD structure.

6.2.3 `LE Bluetooth Device Address`. The `LE Bluetooth Device Address` [23, Part A, sec. 1.16] AD type can be used by a Peripheral to broadcast a local device address and its class (Public or random). Nevertheless, the Bluetooth Core Specification [23, Part A, sec. 1] requires not to include such an AD structure in the advertising data. Especially, this is important for devices that use random BD_ADDR. However, we discovered that a number of devices using random addresses are including this AD structure in their advertising data and expose their Public device addresses. Leveraging the OUI of those advertised addresses, we found out that manufacturers such as *LG Innotek* and *Arcadyan* expose their Public BD_ADDR this way.

6.3 Proximity protocols

In advertisement packets, AD structures such as the `Manufacturer Specific Data` and the `Service Data-16-bit UUID` structures can be used to carry application specific data. In the following, we focus on two prevalent proximity protocols found within advertisement packets and discuss on the fact that their broadcasted data can expose information that can reduce or even negate the address randomization mechanism.

6.3.1 Apple Handoff. The *Apple* Handoff[8] protocol is part of the *Apple* Continuity features and was introduced in *Apple* iOS 8 and *Apple* OS X Yosemite operating systems. It allows users to switch from one *Apple* device to another and continue an ongoing activity seamlessly. To enable this feature, *Apple* devices rely on *Apple* Handoff data carried by the `Manufacturer Specific Data` AD structure. The format of such an *Apple* Handoff structure is presented in Figure 6. It starts with an identifier indicating the type of the advertising data, in this case *Apple* Handoff (`0x0c`), and includes several fields such as a frame *counter*, the *Apple* Handoff data and an *Apple Nearby Identifier*.

Counter. The counter is a 2-byte integer incremented over time, but that can remain unchanged in several consecutive advertisement packets. We found that this counter is not reset when the BD_ADDR changes in the randomization scheme (see Figure 7). This means that based on this counter, it is possible to link different addresses belonging to the same device, as it has been shown with 802.11 probe requests frames [26].

Apple Nearby Identifier. According to the *Apple* iOS 12.3 security guide [2], the *Apple* Nearby Identifier is used for the mutual identification of devices linked to the same *Apple* iCloud account.

To prevent tracking based on this identifier, the *Apple* Nearby protocol periodically generates a new identifier every time the

[8] https://support.apple.com/en-ca/HT204681#handoff

	Time (s)	BD_ADDR	Apple Handoff Data		
			Cnt	Data	Nearby Id
(1)	899.256	59:07:ee:1e:6c:72	0003	1e47..	10050b1c4f087d
	899.820	59:07:ee:1e:6c:72	0103	a135..	10050b1c4f087d
	900.003	76:46:5d:85:9e:f2	0103	a135..	10050b1c0b5c1f
	900.252	76:46:5d:85:9e:f2	0203	0f48..	10050b1c0b5c1f
(2)	1799.762	76:46:5d:85:9e:f2	-	-	10050b1c0b5c1f
	1799.990	76:46:5d:85:9e:f2	-	-	10050b1c0b5c1f
	1800.091	6d:01:ff:0a:52:84	-	-	10050b1c0b5c1f
	1800.203	6d:01:ff:0a:52:84	-	-	10050b1c9d88fb

Figure 7: Sequence of *Apple* Handoff advertisement packets showing that (1) the counter is not reset after the change of BD_ADDR at 900.003 and (2) the BD_ADDR and *Apple* Nearby Identifier changes are not synchronized. At 1800.091, the old *Apple* Nearby Identifier is used with the new BD_ADDR.

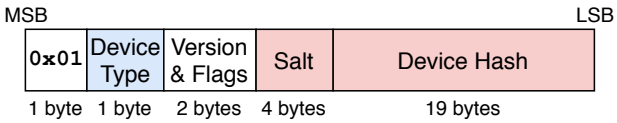

Figure 8: Format of the *Microsoft* CDP `Manufacturer Specific Data` AD structure.

BD_ADDR is changed. However, based on our observations on an *Apple* iPhone 6 and an *Apple* iPhone 8 smartphone, the change of the *Apple* Nearby Identifier is not completely synchronized with the change of BD_ADDR: for a short duration after the BD_ADDR change, the old *Apple* Nearby Identifier is used with the new device address (see Figure 7). This means that based on the *Apple* Nearby Identifier, it is possible to link two BD_ADDR used by the same device.

In our dataset, we found that among the 31000 Private addresses that include an *Apple* `Manufacturer Specific Data` AD structure, more than 79% of them include an *Apple* Nearby Identifier field.

6.3.2 Microsoft Connected Devices Platform (CDP). The *Microsoft* CDP [19] protocol provides a discovery system for *Microsoft* devices to authenticate and verify themselves as well as a way to exchange data. Similarly to the *Apple* Handoff protocol, data of *Microsoft* CDP are carried by the `Manufacturer Specific Data` AD structure. Based on the *Microsoft* CDP [19] specifications along with empirical observations on advertisement packets, we have identified the format of the *Microsoft* CDP structure (see Figure 8). In particular, it includes an identifier that is derived from a salted hashing of the unique device identifier [19, sec. 2.2.2.2.3].

Using a BLE-enabled laptop running *Microsoft* Windows 10 left idled, we found that the lifetime of the BD_ADDR is around 16 minutes while the *Device Hash* lasts for approximately 60 minutes (see Figure 9). First, we can observe that Random Non-resolvable addresses used to advertise *Microsoft* CDP data last longer than the Bluetooth Core Specification recommended duration of 15 minutes. Then, the switch to a new *Device Hash* is not synchronized with the change of BD_ADDR. In fact, the *Microsoft* CDP advertising data undermine the address randomization scheme through the exposure of an identifier whose lifetime overlaps the one of the BD_ADDR.

Time (s)	BD_ADDR	Microsoft CDP Data
		Device Hash
959.522	37:ee:cb:91:79:0a	db950efc53eff7e427f2a91ae9a67b...
959.719	18:e3:48:43:af:84	db950efc53eff7e427f2a91ae9a67b...
1919.074	2d:39:47:eb:2c:e8	db950efc53eff7e427f2a91ae9a67b...
2879.527	19:fc:04:f1:f3:9a	db950efc53eff7e427f2a91ae9a67b...
3599.189	19:fc:04:f1:f3:9a	4658a402b7da02e09585cb8c4aa1c7...

Figure 9: Sequence of *Microsoft* CDP advertisement packets in which the lifetime of the *Device Hash* overlaps the BD_ADDR randomization scheme. At 959.719, the BD_ADDR changes while the *Device Hash* remains identical until 3599.189.

From the dataset, we computed that more than 89% of Random Non-resolvable addresses follow the format of the data shown in Figure 8. Leveraging the *Device Type* field found within the *Microsoft* CDP data (see Figure 8), we found that 96% of such Random Non-resolvable addresses are broadcasted from *Microsoft* Windows 10 computers (desktops and laptops). This suggests that a large fraction of the BLE-enabled *Microsoft* devices are in fact trackable, despite the use of address randomization. In addition, it appears that most BLE-enabled *Microsoft* Windows 10 computers are affected by this issue as we found that *CDPUserSvc*, the service responsible for the broadcast of those *Microsoft* CDP messages, is enabled by default.

7 INFERRING INFORMATION FROM ADVERTISING DATA

Advertising data include information that can cause privacy threats beyond tracking. Indeed, we found that the characteristics of a device such as its manufacturer, model or type can be inferred. Those elements of information can become a privacy threat in the context of inventory attacks and profiling [9], where an attacker will try to infer information based on the device carried by the user. For instance, a medical device can reveal a medical condition and profiling individuals based on their devices is a direction taken by some companies [28]. This section presents a review of the information that can be inferred from advertising data. Our findings are summarized in Table 4.

7.1 Device model and manufacturer

Public BD_ADDR. The Public device address is an identifier that can be exposed in the `AdvA` field of the advertising payload or in other AD structures (see Section 6). This identifier can be simply leveraged to identify the device manufacturer because, as a MAC address [24, Vol 2, Part B, sec. 1.2], its 24 most significant bits (MSB) part corresponds to the OUI of the manufacturer.

Furthermore, since MAC addresses are typically allocated by batch, a Public address could be used to identify the device model. This approach has been demonstrated with Wi-Fi MAC addresses [17] and could be reproduced with BLE Public BD_ADDR. As shown in Section 5.2, some Random Static addresses also appear to be allocated by range. In those cases, the device model could be also inferred from the Random Static address.

Table 4: Summary of the information that can be inferred from BLE advertising data.

Advertising elements	*Manuf.*	*Model*	*Type*	*Owner id.*
Public BD_ADDR	✓		✓	
Device names	✓	✓	✓	✓
`Manufacturer Specific Data`	✓	✓	✓	
Company IDs	✓		✓	
Manufacturer data	✓	✓	✓	
Service UUIDs	✓	✓	✓	
`Appearance`			✓	
`Class of Device`			✓	

Device names. As human friendly descriptors, the `Complete Local Name` and `Shortened Local Name` AD structures often include identifiers of manufacturers and models of devices. As a result, those structures are two other sources of information that can be used to deduce the manufacturer.

For instance, we identified that devices including patterns such as `LE-Bose` and `Jabra` in their device names are respectively *Bose* and *Jabra* manufactured headsets. In addition, we found that devices such as *Garmin* fitness trackers and *Bang&Olufsen* headphones include the name of their models in their device names.

Company IDs and service UUIDs. Advertising data can include several identifiers tight to companies. For instance, this is the case of the company IDs[9] that are included in the `Manufacturer Specific Data` AD structure. Like company IDs, members UUIDs[10] are 16-bit identifiers assigned by the Bluetooth SIG. Such members UUIDs can be included in the `Service Data-16-bit UUID` and `Complete List of 16-bit Service Class UUIDs` AD structures. The presence of those indicators suggests that the device features a service that has been designed by the corresponding company. Indeed, for the sake of compatibility, it is possible that a service is implemented by other companies than the one that has designed it. However, we found that, in practice, those identifiers are generally included by devices of the associated manufacturer.

In addition, some companies can design their own 16-bit UUIDs. This information is not publicly available but can be obtained analyzing the advertisement traffic generated by the devices. Similarly, the 128-bit service UUIDs can be generated by manufacturers and can be leveraged as well. For instance, the `adab09ad-6e7d-4601-bda2-bffaa68956ba` custom 128-bit service UUID broadcasted by *Fitbit* fitness trackers is associated with the *Fitbit* manufacturer.

From our dataset, we computed that more than 78% of Stable and 95% of Private addresses are associated with at least one company identifier (company ID or members UUID) in the advertising payload, indicating that the manufacturer could be identified.

[9]https://www.bluetooth.com/specifications/assigned-numbers/company-identifiers
[10]https://www.bluetooth.com/specifications/assigned-numbers/16-bit-uuids-for-members

7.2 Device type

Public BD_ADDR:. As previously observed, the manufacturer of a device can be derived from its Public BD_ADDR. Nevertheless, when this manufacturer is focused on a single class of products, it becomes possible to infer the type of the device from the Public address. For instance, the OUI associated with *Nintendo* and *OculusVR* respectively reveal a video game platform and a virtual reality headset.

Service UUIDs. Service UUIDs are processed by Central to discover services offered by nearby Peripheral before establishing a connection. Some service UUIDs assigned by the Bluetooth SIG reveal particular characteristics of a device that can lead to the inference of its type. For instance, looking at the online Bluetooth SIG database[11] of 16-bit service UUIDs, we can infer that devices which broadcast the *Running Speed and Cadence* service UUID (`0x1814`) are likely fitness trackers while those that advertise the *Insulin Delivery* service UUID (`0x183a`) are healthcare devices.

In addition, as mentioned in Section 7.1, a custom 128-bit service UUID can disclose attributes of a device such as its manufacturer or its model that could lead to an implicit identification of its type. For instance, the `aa745be2-9025-4bf2a318-91f3dba2999f` 128-bit service UUID is associated with *Garmin* Nuvi GPS while `0000de00-3dd4-4255-8d62-6dc7b9bd5561` is advertised by *Nikon* cameras. Nevertheless, even though those service UUIDs are customized by the manufacturers, we discovered that some of them are hardcoded in open-source specifications and codes. This is especially the case with the *Google* hearing aid[12] specifications and the *Sony* SmartBand SWR-10[13] open API. Therefore, such online resources can be also leveraged to identify the type of a device.

`Manufacturer Specific Data`. The data carried by the `Manufacturer Specific Data` AD structure can include fields indicating the type of the device. Indeed, this is the case of *Garmin* (company ID `0x0087`), for which the 16 MSB of data disclose the model of the device: `0x0657` and `0x0802` respectively indicate *Garmin* Forerunner 620 and *Garmin* Fenix 3 fitness smartwatches. Similarly, for *Apple* (company ID `0x004c`), the 8 MSB of data expose the type of the device: `0x07` and `0x0b` respectively disclose *Apple* AirPods earphones and *Apple* Watch smartwatches. We also discovered that other experimental results[14] support and confirm some of our observations.

Moreover, we found that the *Microsoft* CDP proximity protocol includes a *Device Type* field (see Figure 8) that provides information on the nature of the device. As a consequence, when the value of this field is filled accordingly to the *Microsoft* CDP specifications [19, sec. 2.2.2.2.3], we can infer the type of the advertising device by lookup in the specifications: Xbox One (`0x01`), Windows 10 Desktop (`0x09`), Windows 10 Phone (`0x0b`), etc.

[11]https://www.bluetooth.com/specifications/gatt/services
[12]https://source.android.com/devices/bluetooth/asha
[13]https://github.com/fbarriga/sony-smartband-open-api
[14]https://github.com/reelyactive/advlib#manufacturers

`Appearance` *and* `Class of Device`. The Bluetooth Core Specification defines two AD types, `Appearance`[15] and `Class of Device`[16], that are designed to carry information about the nature of a device and more specifically its external appearance. For instance, the `Appearance` code `0x00c1` disclose a sport watch, while the `Class of Device` code `0x022808` indicates a toy vehicle. More worrisome, certain codes indicate specific medical devices: `Appearance` codes such as `0x0d00` and `0x0d48` respectively betray a glucose monitor and an insulin pen, while `Class of Device` codes such as `0x02292c` and `0x022930` respectively denote a medication monitor system and a knee prosthesis.

In our dataset, we found that more than 5.3% of Public and 13.3% of Random Static addresses include either an `Appearance` or a `Class of Device` AD structure in their advertising data. Also, for Private addresses, only 0.01% of Random Resolvable BD_ADDR broadcast data carried by the `Appearance` AD structure while the `Class of Device` type is not advertised. Among the most common appearances and class of devices, we found that devices such as *Tile* iTag keyrings, *Giant* RideSense cycling sensors and *Diggro* ID10x heart rate belts explicitly advertise their external appearances allowing a passive observer to infer the type of the device based on those information.

7.3 Identity of the owner

The `Complete Local Name` and `Shortened Local Name` AD structures carry a textual description of the device that can be often customized by users. As a result, a number of those device names include the identity of the owner under the form of a full name, a firstname, a lastname or a nickname. For instance, we found that *Bose* headsets and *Apple* smartphones broadcast such device names that include the name of the owner: `LE-Bose Alice` and `Bob's iPhone`. During our dataset anonymization process (see Section 3.3), we identified and subsequently anonymized a total of 10.9% of the 1300 distinct device names advertised from 0.4% of the addresses of the dataset.

8 RELATED WORK

Exposure of personal information through wireless networks have been the subject of numerous works. For instance, it has been highlighted that 802.11 local service discovery exposes identifying information [21], relationships [21] and social links [6]. Most importantly, link layer identifiers such as MAC addresses can be used to track users in the physical world [4, 10].

Several works have focused on the privacy aspects of BLE. Das et al. demonstrated [7] how the BLE traffic generated by a fitness wristband can be leveraged to infer the activity of its owner.

In BLEB [11], the authors introduced a BLE botnet of smartphones for wide scale tracking and highlighted that BLE privacy features are not used by some wearable devices. Fawaz et al. introduced *BLE-Guardian* [8], a solution to protect BLE users from privacy threats. In addition, they discussed several privacy issues of BLE advertisement, some of which are analyzed in detail in our paper. In particular, they noted that some manufacturers do not use random addresses, and that some of the temporary identifiers are used for long periods of time. We consolidate those observations by providing an updated and detailed review of those issues and identifying new privacy concerns such as those introduced by the proximity protocols.

Recently, Becker et al. demonstrated [3] that the LE Privacy feature can be defeated leveraging the advertising payload. We complement this work by deeply investigating the device address randomization scheme not limiting us to empirical observations. Moreover, based on our large-scale measurement, we are able to provide insights on the current adoption and implementations of the LE Privacy feature in the wild. Finally, we go beyond tracking concerns showing how advertising data can leak private information.

Martin et al. showed [17] that the MAC address of 802.11 devices can reveal the type of the device in addition of its manufacturer. Our study shows that the BLE advertising payload can be used to this aim.

In their blog posts [14, 15], Lester and Stone pointed out that some elements contained in BLE advertisement packets can be leveraged to identify the nature of the device. We complement those observations presenting a comprehensive analysis of the AD structures that can expose the model and the type of a device.

Several tools for parsing the content of BLE advertising data have been produced by the community. In particular, the *advlib* [12] JavaScript library and the *RaMBLE* [14] Android mobile application are capable of decomposing and interpreting a number of AD structures. Our Python parser consolidates the features of those contributions and add several elements such as the decomposition of 128-bit UUIDs, manufacturer specific data and service data, to name a few.

To reduce the risk of user tracking, several approaches have been proposed such as the use of temporary identifiers [10] and protocols based on cryptographic primitives [21, 27].

Use of disposable link layer identifiers [10] have been progressively adopted by the industry [26]. However, several works [16, 26] have shown the limitation of MAC address randomization in the context of 802.11. Vanhoef et al. [26] and Martin et al. [16] showed that MAC randomization can be defeated thanks to the data contained in the body of the frame. Several of the issues affecting BLE and discussed in our paper, such as non-reset counters and static identifiers, were also found in 802.11 by those works [16, 26]. Finally, timing of frames has been also considered as a way to defeat 802.11 address randomization [18].

9 CONCLUSION

This study presented a detailed analysis of the privacy provisions and issues in current implementations of the BLE advertising mechanism. First, we found that even if address randomization is widely adopted by vendors, a number of devices still use Stable addresses, exposing their owners to tracking. Furthermore, we found that this randomization scheme is not always correctly implemented as some devices exceed the recommended maximum duration of the random addresses. We also identified that, despite the use of a random device address, the content of the advertisement packets can be used to track users. Indeed, the address randomization scheme can be rendered useless by counters and static identifiers

[15] https://www.bluetooth.com/wp-content/uploads/Sitecore-Media-Library/Gatt/Xml/Characteristics/org.bluetooth.characteristic.gap.appearance.xml
[16] https://www.bluetooth.com/specifications/assigned-numbers/baseband

included by some devices in their advertising data. In particular, we found that custom data included in proximity protocols of *Apple* and *Microsoft* can be leveraged to defeat the randomization scheme. Finally, we showed how the advertising data can be mined to infer the nature of a BLE device, exposing the owner to further privacy threats such as inventory attacks and profiling.

We informed the manufacturers of the privacy issues identified in this paper[17].

A part of the issues presented in this paper are the results of manufacturers that do not follow the BLE specifications. For instance, the inclusion of the `LE Bluetooth Device Address` AD structure in advertising data is clearly against the specifications. Nevertheless, most of the issues are the results of practices that are not forbidden by the specifications. For instance, using a custom UUID that includes a device address is compliant with the specifications. Similarly, there are no instructions on the nature of the information that can be carried by the `Manufacturer Specific Data` and `Service Data` AD structures. In fact, the BLE specifications do not provide any guidelines about the content of the advertising payload with regard to the privacy implications.

Currently, the privacy considerations of the BLE specifications are limited to the management of random link layer identifiers. However, as we showed in this paper, other elements of the advertising mechanism can be used to breach privacy. Thus, it is important to complement the specifications with additional requirements that would cover privacy issues such as those discussed in this paper.

ACKNOWLEDGEMENTS

This work was supported by the INSA Lyon - SPIE ICS IoT chair and the H2020 SPARTA Cybersecurity Competence Network project.

REFERENCES

[1] 2014. IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture. *IEEE Std 802-2014 (Revision to IEEE Std 802-2001)* (June 2014), 1–74. https://doi.org/10.1109/IEEESTD.2014.6847097

[2] Apple. 2019. *iOS Security - iOS 12.3.* https://www.apple.com/business/site/docs/iOS_Security_Guide.pdf Accessed: 2019-08-30.

[3] Johannes K Becker, David Li, and David Starobinski. 2019. Tracking Anonymized Bluetooth Devices. *Proceedings on Privacy Enhancing Technologies* 2019, 3 (2019), 50–65.

[4] B. Bonne, A. Barzan, P. Quax, and W. Lamotte. 2013. WiFiPi: Involuntary tracking of visitors at mass events. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a.* 1–6. https://doi.org/10.1109/WoWMoM.2013.6583443

[5] Indra Mohan Chakravarti, Radha Govira Laha, and Jogabrata Roy. 1967. Handbook of methods of applied statistics. *Wiley Series in Probability and Mathematical Statistics (USA) eng* (1967).

[6] Mathieu Cunche, Mohamed-Ali Kaafar, and Roksana Boreli. 2014. Linking wireless devices using information contained in Wi-Fi probe requests. *Pervasive and Mobile Computing* 11 (April 2014), 56–69. https://doi.org/10.1016/j.pmcj.2013.04.001

[7] Aveek K Das, Parth H Pathak, Chen-Nee Chuah, and Prasant Mohapatra. 2016. Uncovering privacy leakage in ble network traffic of wearable fitness trackers. In *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications.* ACM, 99–104.

[8] Kassem Fawaz, Kyu-Han Kim, and Kang G Shin. 2016. Protecting Privacy of BLE Device Users. In *25th USENIX Security Symposium (USENIX Security 16).* 1205–1221.

[9] Ben Greenstein, Ramakrishna Gummadi, Jeffrey Pang, Mike Y Chen, Tadayoshi Kohno, Srinivasan Seshan, and David Wetherall. 2007. Can Ferris Bueller Still Have His Day Off? Protecting Privacy in the Wireless Era.. In *HotOS.*

[10] Marco Gruteser and Dirk Grunwald. 2005. Enhancing location privacy in wireless LAN through disposable interface identifiers: a quantitative analysis. *Mobile Networks and Applications* 10, 3 (2005), 315–325.

[11] Taher Issoufaly and Pierre Ugo Tournoux. 2017. BLEB: Bluetooth Low Energy Botnet for large scale individual tracking. In *Next Generation Computing Applications (NextComp), 2017 1st International Conference on.* IEEE, 115–120.

[12] Mohamed Imran Jameel and Jeffrey Dungen. 2015. Low-power wireless advertising software library for distributed M2M and contextual IoT. In *Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on.* IEEE, 597–602.

[13] TW Kirkman. 1996. Statistics to use: Kolmogorov-Smirnov test. *College of Saint Benedict and Saint John's University. Retrieved October* 7 (1996), 2008.

[14] Scott Lester. 2015. The Emergence of Bluetooth Low Energy. (2015). https://www.contextis.com/blog/the-emergence-of-bluetooth-low-energy Accessed: 2019-08-30.

[15] Scott Lester and Paul Stone. 2016. Bluetooth LE - Increasingly popular, but still not very private. (2016). https://www.contextis.com/en/blog/bluetooth-le-increasingly-popular-still-not-very-private Accessed: 2019-08-30.

[16] Jeremy Martin, Travis Mayberry, Collin Donahue, Lucas Foppe, Lamont Brown, Chadwick Riggins, Erik C Rye, and Dane Brown. 2017. A study of MAC address randomization in mobile devices and when it fails. *Proceedings on Privacy Enhancing Technologies* 2017, 4 (2017), 365–383.

[17] Jeremy Martin, Erik Rye, and Robert Beverly. 2016. Decomposition of MAC address structure for granular device inference. In *Proceedings of the 32nd Annual Conference on Computer Security Applications.* ACM, 78–88.

[18] Celestin Matte, Mathieu Cunche, Franck Rousseau, and Mathy Vanhoef. 2016. Defeating MAC Address Randomization Through Timing Attacks. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks - WiSec '16.* ACM Press.

[19] Microsoft. 2019. *Microsoft Connected Devices Platform Protocol Version 3.* https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-cdp/f5a15c56-ac3a-48f9-8c51-07b2eadbe9b4 Accessed: 2019-08-30.

[20] Dieter Oosterlinck, Dries F Benoit, Philippe Baecke, and Nico Van de Weghe. 2017. Bluetooth tracking of humans in an indoor environment: An application to shopping mall visits. *Applied geography* 78 (2017), 55–65.

[21] Jeffrey Pang, Ben Greenstein, Srinivasan Seshan, and David Wetherall. 2007. Tryst: The Case for Confidential Service Discovery.. In *HotNets*, Vol. 2. 1.

[22] Bluetooth SIG. 2010. *Bluetooth Core Specification v4.0.* https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=456433 Accessed: 2019-08-30.

[23] Bluetooth SIG. 2019. *Bluetooth Core Specification Supplement v8.0.* https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=457081 Accessed: 2019-08-30.

[24] Bluetooth SIG. 2019. *Bluetooth Core Specification v5.1.* https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=457080 Accessed: 2019-08-30.

[25] Bluetooth SIG. 2019. *Bluetooth Market Update 2019.* Technical Report. https://www.bluetooth.com/wp-content/uploads/2018/04/2019-Bluetooth-Market-Update.pdf Accessed: 2019-08-30.

[26] Mathy Vanhoef, Celestin Matte, Mathieu Cunche, Leonardo S. Cardoso, and Frank Piessens. 2016. 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 (ASIA CCS '16).* ACM, New York, NY, USA, 413–424. https://doi.org/10.1145/2897845.2897883

[27] Ford-Long Wong and Frank Stajano. 2005. Location privacy in bluetooth. In *European Workshop on Security in Ad-hoc and Sensor Networks.* Springer, 176–188.

[28] Asaf Zomet and Shlomo Reuben Urbach. 2016. Privacy-aware personalized content for the smart home. US Patent App. 14/638,937.

[17] Notifications have been sent to: *Apple*, *Arcadyan*, *Bang&Olufsen*, *Boosted Boards*, *Bose*, *Diggro*, *Fitbit*, *Garmin*, *Giant*, *Google*, *Jabra*, *LG Innotek*, *Logitech*, *Microsoft*, *Nikon*, *Nintendo*, *Nokia/Withings*, *OculusVR*, *Samsung*, *Sony*, *Tile* and *Xiaomi*.