Protecting Privacy of BLE Device Users

Kassem Fawaz* Kyu-Han Kim† Kang G. Shin*
*The University of Michigan †Hewlett Packard Labs

Abstract

Bluetooth Low Energy (BLE) has emerged as an attractive technology to enable Internet of Things (IoTs) to interact with others in their vicinity. Our study of the behavior of more than 200 types of BLE-equipped devices has led to a surprising discovery: the BLE protocol, despite its privacy provisions, fails to address the most basic threat of all—hiding the device's presence from curious adversaries. Revealing the device's existence is the stepping stone toward more serious threats that include user profiling/fingerprinting, behavior tracking, inference of sensitive information, and exploitation of known vulnerabilities on the device. With thousands of manufacturers and developers around the world, it is very challenging, if not impossible, to envision the viability of any privacy or security solution that requires changes to the devices or the BLE protocol.

In this paper, we propose a new device-agnostic system, called `BLE-Guardian`, that protects the privacy of the users/environments equipped with BLE devices/IoTs. It enables the users and administrators to control those who discover, scan and connect to their devices. We have implemented `BLE-Guardian` using Ubertooth One, an off-the-shelf open Bluetooth development platform, facilitating its broad deployment. Our evaluation with real devices shows that `BLE-Guardian` effectively protects the users' privacy while incurring little overhead on the communicating BLE-devices.

1 Introduction

Bluetooth Low Energy (BLE) [4] has emerged as the *de facto* communication protocol in the new computing paradigm of the Internet of Things (IoTs) [8, 9, 15, 23, 24, 39]. In 2013, over 1.2 billion BLE products were shipped [9], with this number expected to hit 2.7 billion in 2020 [3]. BLE-equipped products are embedded and used in every aspect of our lives; they sense nearby objects, track our fitness, control smart appliances and toys provide physical security, etc. The BLE protocol owes this proliferation to its low energy and small processing footprint as well as its support by most end-user devices [20], such as PCs, gateways, smartphones, and tablets.

A BLE-equipped device advertises its presence to let interested nearby devices initiate connections and glean relevant information. These advertisements, however, are a double-edged sword. An unauthorized, potentially malicious, party can use these advertisements to learn more about the BLE-equipped devices of a certain user or in a specific environment [22], generally referred to in literature as the *inventory attack* [42]. Revealing the device's presence is the stepping stone toward more serious privacy and security attacks with grave consequences in the case of medical devices for example, especially for high-value targets [31].

The BLE specification contains some privacy provisions to minimize the effects of inventory attacks and ensuing threats, namely *address randomization* and *whitelisting*. A BLE device is supposed to randomize its address to prevent others from tracking it over time. Moreover, only devices with a pre-existing trust relationship (whitelisted devices) are supposed to access the BLE-equipped device.

In this paper, we first analyze how existing BLE's privacy measures fare in the real-world deployments through our own data-collection campaign. To the best of our knowledge, this is the first study that systematically analyzes threats to the BLE-equipped devices in the wild. We recruited participants from our institution and the PhoneLab testbed [27] to collect the BLE advertisements in their vicinity. We have collected and analyzed the advertisements from 214 different types of BLE-equipped devices. Analyzing our dataset has led to a surprising discovery: BLE advertisements, due to poor design and/or implementation, leak an alarming amount of information that allows the tracking, profiling, and

fingerprinting of the users. Furthermore, some devices allow external connections without an existing trust relationship. Unauthorized entities can access unsecured data on the BLE-equipped devices that might leak sensitive information and potentially inflict physical harm to the bearer.

Almost all of the existing approaches addressing some of the above threats rely on mechanisms that necessarily include changes to the protocol itself or to the way the BLE-equipped devices function [21, 40]. Changing the operation of such devices, post-production, requires their patching by securely pushing a firmware update. With thousands of manufacturers and developers around the world, it is very challenging, sometimes impossible, to guarantee firmware patches to the millions of already deployed devices [11]. Even a security-aware user might lack the ability to update the firmware of a BLE-equipped device. Patch management is, therefore, the leading security challenge in the emerging IoTs [10, 19] (including BLE-equipped devices) for many reasons:

- Manufacturers might lack the ability to apply OTA updates [1] for some deployed BLE-equipped devices because they (such as a BLE-equipped pregnancy test) are neither programmable nor equipped with an Internet connection.
- Customers might neither receive news about the update nor be able to apply an update even if available. For example, a month after the 2013 "Foscam" webcams hacking incident, 40,000 of 46,000 vulnerable cameras were not updated although a firmware update was available [17].
- Companies do not have enough financial incentives or resources to maintain the devices post deployment [34]. For example, Samsung discontinued two lines of smart refrigerators after 2012 so that customers can't receive updates for their purchased refrigerators [6].

There is, therefore, a need for a new class of practical approaches to mitigate the privacy threats to BLE-equipped devices. In this paper, we seek to answer the following related question: *can we effectively fend off the threats to BLE-equipped devices: (1) in a device-agnostic manner, (2) using COTS (Commercial-Off-The-Shelf) hardware only, and (3) with as little user intervention as possible?*

We present `BLE-Guardian` as an answer to the above question. It is a practical system that protects the user's BLE-equipped devices so that *only* user-authorized entities can discover, scan, or connect to them. `BLE-Guardian` relies on an external and off-the-shelf Bluetooth radio as well as an accompanying application. Therefore, a user can easily install (and control) `BLE-Guardian` to any BLE gateway, be it a smartphone, tablet, PC, Raspberry PI, Artik-10, etc. The external radio achieves the physical protection, while the application, running on the gateway, enables the user to interact with `BLE-Guardian`.

`BLE-Guardian` provides privacy and security protection by targeting the root of the threats, namely the advertisements. In particular, `BLE-Guardian` opportunistically invokes reactive jamming to determine the entities that can observe the device existence through the advertisements (*device hiding* module), and those that can issue connection requests in response to advertisements (*access control* module). In a typical BLE environment, however, achieving `BLE-Guardian`'s objective is rather challenging. Many BLE-equipped devices, including the ones to be protected, advertise on the same channel; while at the same time other devices, in response to advertisements, issue scan and connection requests. The timing is of an essence for `BLE-Guardian`; it invokes jamming at the right time for the right duration. Therefore, `BLE-Guardian` does not inadvertently harm other devices, preserves the ability of authorized entities to connect the BLE-equipped device, and always hides the BLE-equipped device when needed.

More than one device might be authorized to connect to the BLE-equipped device. `BLE-Guardian` differentiates the scan and connection requests originating from authorized devices versus those that are fraudulent. This is particularly challenging as the BLE advertisement channel lacks any authentication mechanism for the advertisements and connections. `BLE-Guardian` utilizes Bluetooth classic as an out-of-band (OOB) channel to authorize a device after obtaining the user's permission. It uses the OOB channel to instruct the connecting device to issue ordinary connection requests with (varying) special parameters that other unauthorized devices can't predict. It also alerts the user when unauthorized parties attempt connection to the user's BLE devices.

`BLE-Guardian` achieves its objectives with minimum requirements from the external radio. Effectively, `BLE-Guardian` operates with a radio that offers only the basic capabilities of reception and transmission on the BLE channels. As a result, `BLE-Guardian` avoids employing sophisticated and customized (thus impractical) radios and signal processing approaches.

We implement `BLE-Guardian` using the commercially available Ubertooth One[1] USB dongle so that `BLE-Guardian` can be easily installed on any BLE gateway. We also implement accompanying apps for different BLE gateways, such as Android and Raspberry PI. We evaluate `BLE-Guardian` using several BLE devices for different real-world scenarios, where we assess its effectiveness in combating privacy threats, its low over-

[1] https://greatscottgadgets.com/ubertoothone/

head on the channel and devices, and little disruption to the operation of legitimate BLE devices. In particular, BLE-Guardian is able to protect up to 10 class-2 target BLE-equipped devices within a 5m range with less than 16% energy overhead on the gateway.

The rest paper is organized as follows. Section 2 discusses the related work. Section 3 provides the necessary BLE background. Section 4 states the privacy threats arising from BLE advertisements through our data-collection campaign. Section 5 details the design of BLE-Guardian. Section 6 presents the implementation of BLE-Guardian and evaluates its effectiveness. Finally, the paper concludes with Section 7.

2 Related Work

There have been limited efforts related to BLE devices that target the security and privacy threats resulting from the devices revealing their presence. The only exception is the work by Wang [40], where a privacy enhancement is proposed for BLE advertisements to ensure confidentiality and prevent replay attacks as well as tracking. This enhancement is based on providing an additional 3-way handshake between the peripheral and the gateway. Unarguably, this enhancement changes both the protocol and the peripheral which is highly impractical as we argued before.

Another related field of research includes wearable and body-area networks. The work by Leonard [21] uses a honeypot to lure in adversaries that attempt to attack the user's wearable devices. The honeypot uses a set of helper nodes to expose fake services with known weaknesses so that the attacker connects to them. This work, however, doesn't handle the privacy threat arising from BLE advertisements. A determined attacker will be able to distinguish fake traffic from legitimate one based on RF signatures from the devices and issue connections to the user's real devices.

Other relevant work includes approaches to protecting medical devices. Mare *et al.* [25] propose a mechanism that protects health sensors when communicating with a gateway. The proposed system, albeit relevant, doesn't apply for the BLE ecosystem. It also mandates changing the medical devices. Gollakota *et al.* [12] propose an external device, called *Shield*, that the user wears to control access to his/her embedded medical device. Shield implements friendly jamming so that only an authorized programmer can communicate with the medical device.

BLE-Guardian takes an entirely different approach by targeting the control plane of the BLE protocol instead of the data plane. BLE-Guardian does not need to continually protect an ongoing authorized connection and more importantly need not invoke jamming signal cancellation that requires accurate estimation of channel condition in a dynamic mobile indoor environment as well as a full duplex radio. BLE-Guardian constitutes a reference design that can function with any radio that has reception and transmission capabilities on the 2.4 GHz band. BLE-Guardian, also, considers far less restrictive scenarios than Shield. It does not have to be within centimeters of the device-to-be-protected as the case with Shield. Moreover, BLE-Guardian's practical design allows scaling up protection for multiple devices (multiple connectors and protected devices) simultaneously, which is not the case for Shield that considers a two-device scenario only [36].

Finally, researchers have explored ways to reduce information leaks from sensors in a smart home environment [30, 37]. Srinivasan *et al.* [37], Park *et al.* [30], and Schurgot *et al.* [35] propose a set of privacy enhancements that include perturbing the timing of broadcasted sensory data along with padding the real sensory data with fake data to confuse the adversary. These protocols fail to address the threats resulting from BLE advertisements and have the shortcoming of requiring changes to the sensors as well.

3 BLE Primer

The BLE (Bluetooth 4.0 and newer) protocol has been developed by the Bluetooth SIG to support low power devices such as sensors, fitness trackers, health monitors, etc. Currently, more than 75,000 devices in the market support this protocol along with most of more capable devices such as smartphones, tablet, PCs, and recently access points [2].

3.1 BLE States

A BLE device assumes either a central or peripheral role. A peripheral device is typically the one with lower capabilities and with the information to advertise. The central device, typically an AP, PC, or smartphone, scans for advertisements and initiates connections.

The BLE specification places a higher burden on the central device. It is responsible for initiating the connection and thus has to keep scanning until it receives an advertisement. Conversely, the peripheral (prior to its connection) sleeps for most of the time and only wakes up to advertise, which helps save its limited energy.

3.2 Advertisements

BLE advertisements are instrumental to the operation of the protocol, and constitute the only means by which a device can be discovered. The specification defines 4 advertisement message types as shown in Table 1, and 3

Table 1: The four types of BLE advertisements.

Type	Advertising Interval
ADV_IND	20ms − 10.24s
ADV_DIRECT_IND	3.75ms
ADV_NONCONN_IND	100ms − 10.24s
ADV_SCAN_IND	100ms − 10.24s

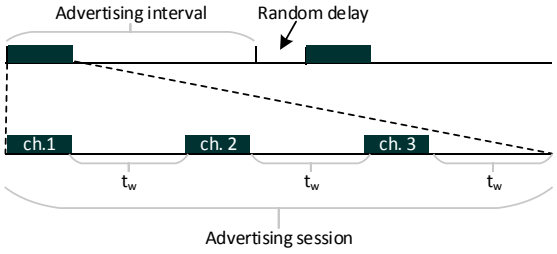


Figure 1: The advertisement pattern in BLE.

advertisement channels: 37 (2402MHz), 38 (2426MHz), and 39 (2480MHz).

ADV_DIRECT_IND (introduced in Bluetooth 4.2) is a special advertisement; it enables fast reconnection between the central and the peripheral devices. The peripheral, when turned on, will broadcast advertisements at a fast rate (once every 3.75ms, for 1.28 seconds) that are directed to the client (with a pre-existing trust relationship) before assuming the central role. The advertisement message only contains the message type, source, and destination addresses.

The other three advertisements are similar to each other in that they are periodic. The advertisement interval determines the frequency with which each device advertises. This interval has to be chosen, at configuration time, between 20ms and 10.24 seconds (at increments of 0.625ms) for the ADV_IND advertisement and between 300ms and 10.24 seconds for the other two advertisements. To prevent advertisements of different devices from colliding with each other, each device waits for a random amount of time between 0 and 10ms (in addition to the advertisement interval) before it advertises (Fig. 1).

The advertisement session constitutes the period when the device is actually advertising. During each advertisement session, the device advertises on the three advertisement channels given a pre-configured channel sequence. Before switching to the next channel, the device has to wait for a preset period of time (less than 10ms) for scan and connection requests (t_w in Fig. 1). We will henceforth refer to the advertisement interval, the channel sequence, and the waiting time as the *advertisement pattern*.

Each advertisement message contains the message type, source address, along with some of the services offered by the device and their respective values. The specification defines a set of services that have unique UUIDs, such as device name. The message is limited in length, and hence, to get more information about the device, an interested device can either issue a scan request to which the advertising device responds with a scan response or connect to the advertising device.

3.3 Connections

Not all BLE devices accept connections; devices that use ADV_NONCONN_IND advertisement messages run in transmit mode only and they don't accept any scan or connection requests such as iBeacons.

Also, devices advertising with ADV_SCAN_IND messages don't accept connections but accept scan requests. Particularly, when the device broadcasts an advertisement message on some channel, it listens on the same channel for some time (less than 10ms) before switching to the next channel. It waits for scan requests from clients wanting to learn more information to which it responds with a scan response.

Devices that advertise using ADV_IND messages are scannable and connectable; they respond to scan messages and connection requests. After sending an advertisement message, the device listens for connection requests. The connection request contains the source and destination addresses along with other connection parameters. These parameters contain the connection interval, the timeout, and the slave interval. When connected, the device starts frequency hopping according to a schedule negotiated with the central. If the device (now peripheral) doesn't receive any communication from the central over the period defined by the "timeout interval", it drops the connection.

While connected, the device must not broadcast connectable advertisement messages (the first two types of Table 1). It can, however, still broadcast non-connectable advertising messages to share information (the last two types of Table 1) with other clients in its transmission range which still leaks information about the device's name, type, and address.

3.4 Privacy and Security Provisions

The BLE specification borrows some security provisions from classical Bluetooth to establish trust relationships between devices, a process known as *pairing*. When the device boots for the first time, it will advertise using ADV_IND with its public Bluetooth address. The user can then pair a smartphone (or other BLE-equipped device) so that the two devices exchange a secret key that will enable future secure communication.

Once a BLE-equipped device is paired with another device, it can invoke more privacy and security provisions. The first provision is whitelisting, and the device will only accept connections from devices it has been paired with before, i.e., those that are whitelisted. Also, the device might accept connections from any client but might require higher security levels for some of the services it exposes so that only authorized users access sensitive content.

Finally, the BLE specification defines a privacy provision based on address randomization to prevent device tracking. When two devices are paired, they exchange an additional key called the *Identity Resolution Key* (IRK). The device uses this key to generate a random address according to a timer value set by the manufacturer, which it will use to advertise. This random address will be resolved by the paired device using the same key. As this random address is supposed to change regularly, curious parties shouldn't be able to track a BLE-equipped device. Devices that don't utilize address randomization can resort to direct advertising (ADV_DIRECT_IND) to enable fast and private reconnections.

These privacy provisions are akin to those proposed earlier in the context of WiFi networks. Researchers have long identified privacy leaks from the consistent identifiers broadcasted by wireless devices. They proposed privacy enhancements that include randomizing or frequent disposing of MAC addresses [14, 18] and hiding them through encrypting the entire WiFi packets [13]. These enhancements require introducing changes for the client devices.

4 Threats from BLE Devices

While, in theory, BLE's privacy provisions might be effective to thwart threats to the user's privacy, whether or not various manufacturers and developers properly implement them is an entirely different story. In what follows, we investigate how the BLE privacy provisions fare in the wild using a dataset collected in our institution and using the PhoneLab testbed [27] of SUNY Buffalo.

We developed an Android app that collects the content of the advertisement messages. We recruited users from our institution and from the PhoneLab testbed. We didn't collect any personal information about the users and thus obtained non-regulated status from the IRB of our institution. One could view this study as crowdsourcing the analysis of BLE devices; instead of purchasing a limited set of devices and analyzing them, we monitored the behavior of a broad range of devices in the wild. Analyzing the advertisements we collected from 214 different types of devices (sample of these devices shown in Tables 2 and 3), we observed:

Table 2: A sample of devices with revealing names.

Name	Type
LG LAS751M(27:5D)	music streaming
JS00002074	digital pen
ihere	key finder
spacestation	battery/storage extension
Jabra PULSE Smart	smartbulb
DEXCOMRX	Glucose monitor
Clover Printer 0467	printer
Frances's Band ea:9d LE	smartband
Gear Fit (60ED)	activity tracker
Lyve Home-00228	photo storage
Matthias-FUSE	headset
Richelle's Band b2:6a LE	smartband
vivosmart #3891203273	activity tracker
KFDNX	key fob
OTbeat	heart rate monitor
Thermos-4653	Smart Thermos
POWERDRIVER-L10C3	smart power inverter

Table 3: A sample of devices with consistent addresses for more than a day.

Name	Type	Days observed
One	activity tracker	37
Flex	activity tracker	37
Zip	activity tracker	37
Surge	activity tracker	36
Charge	activity tracker	36
Forerunner 920	smartwatch	36
Basis Peak	sleep tracker	25
MB Chronowing	smartwatch	15
dotti	pixel light	7
UP MOVE	fitness tracker	2
GKChain	laptop security	2
Gear S2 (0412)	smartwatch	2
Crazyflie	quadropter	1
Dropcam	camera	1

1. Two advertisement types (ADV_NONCONN_IND and ADV_SCAN_IND) require a fixed address which would enable tracking of a mobile target.
2. Devices that are bonded to the users advertise using ADV_IND messages instead of the more private ADV_DIRECT_IND.
3. Some devices, albeit not expected to do so, use their public Bluetooth addresses in advertisements. This also enables tracking as well as identifying of the device manufacturer.
4. Other devices implement poor address randomization by flipping some bits of the address rendering them ineffective against tracking. This has also been identified in other studies of BLE devices [22].
5. A large number of devices advertise their names (Table 2), revealing sensitive information about them, the user, and the environment. Also, some of the device names contain personal information

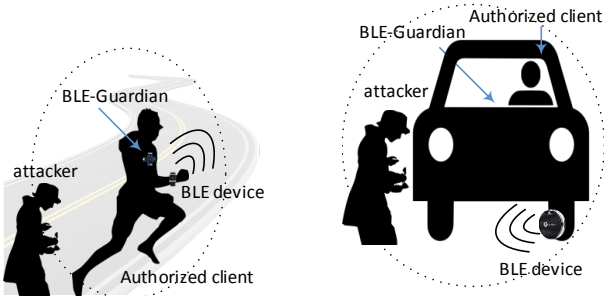
or unique identifiers that may enable user tracking.

6. Some devices use a consistent Bluetooth address for long periods of time which renders address randomization ineffective (Table 3). Examples include various types of wristbands (Fitbit Flex, Forerunner 920, etc.), headsets, smartwatches (Apple Watch or Samsung Gear), etc. This has also been identified by Das *et al.* [7], where they analyzed the advertisements of BLE-equipped fitness trackers. Das *et al.* found the fitness trackers constantly advertising with consistent (non-private) BLE addresses. In our experiments, we observed that Flex and One kept the same address for 37 days, so did One and Charge for 30 days.

7. Some devices accept connections from non-bonded devices. This allows access to the services on the device including the unique manufacturer ID, for instance, which allows for user tracking regardless of the device's address. For example, we were able to connect to various devices and access data from them without any existing trust relationship, such as various Fitbit devices (One, Zip Flex, Charge), Garmin Vivosmart, digital pens, etc. It is worth noting that we connected to these devices under controlled experimental settings, not in the wild. As a result, an external access control mechanism is necessary to protect such devices.

The above observations indicate that the address randomization, part of the BLE specifications, fails to provide the promised privacy protection. Various developers and manufacturers do not implement it properly; they rely on public Bluetooth addresses, apply weak randomization, or keep a consistent address for a long time. On the other hand, even if address randomization is properly implemented, other information in the advertisement or in the device might contain unique information (device name or id) that allows for its tracking.

Moreover, data accessed from an advertisement or the device (once connected) might reveal sensitive information about the user or the environment. Through our data collection campaign, we were able to detect different types of glucose monitors, wristbands, smart watches, fitness trackers, sleep monitors, laptops, smartphones, laptop security locks, security cameras, key trackers, headsets, etc. Knowing which type of glucose monitor the user is carrying or the type of physical security system s/he has installed could lead to serious harm to the user. Finally, an adversary might use such advertisement data as side information to infer more about the user's behavior. For example, a temperature sensor constantly reading 60 °F in winter would indicate a vacant property [41] which may invite in a thief.



(a) Mobile (b) Vehicle

Figure 2: Example deployments of `BLE-Guardian`.

5 BLE-Guardian

`BLE-Guardian` addresses the aforementioned privacy threats by allowing only *authorized clients* to discover, scan, and connect to the user's BLE-equipped device. Before delving into the inner workings of `BLE-Guardian`, we first describe the system and threat models.

5.1 System and Threat Models

5.1.1 System Model

A typical BLE scenario involves two interacting entities: the client and the BLE-equipped device. The BLE device broadcasts advertisements to make other nearby clients aware of its presence along with the services/information it is offering. A client can then connect to the device to access more services/information and control some of its attributes, in which case it will be the BLE-device's gateway to the outside world.

The user's mobile device (e.g., smartphone or tablet) acts a gateway where BLE devices are wearable (e.g., fitness trackers), or mHealth devices (e.g., Glucose monitor) (Fig. 2a). In a home environment, the user's access point, PC, or even mobile device, serves as a gateway for BLE devices that include smart appliances, webcams, physical security systems, etc. Last but not least, a smart vehicle or the driver's mobile device operate as gateways (Fig. 2b) for the different BLE-equipped sensors in the vehicle, such as tire pressure.[2] An interested reader could consult the work of Rouf *et al.* [32] for a discussion on the security and privacy risks of a wireless tire pressure sensor.

`BLE-Guardian` leverages the existence of gateways near the BLE-equipped devices to fend off unauthorized clients scanning and connecting to them. It comprises both hardware and software components. The hardware

[2] https://my-fobo.com/Product/FOBOTIRE

component is an external Bluetooth radio that connects physically to the gateway, while the software component is an accompanying application that runs on the gateway. BLE-Guardian requires another software component to run on the clients willing to discover and connect to the user's BLE devices. The user, be it an owner of the BLE-equipped device or a client, interacts with BLE-Guardian through its software components.

5.1.2 Threat Model

BLE-Guardian only trusts the gateway on which it is running. Otherwise, the entire operation of BLE-Guardian will be compromised and will fail to provide the promised privacy provisions. BLE-Guardian achieves privacy protection at the device level, so that if it authorizes a client to access the BLE device, all applications running on that device will have same access privileges. This security/privacy dimension is orthogonal to BLE-Guardian and has been addressed elsewhere [28]. It also requires the user's BLE device — the one to be protected — to comply with the BLE specifications.

BLE-Guardian protects a target BLE-equipped device against an adversary or an unauthorized/unwanted device that sniffs the device's advertisements, issues scan requests and attempts to connect to the device. The adversary aims to achieve three objectives based on the BLE devices that the user is deploying:

1. **Profiling:** The adversary aims to obtain an inventory of the user's devices. Based on this inventory, the adversary might learn the user's health condition, preferences, habits, etc.
2. **Tracking:** The adversary aims to monitor the user's devices to track him/her over time, especially by exploiting the consistent identifiers that the devices leak as we observed in Section 4.
3. **Harming:** The adversary aims to access the user's BLE device to learn more sensitive information or even to control it. Both will have severe consequences for the user, especially if a certain device is known to have some vulnerabilities that allow remote unauthorized access [26].

This adversary can have varying passive and active capabilities, from curious individuals scanning nearby devices (e.g., using a mobile app), to those with moderate technical knowledge employing commercial sniffers, all the way to sophisticated adversaries with software-defined radios.

A *passive* attacker is capable of sniffing all the communications over advertisement channels including those that fail the CRC check. This includes all commercial Bluetooth devices and existing Bluetooth sniffers in the market, such as the Texas Instruments CC2540 chip. The adversary might possess further capabilities by employing MIMO receiver that could recover the original signal from the jammed signal [38], especially in stationary scenarios. We refer to this adversary as the *strong passive* attacker.

Furthermore, the adversary is capable of injecting traffic into any Bluetooth channel at any given point of time, but will "play" within the bounds of the BLE specifications when attempting communication with the BLE device. This is a reasonable assumption, as the device won't otherwise respond to any communication. We refer to such an adversary as the *active* attacker. On the other hand, the attacker might be able to transmit with higher power than allowed by regulatory bodies, in which case we refer to as the *strong active* attacker.

Thus, we have four classes of attackers referring to the combinations of their passive and active capabilities as shown in the first column of Table 4.

Attacks, including jamming the channel completely, masquerading as fake devices to trick the users to connect to them, or attacking the bonding process are orthogonal to our work. Such attacks have been addressed by O'Connor and Reeves [29] and Ryan [33]. Finally, once BLE-Guardian enables the authorized client to connect to the BLE device, it won't have any control over what follows later.

5.2 High-Level Overview

BLE-Guardian is a system the user can use out of the box; it only requires installing a hardware component (an external Bluetooth radio) to the gateway and running an app on the gateway to control and interface with the Bluetooth radio. Conceptually, BLE-Guardian consists of *device hiding* and *access control* modules. The device hiding module ensures that the BLE device is invisible to scanners in the area, while the access control module ensures that only authorized clients are allowed to discover, scan, and connect to the BLE device.

Fig. 3 shows the high-level operation of BLE-Guardian from the moment a user designates a BLE device to be protected all the way to enabling authorized client connection to the protected device. The high-level operation of BLE-Guardian takes the following sequence:

1. The user installs the hardware component along with the accompanying app on the gateway.
2. The user runs the app, which scans for BLE devices nearby. The user can then choose a device to hide.
3. The device hiding module of BLE-Guardian starts by learning the advertisement pattern of the target BLE device along with that of the other devices in

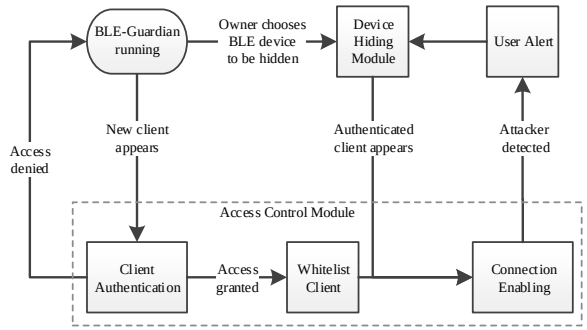


Figure 3: The modules of BLE-Guardian and their underlying interactions.

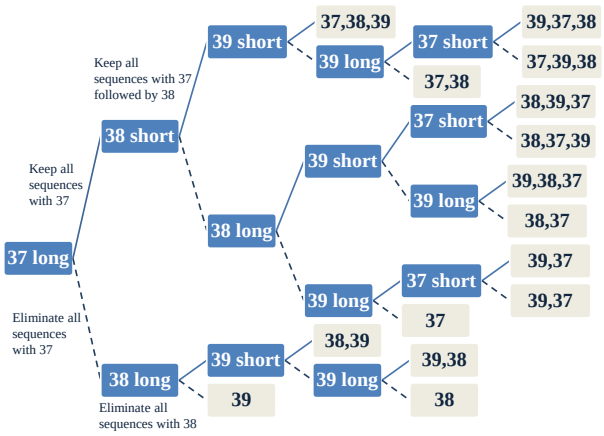


Figure 4: The learning algorithm followed by BLE-Guardian. The blue boxes refer to monitoring each channel either for a short period of time (less than 10ms) or for a longer period of 10.24 seconds. Depending on whether an advertisement is detected on the channel some sequences are eliminated till a sequence is decided on (gray boxes).

the area. The device hiding module then applies reactive jamming to hide the device.

4. When a new client enters the area and wants to discover the user's devices, it communicates with the access control module so that the user can either grant or reject authorization.
5. If the user authorizes the client, the access control module advertises *privately* on behalf of the BLE device to let the authorized client scan and connect to it.
6. BLE-Guardian monitors if other unauthorized entities are attempting to connect to the BLE device; in such a case, it blocks the connection and alerts the user.

5.3 Device Hiding

The hiding module is responsible for rendering the BLE device invisible to other scanning devices. The hiding module jams the device's advertisement session to achieve this invisibility. In particular, it targets three types of advertisements, ADV_IND, ADV_NONCONN_IND, and ADV_SCAN_IND of Table 1, which are periodic and leak more information about the user as we indicated earlier.

Hiding the BLE device is, however, challenging for two reasons. The hiding module must jam the BLE device precisely at the moment it is advertising. Also, it must not disrupt the operation of other devices advertising in the same area.

5.3.1 Learning

The hiding module first learns the target BLE device's advertising pattern before jamming to hide its presence. The device's advertisement pattern comprises the advertising interval, advertising channel sequence, and the time to listen on the individual channels. Fortunately, the latter two parameters are deterministic and can be observed directly, which is not the case for the advertising interval. The BLE specification leaves it to the device to determine the advertising pattern, so that there are 15 possible permutations of the channel sequence.

As shown in Fig. 4, BLE-Guardian follows a process of elimination to identify the advertising sequence of the BLE device using a single antenna. In the worst case, it will take three advertising intervals to learn the entire advertising sequence of a BLE-equipped device. This corresponds to the longest path of Fig. 4, where BLE-Guardian monitors each channel for the maximum advertising interval of 10.24 seconds. At the same time, it would have identified the time the BLE device spends listening on each channel before switching to the next channel.

While observing the advertising sequence of the BLE device, the hiding module keeps track of the interval separating the consecutive advertisements sessions. The hiding module observes a set of inter-advertisement intervals, $t_i = adv + p$, where adv is the actual advertisement interval as set by the device and p is a random variable representing the random delay such that $p \in unif(0, 10ms)$. Also, BLE-Guardian will perform the same process for all advertising devices in the same area at the same time to learn their advertising parameters as well. Learning other devices' advertising at the same time will be useful as evident below.

5.3.2 Actuation

After identifying the advertising pattern, the hiding module needs to just detect the start of the advertisement session. Then, it jams the advertising channels according to their advertisement sequence. There is a caveat, though; the hiding module needs to detect the advertisement before it can be decoded. Otherwise, the rest of the jamming will not be effective.

From monitoring earlier advertisements, the hiding module obtains a set of t_i's of different devices' advertisements, including the BLE device to be hidden. The advertisement interval will be $adv = t_i - p$ for each observed inter-advertisement interval. Each observed advertisement will be used to improve the estimation of the advertisement interval. For N observed intervals, we have:

$$adv = \frac{1}{N}\sum_{i=1}^{N}(t_i - p) = \frac{1}{N}\sum_{i=1}^{N} t_i - \frac{1}{N}\sum_{i=1}^{N} p. \qquad (1)$$

Let $P = \frac{1}{N}\sum_{i=1}^{N} p$, the random variable P is drawn from the distribution $\frac{1}{N}p * \frac{1}{N}p * \frac{1}{N}p \ldots \frac{1}{N}p$. Since the single random delays p are i.i.d., the mean of P will be equal to 5 (mean of the original distribution of p) and the standard deviation of $\sqrt{\sum_{i=1}^{N}\sigma_p} = \frac{5}{N\sqrt{(3)}}$. The hiding module estimates adv as:

$$adv' = E(adv) = \frac{1}{N}\sum_{i=1}^{N} t_i - 5. \qquad (2)$$

The standard deviation of P will get lower as N increases; it defines the error in the estimate of adv as defined by Eq. (1). Given previous N observed advertisements from the BLE device, the hiding module can predict the next advertisement to happen at $adv_{next} \in [adv_{low},\ adv_{high}]$ such that:

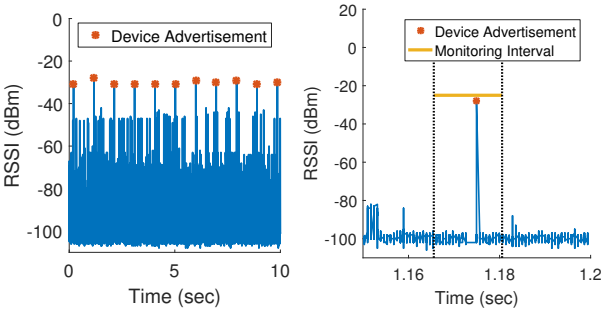
$$adv_{low} = T_N + adv' - e \qquad (3)$$

$$adv_{high} = T_N + adv' + e + 10, \qquad (4)$$

where T_N is the time of the last advertisement and e is the 90^{th} percentile value of P (symmetric around the mean) which approaches 0 as N increases (so that $adv_{high} - adv_{low}$ approaches $10ms$).

Starting from the last observed T_N of the target BLE device, the advertisement hiding module computes adv_{low} and adv_{high}. Also, it enumerates the list of other devices expected to advertise within the interval $[adv_{low},\ adv_{high}]$.

The device hiding module always listens on the first channel of the advertising sequence of the BLE device to be hidden. During the interval $[adv_{low},\ adv_{high}]$, the device hiding module will sample the RSSI of the channel very frequently (every 25μs). When the received RSSI is $-90dBm$ or higher (the peaks of Fig. 5a), `BLE-Guardian` determines that there is a transmission currently starting to take place. The device hiding module moves immediately to jam the channel on which it is listening. Since the transmission of a typical advertisement message takes 380μs to finish [16], jamming the channel will prevent the message from being decoded by other receivers.



(a) Over 10 seconds. (b) Single adv. interval.

Figure 5: RSSI at channel 37 when a device is advertising at a distance of 1m at the interval of 960ms.

At this point, two situations might arise; (1) the target BLE device is the only device expected to be advertising at this time instant, or (2) some other device is expected to be advertising in the same interval. In the first situation, the target BLE device is most probably responsible for this transmission as part of its advertisement session. The device hiding module repeats the same process (sample RSSI and jam) over the rest of the channels to confirm that transmissions follow the device's advertising pattern. Fig. 5b shows an example interval where there is only one device advertising.

In the second situation, the device hiding module can't readily ascertain whether the transmission belongs to the target BLE device or not. This will take place when the observed transmission sequence matches the advertising sequence of the target BLE device and some other device that is expected to advertise at the same interval. To resolve this uncertainty, immediately after jamming the advertising message (400μs after commencing jamming on the channel), the device hiding module lifts jamming and sends scan requests for devices other than the target device. The device hiding module then listens on the channel to observe if a scan response is received. Despite its advertisement being jammed, any device will still be listening on and will respond to scan requests. Depending on whether a scan response is received or not, `BLE-Guardian` can associate the transmission with the correct device, be it the target BLE device or some other

device.

The device hiding module then adjusts the next monitoring interval according to the observed transmissions in the current intervals as follows:

$$adv_{low} = min(T_N) + adv' - e \quad (5)$$

$$adv_{high} = max(t_N) + adv' + e + 10, \quad (6)$$

where T_N represents the instants of the transmissions possibly matching the advertisement of the target BLE device in the current monitoring interval.

Note that we don't utilize the power level per se, or any physical-layer indicator, to indicate whether the same device is transmitting or not, as it is sensitive to the environment and the distance between `BLE-Guardian` and the target BLE device. To actually perform the jamming, the device hiding module continuously transmits at the maximum power for the specified interval.

`BLE-Guardian` may jam the advertisements of non-target devices which might disrupt their operation, which we referred to as the second situation above. Nevertheless, because of the random delay introduced by the device before each advertisement, the aforementioned "collision" events become unlikely. In Appendix A, we use renewal theory to show that the expected number of another device's advertisements within the expected advertising interval of the target BLE-equipped device will always be less than 1. This applies when `BLE-Guardian` protects a single BLE-equipped device. Our evaluation in Section 6 confirms this observation.

5.4 Access control

So far, `BLE-Guardian` has hidden the target BLE device, so neither authorized nor unauthorized entities have access to the device. It is the access control module that authorizes client devices and enables their access to the target BLE device.

5.4.1 Device Authorization

`BLE-Guardian` utilizes Bluetooth classic (BR/EDR) as an out-of-band (OOB) channel to authorize end-user devices intending to scan and access the target BLE device. `BLE-Guardian` runs in server mode on the gateway waiting for incoming connections, while the "authenticating" device will have `BLE-Guardian` running in client mode to initiate connections and ask for authorization. The choice of Bluetooth Classic as an OOB channel is natural; most end-user devices (such as smartphones) are dual-mode, supporting both BLE and Bluetooth classic. Moreover, Bluetooth classic already contains pairing and authentication procedures, eliminating the need for a dedicated authentication protocol. Last but not least, a

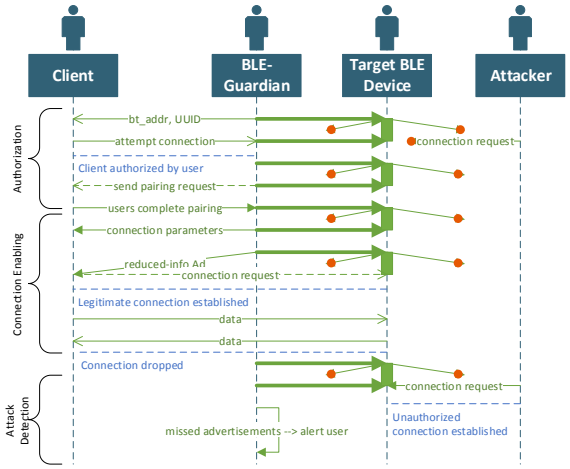


Figure 6: The sequence diagram of the access control module. Thin green lines from the target device designate the advertisements. Thick green lines from `BLE-Guardian` designate the jamming signal.

Bluetooth-equipped end-user device will be able to communicate simultaneously over Bluetooth classic and BLE so that it can communicate with both `BLE-Guardian` and the target BLE device.

Fig. 6 depicts the interactions between `BLE-Guardian` and a client device when they connect for the first time. `BLE-Guardian` will be listening on the gateway over a secure RFCOMM channel with a specified UUID. The gateway, however, won't be running in discoverable mode so as to prevent others from tracking the user. It is up to the party interested in authenticating itself to obtain the Bluetooth address of the user's gateway as well as the UUID of the authentication service.

Once the client end-user device obtains the Bluetooth address and UUID, it can initiate a secure connection to the gateway. This will trigger a pairing process to take place if both devices are not already paired. `BLE-Guardian` relies on Bluetooth pairing process to secure the connections between the gateway and the client device. For future sessions, an already paired client device can connect to `BLE-Guardian` without the need for any user involvement. The owner can also revoke the privileges of any client device by simply un-pairing it.

5.4.2 Connection Enabling

The device hiding module of `BLE-Guardian` jams the entire advertising sequence of the target BLE device, including the period it listens for incoming scan or connection requests so that it cannot decode them. Therefore, both unauthorized and authorized clients cannot access the target BLE device (the case of an adversary using

high transmission power will be discussed later). Fig. 6 shows the procedure that BLE-Guardian follows to enable *only* the authorized clients access to the target BLE device.

Immediately after the last advertisement of a single advertisement session, when the target device is the only one expected to be advertising, the access control module lifts the jamming. This ensures that the BLE device will not be advertising until the next adverting session, and it is currently listening for scan and connection requests. Then, BLE-Guardian advertises on behalf of the target BLE device on the same channel. The advertisement message contains only the headers and the address of the previously hidden device. It is stripped of explicit identifiers, hence leaking only limited information about the BLE device for a brief period.

At the same time, BLE-Guardian will communicate to the authenticated client app the address of the BLE device and a *secret* set of connection parameters over the OOB channel. BLE-Guardian's app running on the client device will use the address and the parameters to initiate a connection to the BLE device. The connection initiation procedure is handled by the Bluetooth radio of the client device, which scans for the advertisement with the provided address. After receiving such an advertisement, it sends a connection request after which both devices will be connected.

The above procedure will not break the way BLE scans and connections take place. It doesn't matter from which radio the actual advertisement was coming. From the perspective of the BLE device, it will receive a scan or connection request while waiting for one. On the other hand, the client device will receive an advertisement message while also expecting one.

5.5 Security and Privacy Features

BLE-Guardian addresses the tracking and profiling threats discussed in Section 5.1.2. It hides the advertisements, which are used as the main means to track users. It only exposes the advertisement for a very short period when enabling others to connect. Furthermore, BLE-Guardian greatly reduces the profiling threat by hiding the contents of the advertisement which leak the device name, type, and other attributes.

A strong passive attacker [38] can still detect the "hidden" peripheral by recovering the real advertisement, so that it can connect to the BLE-equipped device. Distinguishing legitimate connection requests based on the Bluetooth address of the initiator is not effective; an attacker could easily spoof its Bluetooth address to impersonate the authorized client. Therefore, BLE-Guardian uses the connection parameters of the connection request to distinguish fraudulent connection requests from legitimate ones. Legitimate connection requests contain the set of "secret" connection parameters communicated earlier to the client.

Table 4: The protections offered by BLE-Guardian.

Adversary	Profiling Protect.	Tracking Protect.	Access Control	User Alert
Passive & Active	✔	✔	✔	✔
Strong Passive & Active	–	✔	✔	✔
Passive & Strong Active	✔	✔	–	✔
Strong Passive & Strong Active	–	✔	–	✔

The probability of the attacker matching a particular set of connection parameters is very low. According to the specification, there are more than 3 million possible combinations of values for the connection, slave, and timeout intervals. If the connection is established based on a fraudulent connection request, then BLE-Guardian prevents the connection from taking place. The connection request already contains the hopping sequence initiation. BLE-Guardian hops to the next channel and jams it so as to prevent the BLE device from receiving any message from the connected unauthorized device. The BLE device drops the connection since it receives no message on the channel.

An attacker might abuse this connection process by constantly attempting to connect to the BLE device, thus depriving the authorized client of access. This will always be possible, even when BLE-Guardian is not deployed. BLE-Guardian observes such a situation from a high frequency of fraudulent connection requests and alerts the user of this threat. As it will be evident in Section 6, an active attacker injecting messages to the advertising channel can't affect the operations of BLE-Guardian.

A strong active adversary, however, can override BLE-Guardian's jamming and issue connection requests that the BLE-equipped device will decode. While jamming, BLE-Guardian runs in transmit mode and can not monitor the channel for incoming requests. Nevertheless, it detects that the BLE device is missing its advertising intervals, signifying that it was connected without BLE-Guardian's approval. In such a case, BLE-Guardian alerts the user of the existence of a strong adversary nearby.

Finally, Table 4 summarizes BLE-Guardian's capabilities when faced with the various adversaries described in Section 5.1.2.

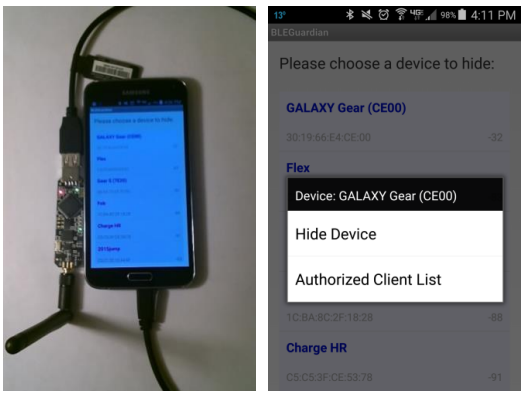


Figure 7: The deployment scenario for BLE-Guardian for a mobile user (left) and the main UI (right).

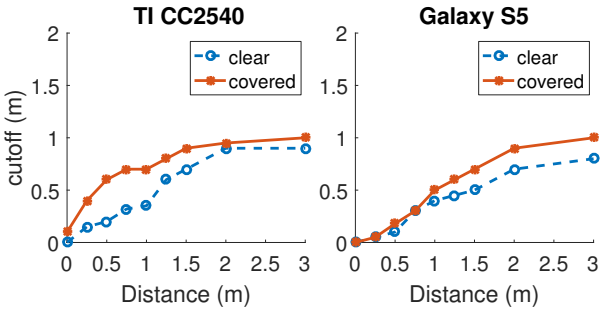


Figure 8: The cutoff distance as a function of the distance between BLE-Guardian and the target device.

6 Implementation and Evaluation

We now present a prototype of BLE-Guardian along with its evaluation.

6.1 Implementation

We implement BLE-Guardian using Ubertooth One radio which is an open platform for Bluetooth research and development. It can connect to any host that supports USB such as Raspberry Pi, Samsung's Artik-10, PC, smartphone (Fig. 7 (left)), etc. Since communication over USB incurs latency in the order of a few milliseconds, we implement most of BLE-Guardian's functionalities inside Ubertooth One's firmware to maintain real-time operation.

We also implement the software component of BLE-Guardian on Linux and Android. Fig. 7 (right) shows a screenshot of the BLE-Guardian app while running on Android in server mode where the user can choose the device to protect and control its authorized client list. The app communicates the Bluetooth address of the chosen device to the Ubertooth One radio.

BLE-Guardian requires running in privileged mode on the client device in order to be able to connect with modified connection parameters. This is easily achievable on Linux-based clients, but might not be the case for mobile devices. In other words, BLE-Guardian, while running in client mode on Android, requires root access to be able to issue connection requests with a set of specified connection requests. Also, BLE-Guardian (if running in privileged mode on the client device) can modify content of the advertisement message from the BLE scanner to the user-level application to reconstruct the original hidden advertisement. As such, user-level applications (on the trusted client) will receive the original advertisement, which does not break their functionality.

Maintaining BLE-Guardian is easy; it only requires updating the application running on the gateway which usually takes place without the user's intervention (e.g., mobile app updates). This application interacts with the hardware component and applies updates, if necessary, through pushing firmware updates, a process which is also transparent to the users.

6.2 Evaluation

To evaluate BLE-Guardian, we utilize Broadcom BCM20702A0 and Nordic nRF51822 chips as the target BLE devices (both transmitting at 4dBm) and the TI CC2540 dongle as the sniffer node. CC2540 is able to decode the messages on the three advertisement channels, even on those that fail the CRC check. We evaluate using Nordic and Broadcom chips instead of actual BLE products, because these products (such as Fitbits) are mostly powered by the same (Nordic and Broadcom) BLE chips.

6.2.1 Impact of Distance

Due to transmission power limitations (battery or regulatory bodies' constraints), there would always be a small area around the target BLE device where BLE-Guardian won't be able to enact the privacy protection. The transmission from the target BLE device covers the jamming signal of BLE-Guardian. Nevertheless, as the sniffer moves farther away from the target BLE device (in any direction), the jamming signal will cover the advertisements, provided that the BLE device and BLE-Guardian are not very far apart. So, there is a cutoff distance beyond which the adversary can't scan, and connect to the target BLE device.

We study the cutoff distance of a target BLE device (advertising at 20ms) at different distances separating it from BLE-Guardian (between 0 and 3m). At each position, we move the sniffer node (either a CC2540 dongle or Samsung Galaxy S5) around the BLE device, and

record the farthest distance at which it received any advertisement from the BLE device as to study the hidden terminal effect. Furthermore, we repeat each experiment twice, the first with BLE-Guardian clear of any obstacles and the second with it inside a backpack.

It is evident from Fig. 8 that the cutoff distance increases as BLE-Guardian and the BLE device become farther apart. In all of the cases, however, the cutoff distance is less than 1m, even when BLE-Guardian and the BLE device are 3m apart. This also applies when BLE-Guardian is inside the backpack which should reduce the effectiveness of its jamming. Sniffing with a smartphone has a shorter cutoff distance because the smartphone's BLE chip filters out advertisements failing the CRC check so that they are not reported to the OS.

The cutoff distance is enough to thwart tracking and profiling in several scenarios, especially when the user is moving (walking, jogging, biking or driving). In these scenarios, BLE-Guardian is not farther than 1m from the target BLE device. An adversary has to get very close to the user, even if BLE-Guardian is covered in a coat or bag, to be able to scan or connect to the BLE device.

In other cases, the user has to keep his BLE devices (to be protected) close to BLE-Guardian in order to get the best privacy protection possible. Our experiments showed that BLE-Guardian and the target BLE device must be separated by a maximum distance of 5m so that an attacker beyond the cutoff distance won't be able to decode the advertisements. If BLE-Guardian and target BLE device are farther apart than this, then BLE-Guardian's jamming won't be able to cover the entire transmission area of the BLE device. In all circumstances, however, BLE-Guardian detects unauthorized connections and alerts the user accordingly.

6.2.2 Evaluation Setup

Beyond the cutoff distance, BLE-Guardian is capable of hiding the advertisements and controlling access to any target BLE device regardless of its advertising frequency. This protection, however, comes at a cost. In what follows, we evaluate BLE-Guardian's impact on other innocuous devices, the advertising channel, and the gateway. In the evaluation scenarios, we deploy the target BLE devices at distance of 1.5m from BLE-Guardian, and the sniffer between BLE-Guardian and the BLE devices (at a distance of 0.5m from BLE-Guardian). We evaluate BLE-Guardian when protecting up to 10 target devices with the following advertising intervals: 10.24 sec (highest possible), 5 sec, 2.5 sec, 1.25 sec, 960ms, 625ms, 312.5ms, 100ms, 40ms, and 20ms (lowest possible). Note that evaluating with 10 target devices constitutes an extreme scenario; according to our dataset, the average user is bonded to less than 4 devices, which would indicate the number of target devices (i.e. those to be protected).

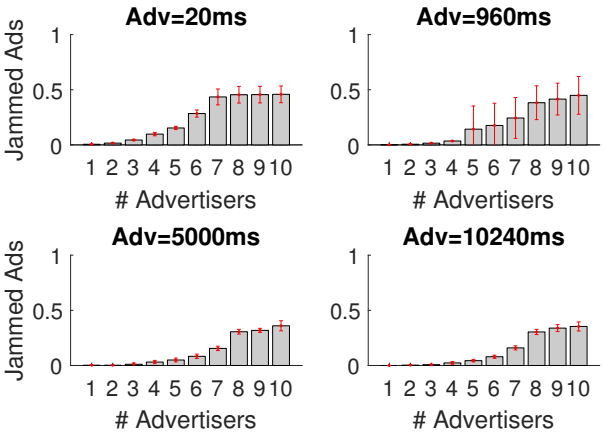


Figure 9: Portion of jammed advertisements of an innocuous BLE device when BLE-Guardian is running and protecting up to 10 advertisers.

6.2.3 Advertisement Hiding

Impact on Other Devices: We first evaluate the number of advertisements, not belonging to the target BLE device(s), BLE-Guardian will jam (Fig. 9). While accidentally jamming other devices doesn't affect the privacy properties of BLE-Guardian, it hinders the services they offer to other users. In particular, we study four scenarios with an innocuous (not the target) BLE device advertising at 20ms, 960ms, 5s, and 10.24s, and a varying number (between 1 and 10) of target devices, which BLE-Guardian protects. Each subset of target devices of size N (≤ 10) contains the top N advertising intervals from the list of Section 6.2.2.

There are two takeaways from Fig. 9. First, BLE-Guardian has little effect on other devices when it protects a relatively low number of devices, or when the advertising interval of the target BLE device(s) is larger than 500ms; in these cases, BLE-Guardian will be less active (bars corresponding to less than 6 target devices in the four plots of Fig. 9). Second, BLE-Guardian has a higher effect on the innocuous device with higher advertising frequencies as observed from top-left plot of Fig. 9, especially when protecting a large number of devices (including those with 20 ms advertising interval).

In the latter case, BLE-Guardian is active for at least half of the time, representing the worst-case scenario of BLE-Guardian's overhead where up to 50% of other devices' advertisements are jammed. However, since the advertisement frequency is high, even with a relatively high rate of jammed advertisements, the user's experience won't be drastically affected. On the other hand,

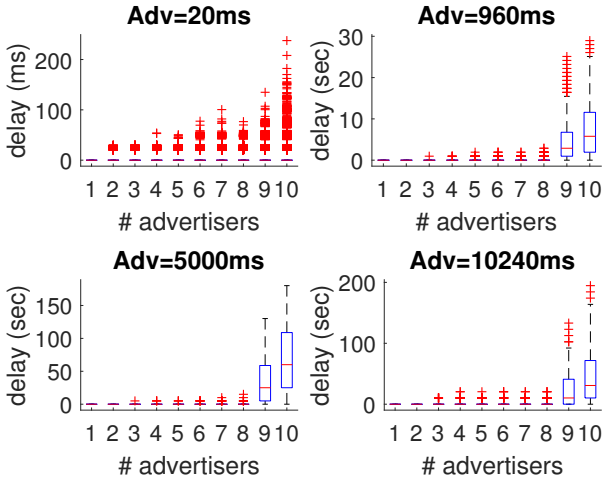


Figure 10: The delay of an authorized client in successfully connecting to the target device when BLE-Guardian is running.

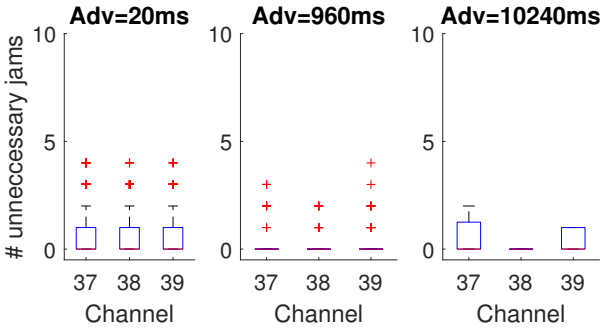


Figure 11: Unnecessary jamming instances with two advertisers at 20ms.

when the target BLE device advertises at lower frequencies, the effect on the advertising channels and consequently other devices will be limited as evident from the rest of the plots of Fig. 9.

Impact on Authorized Access: To enable authorized connections, BLE-Guardian advertises on the behalf of the target BLE device only when it is confident that the target device is listening for connections. BLE-Guardian skips some advertising sessions which will introduce delays to authorized clients attempting connections as reported in Fig. 10. In this scenario, an authorized client is attempting connection to a target device advertising at 20ms, 960 ms, 5s, and 10.24 s, with an additional number of protected devices varying from 1 to 10. In the majority of the cases, the client has to wait for less than a second before successfully receiving an advertisement and issuing a connection. The only exception is the worst case consisting of BLE-Guardian protecting all of the 10 target devices (including devices advertising at intervals less than 100ms). The client might have to wait for up to multiple advertisement intervals before being able to connect. This is evident from the rightmost bar in each of the four plots of Fig. 10.

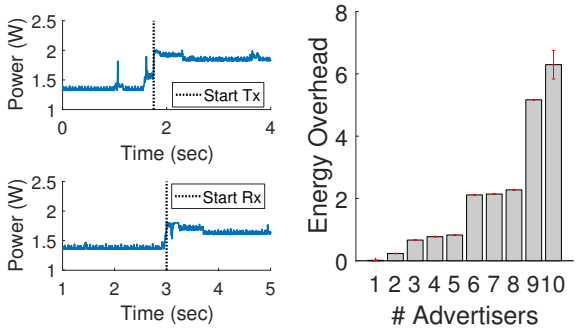
Impact on Advertising Channels Last but not least, we evaluate BLE-Guardian's impact on the advertising channel, which, if high, might leak information about the existence of sensitive device(s). In this experiment, BLE-Guardian protects a single target device advertising at 20ms (the lowest possible), 960ms, and 10240ms (the highest possible). At the same time, two innocuous devices advertise at 20ms, in addition to other 15 devices not under our control advertising at different frequencies (minimum advertisement interval 30ms). In this scenario, BLE-Guardian will be active all the time since the two innocuous advertisers will force it to enlarge its monitoring interval between 20–30ms (while the advertising interval of the target device is only 20ms).

Fig. 11 shows the distribution of the number of unnecessary jammed instances in each interval when the target BLE device is expected to advertise. It is evident that in more than 50% of the intervals when BLE-Guardian is active, the number of unnecessary jamming instances events is 0, indicating a low overhead on the channel. When the target BLE device advertises at a lower frequency, BLE-Guardian is less active (middle and left plots of Fig. 11). These plots match the real-world scenarios observed from our data-collection campaign. Most commercial BLE devices advertise at relatively low frequencies (at intervals between 1 and 10s).

Finally, we evaluate the accuracy of predicting the next advertisement event of the target BLE devices. In all the experiments (including all scenarios), BLE-Guardian can predict the device's advertisements, i.e., the target BLE device advertised in the interval it is expected to. BLE-Guardian is also able to jam all the advertisements of the BLE device over the three advertising channels. This indicates that an attacker can't modify the behavior of BLE-Guardian by injecting traffic into the advertising channels.

6.2.4 Energy Overhead

BLE-Guardian incurs no energy overhead for both the target BLE devices and the authorized clients. Nevertheless, energy overhead is a concern when BLE-Guardian is attached to a smartphone. We measured the energy overhead of BLE-Guardian using a Monsoon power monitor while running on a Samsung Galaxy S4 with Android 4.4.2. In the idle case, BLE-Guardian consumes 1370mW on average. The average power consumption rises to 1860mW while transmitting and 1654mW while receiving as shown in Fig. 12a. Fortunately, BLE-Guardian doesn't sense the channel or per-



(a) Overhead of basic operations.

(b) Average overhead in different scenarios.

Figure 12: The energy overhead of `BLE-Guardian` running on Samsung Galaxy S4.

form jamming frequently. Fig. 12b shows the average energy overhead when `BLE-Guardian` is protecting the set of ten devices (we described earlier) at different advertising intervals. In the worst case of 10 target BLE devices, including a couple advertising at the highest frequency possible, the energy overhead is limited to 16% regardless of whether there are other advertisers in the area. In other cases, when there are less target devices and/or target devices are advertising at a lower frequency, the energy overhead is negligible.

7 Conclusion

BLE is emerging as the most prominent and promising communication protocol between different IoT devices. It, however, accompanies a set of privacy risks. An adversary can track, profile, and even harm the user through BLE-equipped devices that constantly advertise their presence. Existing solutions are impractical as they require modifications to the BLE-equipped devices, thereby making their deployment difficult. In this paper, we presented a device-agnostic system, called `BLE-Guardian`, which addresses the users' privacy risks brought by BLE-equipped devices. `BLE-Guardian` doesn't require any modification to the protocol and can be implemented with off-the-shelf Bluetooth hardware. We implemented `BLE-Guardian` using Ubertooth One radio and Android, and evaluated its effectiveness in protecting the users' privacy. In future, we plan to explore the data plane by analyzing and reducing the data leaks from BLE devices to unauthorized clients.

8 Acknowledgments

We would like to thank the anonymous reviewers for constructive comments. We would also like to thank Krishna C. Garikipati for useful discussions on this paper. The work reported in this paper was supported in part by the NSF under grants CNS-1114837 and CNS-1505785, and the ARO under grant W911NF-15-1-0511.

References

[1] ARTICLE 29 DATA PROTECTION WORKING PARTY. Opinion 8/2014 on the on recent developments on the internet of things. http://ec.europa.eu/justice/data-protection/article-29/documentation/opinion-recommendation/files/2014/wp223_en.pdf, Sep. 2014. Accessed: 18-01-2016.

[2] ARUBA NETWORKS. Data Sheet: Aruba 320 series access points. http://www.arubanetworks.com/assets/ds/DS_AP320Series.pdf.

[3] BLUETOOTH SIG. Bluetooth SIG Analyst Digest 2H 2014. https://www.bluetooth.org/en-us/Documents/Analyst2014. Accessed: 10-02-2016.

[4] BLUETOOTH SIG. Specification of the Bluetooth System. Version 4.2, Dec. 2014. https://www.bluetooth.org/en-us/specification/adopted-specifications.

[5] COX, D. *Renewal theory*. Methuen's monographs on applied probability and statistics. Methuen, 1962.

[6] CRIST, R. Samsung swings for the fences with a new smart fridge at ces. http://www.cnet.com/products/samsung-family-hub-refrigerator, Jan. 2016. Accessed: 18-01-2016.

[7] DAS, A. K., PATHAK, P. H., CHUAH, C.-N., AND MOHAPATRA, P. Uncovering privacy leakage in ble network traffic of wearable fitness trackers. In *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications* (New York, NY, USA, 2016), HotMobile '16, ACM, pp. 99–104.

[8] DEGELER, A. Bluetooth low energy: Security issues and how to overcome them. https://stanfy.com/blog/bluetooth-low-energy-security-issues-and-how-to-overcome-them/, Jun. 2015. Accessed: 02-02-2016.

[9] DIGI-KEY TECHNICAL CONTENT. Cypress PSoC 4 BLE (Bluetooth Low Energy). http://www.digikey.com/en/articles/techzone/2015/dec/cypress-psoc-4-ble-bluetooth-low-energy, Dec. 2015. Accessed: 12-01-2016.

[10] FEDERAL BUREAU OF INVESTIGATION. Internet of Things Poses Oppotunities for Cyber Crime. https://www.ic3.gov/media/2015/150910.aspx, Sep. 2015. Accessed: 18-01-2016.

[11] FEDERAL TRADE COMMISSION. Internet of Things, Privacy & Security in a Connected World. https://www.ftc.gov/system/files/documents/reports/federal-trade-commission-staff-report-november-2013-workshop-entitled-internet-things-privacy/150127iotrpt.pdf, Jan. 2015.

[12] GOLLAKOTA, S., HASSANIEH, H., RANSFORD, B., KATABI, D., AND FU, K. They can hear your heartbeats: Non-invasive security for implantable medical devices. In *Proceedings of the ACM SIGCOMM 2011 Conference* (New York, NY, USA, 2011), SIGCOMM '11, ACM, pp. 2–13.

[13] GREENSTEIN, B., MCCOY, D., PANG, J., KOHNO, T., SESHAN, S., AND WETHERALL, D. Improving wireless privacy with an identifier-free link layer protocol. In *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services* (New York, NY, USA, 2008), MobiSys '08, ACM, pp. 40–53.

[14] GRUTESER, M., AND GRUNWALD, D. Enhancing location privacy in wireless lan through disposable interface identifiers: A quantitative analysis. *Mobile Networks and Applications 10*, 3 (2005), 315–325.

[15] HART, L. Telit Acquires Wireless Communications Assets to Boost Capabilities in Low-Power Internet of Things Market). http://www.businesswire.com/news/home/20160113005310/en/, Jan. 2016. Accessed: 01-02-2016.

[16] HEYDON, R. *Bluetooth low energy: the developer's handbook*. Prentice Hall, 2012.

[17] HILL, K. 'Baby Monitor Hack' Could Happen To 40,000 Other Foscam Users. http://www.forbes.com/sites/kashmirhill/2013/08/27/baby-monitor-hack-could-happen-to-40000-other-foscam-users, Aug. 2013. Accessed: 18-01-2016.

[18] JIANG, T., WANG, H. J., AND HU, Y.-C. Preserving location privacy in wireless lans. In *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services* (New York, NY, USA, 2007), MobiSys '07, ACM, pp. 246–257.

[19] JOHN PESCATORE. A SANS Analyst Survey: Securing the "Internet of Things" Survey. https://www.sans.org/reading-room/whitepapers/analyst/securing-internet-things-survey-34785, Jan. 2014. Accessed: 18-01-2016.

[20] KUCHINSKAS, S. Bluetooth's smart future in telematics. http://analysis.tu-auto.com/infotainment/bluetooths-smart-future-telematics, March 2013.

[21] LEONARD, A. *Wearable Honeypot*. PhD thesis, Worcester Polytechnic Institute, 2015.

[22] LESTER, S. The Emergence of Bluetooth Low Energy. http://www.contextis.com/resources/blog/emergence-bluetooth-low-energy/, May 2015.

[23] LUTHRA, G. Embedded controllers for the Internet of Things. http://www.edn.com/design/sensors/4440576/Embedded-controllers-for-the-Internet-of-Things/, Oct 2015.

[24] MADAAN, P. IoT for the smarter home. http://www.ecnmag.com/article/2015/05/iot-smarter-home, May. 2015. Accessed: 11-01-2016.

[25] MARE, S., SORBER, J., SHIN, M., CORNELIUS, C., AND KOTZ, D. Hide-n-sense: Preserving privacy efficiently in wireless mhealth. *Mobile Networks and Applications 19*, 3 (2014), 331–344.

[26] MARGARITELLI, S. Nike+ FuelBand SE BLE Protocol Reversed. http://www.evilsocket.net/2015/01/29/nike-fuelband-se-ble-protocol-reversed/, Jan 2015.

[27] NANDUGUDI, A., MAITI, A., KI, T., BULUT, F., DEMIRBAS, M., KOSAR, T., QIAO, C., KO, S. Y., AND CHALLEN, G. PhoneLab: A large programmable smartphone testbed. In *Proceedings of SENSEMINE '13* (New York, NY, USA, 2013), ACM, pp. 4:1–4:6.

[28] NAVEED, M., ZHOU, X., DEMETRIOU, S., WANG, X., AND GUNTER, C. A. Inside job: Understanding and mitigating the threat of external device mis-bonding on android. In *Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS)* (2014), pp. 23–26.

[29] OCONNOR, T., AND REEVES, D. Bluetooth network-based misuse detection. In *Computer Security Applications Conference, 2008. ACSAC 2008. Annual* (Dec 2008), pp. 377–391.

[30] PARK, H., BASARAN, C., PARK, T., AND SON, S. H. Energy-efficient privacy protection for smart home environments using behavioral semantics. *Sensors 14*, 9 (2014), 16235.

[31] PETERSON, A. Yes, terrorists could have hacked Dick Cheneys heart. https://www.washingtonpost.com/news/the-switch/wp/2013/10/21/yes-terrorists-could-have-hacked-dick-cheneys-heart/, Oct. 2013.

[32] ROUF, I., MILLER, R., MUSTAFA, H., TAYLOR, T., OH, S., XU, W., GRUTESER, M., TRAPPE, W., AND SESKAR, I. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In *Proceedings of the 19th USENIX Conference on Security* (Berkeley, CA, USA, 2010), USENIX Security'10, USENIX Association, pp. 21–21.

[33] RYAN, M. Bluetooth: With low energy comes low security. In *Proceedings of the 7th USENIX Conference on Offensive Technologies* (Berkeley, CA, USA, 2013), WOOT'13, USENIX Association, pp. 4–4.

[34] SCHNEIER, B. The internet of things is wildly insecure – and often unpatchable. http://www.wired.com/2014/01/theres-no-good-way-to-patch-the-internet-of-things-and-thats-a-huge-problem, Jan. 2014. Accessed: 18-01-2016.

[35] SCHURGOT, M., SHINBERG, D., AND GREENWALD, L. Experiments with security and privacy in IoT networks. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2015 IEEE 16th International Symposium on a* (June 2015), pp. 1–6.

[36] SHEN, W., NING, P., HE, X., AND DAI, H. Ally friendly jamming: How to jam your enemy and maintain your own wireless connectivity at the same time. In *Security and Privacy (SP), 2013 IEEE Symposium on* (May 2013), pp. 174–188.

[37] SRINIVASAN, V., STANKOVIC, J., AND WHITEHOUSE, K. Protecting your daily in-home activity information from a wireless snooping attack. In *Proceedings of the 10th International Conference on Ubiquitous Computing* (New York, NY, USA, 2008), UbiComp '08, ACM, pp. 202–211.

[38] TIPPENHAUER, N., MALISA, L., RANGANATHAN, A., AND CAPKUN, S. On limitations of friendly jamming for confidentiality. In *Security and Privacy (SP), 2013 IEEE Symposium on* (May 2013), pp. 160–173.

[39] TURK, V. The internet of things has a language problem. http://motherboard.vice.com/read/the-internet-of-things-has-a-language-problem, Jul. 2014. Accessed: 03-02-2016.

[40] WANG, P. Bluetooth low energy-privacy enhancement for advertisement.

[41] WANT, R., SCHILIT, B., AND JENSON, S. Enabling the internet of things. *Computer 48*, 1 (Jan 2015), 28–35.

[42] ZIEGELDORF, J. H., MORCHON, O. G., AND WEHRLE, K. Privacy in the internet of things: threats andchallenges. *Security and Communication Networks 7*, 12 (2014), 2728–2742.

A Analysis of Device Hiding

`BLE-Guardian` may jam the advertisements of non-target devices which might disrupt their operation, which we refer to as the second situation in Section 5.3.2. Nevertheless, because of the random delay introduced by the device before each advertisement, the aforementioned "collision" events become unlikely. In what follows, we show that the expected number of another device's advertisements within the expected advertising interval of the target BLE-equipped device will always be less than 1, when `BLE-Guardian` protects a single BLE-equipped device.

One could view the advertising process of a single BLE-equipped device as a renewal process [5], where each event corresponds to an advertising session. The inter-arrival times, X_i, are nothing but the inter-advertising intervals defined as i.i.d. random variables such that $X_i \sim unif(adv, adv+10)$. The n^{th} advertisement time $T_n = \sum_{i=1}^{n} X_i$ has the distribution defined by the n-fold convolution of distribution of X_i. As n increases, the probability distribution of the n-th advertisement spreads over a larger time interval defined as $A = [n.adv,\ n.(adv+10)]$.

The device hiding module attempts jamming at an interval of width 10ms, as specified before. If this jamming interval falls within the expected advertising interval of some other device, A, then the second situation of Section 5.3.2 might occur. Nevertheless, as n increases the length of interval A increases and thus the expected number of advertisements, from a single device within 10ms should be less than 1. We show below how the expected number of advertisements in a 10ms interval drops between $n = 1$ and $n = 2$. We consider $m(t)$, the expected number of events up to time t, defined as $F_X(t) + \int_0^t m(t-x) f_X(x)\, dx$, where $f_S(s)$ is the probability distribution of X_i which is equal to $unif(adv, adv+10)$ and $F_X(t)$ is the cumulative distribution function given as:

$$F_X(t) = \begin{cases} 0 & t < adv \\ \frac{t-adv}{10} & adv \le t \le adv+10 \\ 1 & t > adv. \end{cases} \tag{7}$$

During the first advertising interval, $t \in [adv, adv+10]$, the expected number of advertisements is $\frac{t-adv}{10} + \int_{adv}^{t} m(t-x)\, dx$ after substituting $F_X(t)$ and $f_X(x)$ with their corresponding expressions. After performing a substitution of variable of $y = t - x$, and since $m(t) = 0$ for $t < adv$, then $m(t) = \frac{t-adv}{10}$. So, if the expected advertising interval of the device hiding module overlaps with the first advertising interval of another advertising device, the expected number of events, $m(adv+10) - m(adv)$, will be 1, which is intuitive.

The second advertisement will take place at the interval $B = [2.adv,\ 2.(adv+10)]$, and we use a similar procedure to derive the expressions for $m(t)$ for $t \in [2.adv,\ 2.adv+10]$ and $t \in [2.adv,\ 2.(adv+10)]$. If the expected advertising interval of the device hiding module overlaps with interval, B, then the expected number of advertisements $m(t+10) - m(t)$ will drop to $\frac{1}{2}$. The same trend will follow for the subsequent advertising intervals; the expected number of another device's advertisements within the expected advertising of the target BLE device will always be less than 1. Our evaluation in Section 6. confirms this observation.

Finally, even if another device, with the same advertising parameters, starts advertising with the target BLE device at the same time, their advertising events will eventually diverge. After N advertisements from both devices, the distribution of $Ta_{N+1} - Tb_{N+1}$, the difference in time between the $N+1$ advertising instants of both devices will be a random variable with mean 0 but with $\sigma = 2.N.\frac{5}{\sqrt{3}}$. As N increases, the standard deviation increases, which in turn decreases the probability of both advertising events taking place within 10ms. The 10ms-advertising interval is the length of interval that the device hiding module expects the target BLE device to advertise.