BlueShield: Detecting Spoofing Attacks in Bluetooth Low Energy Networks

Jianliang Wu[1*], Yuhong Nan[1*], Vireshwar Kumar[1], Mathias Payer[2], and Dongyan Xu[1]

[1]Purdue University, [2] EPFL

{wu1220, nan1, viresh}@purdue.edu, mathias.payer@nebelwelt.net, dxu@cs.purdue.edu

Abstract

Many IoT devices are equipped with Bluetooth Low Energy (BLE) to support communication in an energy-efficient manner. Unfortunately, BLE is prone to spoofing attacks where an attacker can impersonate a benign BLE device and feed malicious data to its users. Defending against spoofing attacks is extremely difficult as security patches to mitigate them may not be adopted across vendors promptly; not to mention the millions of legacy BLE devices with limited I/O capabilities that do not support firmware updates.

As a first line of defense against spoofing attacks, we propose *BlueShield*, a legacy-friendly, non-intrusive monitoring system. BlueShield is motivated by the observation that all spoofing attacks result in anomalies in certain cyber-physical features of the advertising packets containing the BLE device's identity. BlueShield leverages these features to detect anomalous packets generated by an attacker. More importantly, the unique design of BlueShield makes it robust against an advanced attacker with the capability to mimic all features. BlueShield can be deployed on low-cost off-the-shelf platforms, and does not require any modification in the BLE device or its user. Our evaluation with nine common BLE devices deployed in a real-world office environment validates that BlueShield can effectively detect spoofing attacks at a very low false positive and false negative rate.

1 Introduction

An increasing number of IoT devices leverage Bluetooth Low Energy (BLE) to communicate in an energy-efficient manner. As one of the most popular protocols for IoT devices (e.g., smart locks, smart lights, or smart thermostats) [36], BLE will be empowering up to 5 billion devices by 2023 [4]. Given the popularity of BLE, there is an increasing concern about its security due to the discovery of threats that enable illegal device access, user fingerprinting, and inference of the device's sensitive information [13, 20, 26, 37]. While various approaches have been proposed to mitigate some threats [21, 33], most of them focus on the protection of the BLE device itself, less noticed here is the end-to-end communication between a *BLE device* and a *user device* in the network, where a spoofing attack is a major threat.

*The two authors contributed equally.

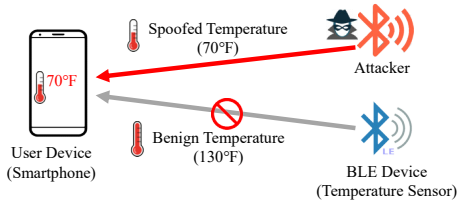

Figure 1: A typical BLE spoofing attack scenario.

Figure 1 presents a spoofing attack where an attacker device impersonates a benign BLE device (e.g., a temperature sensor) and feeds malicious data to the user device (e.g., a smartphone). Spoofing attacks are becoming critical, especially given that BLE devices are now widely integrated into security and privacy-sensitive scenarios [6, 18, 27], including manufacturing plants, medical and health care, and even physical-security monitoring.

The BLE specification (p. 2798 in [5]) provides a variety of authentication mechanisms that can potentially be employed to prevent spoofing attacks. Yet in practice, these mechanisms fail to serve their purpose due to two reasons: (1) A large proportion of BLE devices are limited by their I/O capabilities, and hence cannot employ any secure authentication mechanism. Unsurprisingly, a recent study has revealed that nearly 80% of existing BLE devices communicate with their user devices in plaintext without any authentication [30]. (2) Even in BLE devices employing different security mechanisms, there exist a variety of attack surfaces, at both BLE protocol-level [49] and application-level [37], which can be exploited by malicious attackers to launch spoofing attacks.

Unfortunately, deploying a software-based solution (i.e., firmware updating of BLE devices and software patching on user devices) to prevent spoofing attacks is challenging due to three practical challenges: (1) Software patching cannot defend against zero-day spoof-enabling vulnerabilities [49]. (2) Firmware updates to mitigate the vulnerabilities of BLE devices cannot be uniformly created and applied at a large scale by the wide range of BLE device vendors [21]. (3) Even worse, there are millions of (already deployed) legacy BLE devices in the field that do not support firmware updates due to their limited I/O capabilities.

An Out-of-the-Box Defense. To address the aforementioned challenges, we propose *BlueShield*, a vulnerability-agnostic, legacy-friendly and non-intrusive monitoring system

to protect BLE devices against spoofing attacks. The design of BlueShield is motivated by the critical observation that, in a spoofing attack, the attacker must broadcast *advertising packets* (containing the BLE device's identity) on BLE *advertising channels* to make itself discoverable by the user device. As such, BlueShield detects the spoofing attack by distinguishing the advertising packets transmitted by the attacker and the benign BLE device.

To collect all over-the-air packets on BLE advertising channels, BlueShield sets up a monitoring infrastructure covering the physical environment where BLE devices are deployed. Then, BlueShield inspects a combination of *cyber* and *physical* features extracted from the advertising packets of the target BLE device, and raises an alarm (indicating the presence of a spoofing attack) if it detects any anomaly. Here, while the cyber features include the advertising pattern and state transitions of the BLE device, the physical features include advertising interval, the RF (radio frequency) signal frequency offset, and the RF signal strength (Section 4.1).

The effectiveness of BlueShield against spoofing attacks stems from its two novel mechanisms: (1) To detect an advanced attacker, such as a software-defined radio (SDR)-enabled attacker with the capability to mimic all the physical features, BlueShield employs a *randomized channel switching* mechanism (Section 4.3.1) to collect physical features of the monitored device across different advertising channels. This mechanism brings forth a "moving target defense" so that the attacker cannot reliably predict and mimic the correct values of the monitored features to evade the detection. (2) BlueShield also employs a *selective inspection* mechanism (Section 4.3.2) for triggering the alarm: For each received advertising packet, BlueShield inspects one or more of the "physical" features which are selected based on the determined "cyber" features of the BLE device. Then, an anomalous value is detected by comparing the current physical feature values with their valid ranges. This mechanism effectively mitigates the occurrence of false alarms due to unintentional interference in physical features, while performing in-time detection of spoofing attacks with good precision. As such, the two aforementioned mechanisms ensure that the selected features (even individually) cannot be easily imitated or controlled by an attacker, and hence can be utilized together to robustly detect spoofing attacks.

We evaluate BlueShield on nine BLE devices (including temperature sensors and smart locks) covering different types of devices widely deployed in security/safety-sensitive environments, such as smart homes, museums, and manufacturing plants. We launch various spoofing attacks with different strategies and from different locations in a real-world, noisy office environment. Our evaluation results demonstrate that BlueShield can effectively detect spoofing attacks with an average success rate of 99.28% on our tested BLE devices. Moreover, even in a usage-heavy scenario BlueShield only introduces less than one false alarm a week.

Compared with defenses in the existing literature [21, 28, 37], BlueShield provides four practical advantages: (1) Since the monitoring infrastructure captures and analyzes all traffic on advertising channels, BlueShield supports concurrent monitoring of many BLE devices. (2) BlueShield is deployable in all BLE networks regardless of the capability of the BLE device and the version of its BLE implementation. (3) BlueShield is fully transparent to its deployed environment, i.e., it does not introduce any interference or energy overhead to the BLE devices and user devices. (4) BlueShield does not require any firmware modification or reverse engineering of the BLE devices or user devices.

Our major contributions are summarized as follows:

- We propose BlueShield, a generic, device-agnostic monitoring framework to detect spoofing attacks in BLE networks using a novel combination of selected cyber and physical features of the BLE devices.
- We devise a set of mechanisms to utilize the selected features for detecting spoofing attacks. These mechanisms not only ensure the robustness of BlueShield against an advanced SDR-enabled attacker, but also provide good effectiveness with very low false alarms.
- We have implemented BlueShield using off-the-shelf components and thoroughly evaluated its effectiveness and efficiency in a real-world environment.

2 Background and Motivation

2.1 BLE Basics

BLE is typically deployed in a network that requires an energy-constrained low-cost device (e.g., temperature sensor) to record an attribute/data value (temperature) and communicate the data to a user device (e.g., smartphone) over the wireless medium. In BLE, three radio frequency (RF) channels (channel-37, channel-38, and channel-39) are utilized for advertising and initializing the connection with the user device and are called *advertising channels*. The other channels are utilized for communicating data and are called *data channels*. The typical communication procedure between the BLE device and the user device can be broadly classified into four steps: advertising, connecting, pairing, and accessing.

Advertising. The BLE device publicizes its presence and facilitates its discovery by periodically broadcasting *advertising packets* on the three advertising channels. Each advertising packet contains a unique identifier of the BLE device and the information about services provided by it.

Connecting. The user device scans the three advertising channels and discovers the BLE device using its advertising packet. If the user device intends to establish a connection, it sends a *connection request* packet. If the BLE device accepts the connection, the BLE device and the user device start communication at data channels. We note that during

the connection state, most of the BLE devices stop public advertising and communicate only with the connected user devices. However, some BLE devices may keep advertising even during the connection state.

Pairing. The BLE specification provides multiple pairing methods to establish a secure channel for communication. Through the pairing procedure, the BLE device and the user device can generate a shared key that can be utilized for encryption and authentication of the communicated payloads.

Accessing. After establishing the connection in each session, the user device utilizes services provided by the BLE device by accessing corresponding data values, e.g., a temperature value provided by a smart temperature sensor.

2.2 Spoofing Attacks in BLE Networks

Reasons for Successful Spoofing Attacks. Recent studies have revealed that BLE networks expose various attack surfaces that can be exploited by an adversary to launch spoofing attacks [2, 33, 37, 46, 47]. Since around 80% of BLE devices do not perform secure pairing and communicate data without employing a secure authentication mechanism [30], they are vulnerable to spoofing attacks. Even for the remaining devices that employ authentication mechanisms after pairing securely, there exist zero-day vulnerabilities across different layers of the BLE stack due to flaws in the design and implementation of the BLE specification [1, 2, 46, 47]. All such vulnerabilities can be easily exploited by the attackers to perform spoofing attacks against authentication-enabled devices. In fact, we have also discovered a protocol-level spoof-enabling vulnerability that affects major platforms running the most recent version of iOS and Android [46]. We have responsibly reported it to Apple and Google[1]. Apple has acknowledged the reported vulnerability and assigned CVE-2020-9970 [14] to it.

Limitations of Existing Solutions. A straightforward solution to prevent spoofing attacks could be to upgrade the BLE devices with advanced security features, and in the meantime, adopt security patches to fix vulnerabilities. However, such solutions will not work well in practice due to the extremely fragmented nature of the BLE ecosystem. Specifically, the case-by-case firmware updates and security patches may not be preferably adopted by different vendors at a large-scale, since vendors always tend to sacrifice security for utility. Moreover, it is extremely difficult for those low-end, legacy BLE devices (e.g., beacons) with limited I/O capabilities to adopt any firmware updates or security patches. Furthermore, a solution that fails to ensure backward compatibility with the legacy BLE devices is unlikely to be deployed widely. Hence, these limitations of existing solutions call for a practical and legacy-friendly defense to effectively alleviate the threat of spoofing attacks in a more fundamental way.

[1]While Google has confirmed the vulnerability, we have been told that another researcher reported a similar vulnerability three days earlier than us.

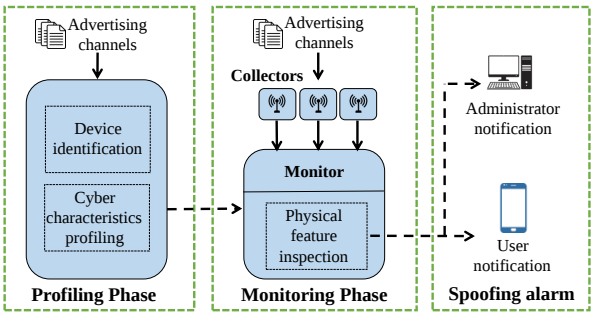

Figure 2: Architecture of BlueShield.

3 Overview of BlueShield

BlueShield is a monitoring framework for detecting spoofing attacks and alarming the user in a *stationary* BLE network. Specifically, it detects the footprint of spoofing attacks by noticing anomalies in the unique combination of cyber and physical features (Section 4.1) of advertising packets containing the BLE device's identity. Since the nature of a successful spoofing attack requires the attacker to broadcast the spoofed advertising packets *before* the appearance of a victim user, such activities expose the attacker in advance and provide a great deal of opportunity for BlueShield to detect before any actual damage is inflicted on the victim user.

Architecture and Workflow. Figure 2 presents the architecture of BlueShield. For each BLE device in an indoor environment that BlueShield aims to protect, the workflow goes through two phases, namely the *profiling phase* and the *monitoring phase*. The profiling phase (Section 4.2) involves the offline procedures to determine the BLE device's authentic protocol-level cyber features, such as MAC address, advertising data, and advertising pattern.

The monitoring phase (Section 4.3) includes runtime procedures to detect spoofing attacks using an infrastructure consisting of three *collectors* and one *monitor*. The three collectors follow a randomized channel switching mechanism (Section 4.3.1) through which each collector is randomly assigned to collect the advertising packets on one of the three advertising channels. Since the authentic value of the physical features (e.g., RF signal's strength) of the advertising packets would be different at different channels, this mechanism ensures that even an attacker with the capability to mimic all the physical features, cannot trick all the three collectors with correctly imitated values at the same time.

After obtaining the feature values of the advertising packets gathered by the three collectors, the monitor employs a selective inspection mechanism (Section 4.3.2) which exploits the cyber features obtained in the profiling phase to select appropriate physical features for an effective runtime inspection. Thereafter, the monitor determines the valid range of the selected physical features over each advertising channel. Finally, if the monitor discovers that the obtained feature val-

ues lie outside their valid ranges, it alarms the user and/or the network administrator about a detected spoofing attack.

Adversary Model. BlueShield is subject to the following assumptions: We assume that the attacker aims to impersonate a target BLE device and launch a spoofing attack against the user device. The attacker is capable of passively sniffing the traffic, and actively crafting and transmitting spoofed packets on BLE channels (e.g., using a customized BLE device). An advanced attacker may even employ advanced evasion techniques (e.g., with an SDR) and try to mimic the features of the wireless signals of the target BLE device. Besides, the attacker can also exploit the vulnerabilities of the authentication mechanism between the BLE device and user device [2]. However, we assume that the attacker cannot fully control the target BLE device either physically (e.g., remove/replace the target device with his own) or remotely (e.g., by corrupting its firmware). Also, we do not consider the attacker that uses jamming, as the detection of jamming is sufficiently covered by existing research [32, 48] which are orthogonal to BlueShield. Lastly, for BlueShield, we assume that the monitoring infrastructure for collecting the advertising packets is secure, i.e., it cannot be compromised by the attacker.

Scope of Detection. BlueShield focuses on detecting spoofing attacks that target *stationary BLE devices in indoor environments*. As such, BlueShield covers a wide range of BLE usage scenarios (e.g., smart homes, museums, and manufacturing plants) [17]. Specifically, a recent report [22] showed that 15 out of the 18 (83.3%) most popular IoT devices in smart homes are stationary. We note that spoofing detection in outdoor environments or for movable devices (e.g., fitness trackers) is out of the scope of BlueShield.

4 Detailed Design

In this section, we first describe the physical and cyber features inspected by BlueShield. We then elaborate on how these features are used in the profiling and monitoring phases for enabling an effective detection of spoofing attacks.

4.1 Inspected Features

4.1.1 Physical Features

Advertising Interval (INT). We define INT as the time between two consecutive advertising packets on the *same* advertising channel. In spoofing attacks, if both the BLE device and attacker are broadcasting advertising packets, the real-time INT can be lower than the expected INT value, and hence can be utilized to detect the spoofing attacks.

Carrier Frequency Offset (CFO). The CFO of a given device depends on its RF circuit's imperfections which induce a unique mismatch/offset between the designated and the actual carrier frequency of the transmitted signal. As a result,

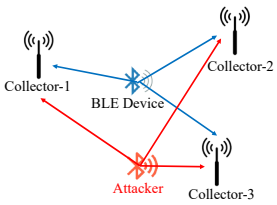

Figure 3: Locations of the BLE device, the collectors, and the attacker.

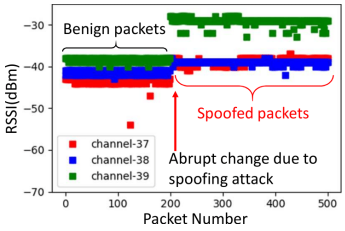

Figure 4: Observed RSSI values under spoofing attacks (with typical channel interference).

CFO can be utilized as a fingerprint of a wireless device [7, 39]. In the case of spoofing attacks, the CFO extracted from advertising packets can be used to differentiate the packets generated by the benign BLE device and the attacker.

Received Signal Strength Indicator (RSSI). RSSI is a numerical estimate of the signal power in the received RF signal, which depends on the distance and the channel interference between a transmitter and receiver [19]. In the prior art, RSSI has been widely used for fingerprinting the location of a wireless device [19, 24, 50]. In the case of spoofing attacks, since the locations of the BLE device and the attacker are different as illustrated in Figure 3, the RSSI values of the benign and spoofed advertising packets can be utilized to discover spoofing attacks. For instance, as shown in Figure 4, the abrupt change in RSSI values of advertising packets indicates the presence of a spoofing attack.

4.1.2 Cyber Features

Although the physical features enable BlueShield to detect spoofing attacks, they share some inherent limitations. Specifically, the INT feature will only work for BLE devices which persistently transmit advertising packets before and after their connections with user devices. In the meantime, as shown in Figure 4, like any feature of wireless signals, the values of the CFO and RSSI can change due to unintentional signal interference (e.g., human movements). Hence, continuously inspecting these two features will trigger a significant number of false alarms. To overcome these challenges, BlueShield employs additional cyber features that are determined by device-specific BLE implementation, to support the usage of the aforementioned physical features.

Advertising Pattern. According to the BLE specification, a BLE device can illustrate one of the two advertising patterns: intermittent and persistent. A BLE device with an *intermittent advertising pattern* stops advertising after connecting to a user device. A BLE device with a *persistent advertising pattern* continues to advertise even after connecting to a user device.

Operation State. The BLE specification defines that every BLE device stays active in one of the two states: advertising and connection. In the *advertising state*, the BLE device

Table 1: An illustrative sample of characteristics of a BLE device recorded during the profiling phase.

Characteristic	Value
Device ID & Name	1, n097w
MAC Address	0xD1 76 A3 1A F4 7F
Advertising Data	0x06 09 4E 30 39 37 57
Advertising Pattern	Intermittent
Lower Bound of INT	1280 ms

makes itself discoverable by broadcasting advertising packets periodically. When the BLE device receives a connection request packet from the user device, the BLE device connects to the user device and makes the transition to the *connection state*. From the connection state, the BLE device returns to the advertising state if it does not communicate with the user device for a specified timeout period or disconnects from the user device. We highlight that for a BLE device with an intermittent advertising pattern, the advertising-to-connection state transition can be detected by observing a connection request packet on an advertising channel.

The two cyber features mentioned above naturally support BlueShield to selectively choose one or more of the physical features for detecting spoofing attacks. Specifically, for BLE devices with the persistent advertising pattern, inspecting only the INT values is sufficient to detect potential spoofing attacks. For other BLE devices that follow the intermittent advertising pattern, the attacker can stop the benign BLE device from advertising by connecting to it. Then the attacker can start transmitting spoofed advertising packets with the same advertising period as the benign BLE device. To detect such an attacker, the CFO and RSSI features can be used once BlueShield detects that there is an advertising-to-connection state transition in the BLE device.

4.2 Profiling Phase

Now we describe the procedures performed in the offline profiling phase of BlueShield. To obtain the data-of-interest of a target BLE device, the monitor records and analyzes the advertising packets of the BLE device. First, from the packet content, the monitor extracts the device name, the MAC address, and the advertising data which can be utilized to identify packets transmitted by the BLE device. We note that although some BLE devices employ address randomization to anonymize their identity (p. 2198 in [5]), some of the fields (e.g., device name) in their packets remain unchanged and can be used to relate the packets to the BLE device [13, 20].

The monitor then computes the INT value by subtracting the time-of-arrival of the current advertising packet from that of the previous advertising packet. As defined by the BLE specification (p. 2750 in [5]), the observed INT is equal to a fixed *advertising period* plus a random delay between 0 and 10 ms. To this end, the monitor calculates the lower bound

(a) Authentic RSSI values (in dBm) corresponding to the BLE device at different advertising channels.

Channel	Collector 1	Collector 2	Collector 3
Channel 37	-60.3	-48.6	-39.2
Channel 38	-63.5	-45.2	-35.8
Channel 39	-58.1	-46.3	-36.9

(b) Assigned channel and corresponding RSSI value during different time periods at each collector.

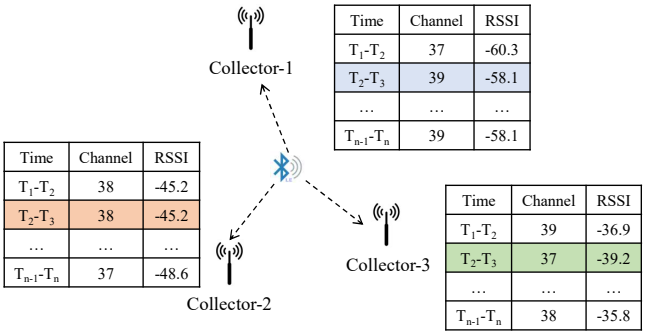

Figure 5: An illustration of the randomized channel switching.

of INT as the shortest observed INT minus 10 ms, which represents the lowest possible interval between any two advertising packets on the same channel. Further, the monitor determines the authentic advertising pattern (i.e., persistent or intermittent) by checking if it observes the BLE device's advertising packets after connecting to the BLE device. Finally, the monitor stores the determined characteristics of the BLE device along with an assigned device identifier (ID) as shown in Table 1. The deployment details regarding the profiling phase are discussed further in Section 7.

4.3 Monitoring Phase

After the profiling phase, BlueShield is ready to detect spoofing attacks in its runtime monitoring phase.

4.3.1 Feature Collection

BlueShield faces the critical challenge of detecting an attacker which can attempt to: (1) start the spoofing attack at any time, (2) transmit spoofed advertising packets at any of the three advertising channels, and (3) hide by trying to mimic the exact values of the selected physical features. To tackle these challenges, BlueShield first places the three collectors at three different locations to comprehensively cover the monitored environment in space, and then ceaselessly records all advertising packets transmitted at all three advertising channels using the following mechanism.

Randomized Channel Switching. In addition to the spatial diversity of the collectors, BlueShield introduces randomness in the monitoring schedule of each collector. To achieve this, BlueShield assigns the three advertising channels to the three

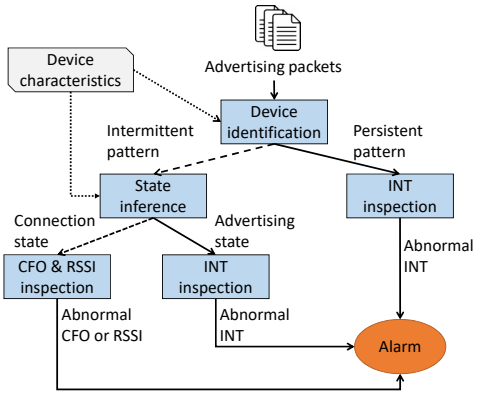

Figure 6: Usage of the cyber-physical features for selective inspection of the physical features in BlueShield.

collectors with a uniformly random distribution. After a very short random interval of time (which is significantly shorter than the BLE device advertising interval), the monitor shuffles and re-assigns channels to the collectors. Since the values of each physical feature (i.e., RSSI and CFO) in different advertising channels are different, the attacker will not be able to precisely predict the expected feature values at runtime. This randomized collection makes BlueShield robust against advanced attackers (e.g., an SDR-enabled attacker) that try to mimic all the monitored physical features at the same time.

Figure 5 demonstrates a running example of this mechanism by presenting the randomly assigned channel and the expected RSSI value at each collector during different periods. As such, since the channel assignment to the collectors is random, an attacker will not be able to accurately guess the current "time-channel-collector" mapping. In other words, the attacker will not know the exact RSSI value that it needs to mimic for a particular collector at a specific time. We provide an analytical evaluation of this mechanism in Section 6.2.2.

4.3.2 Runtime Selective Inspection

After retrieving the advertising packets and their features from the collectors, the monitor proceeds with a runtime selective inspection of these features for each BLE device. As shown in Figure 6, the cyber features of the target BLE device allow BlueShield to adaptively employ the appropriate inspection mechanism over the physical features. This selective inspection significantly lowers the false alarms caused by signal interference as will be further discussed in Section 6.2.1.

Now we describe the inspection mechanisms as follows.

INT Inspection. The monitor proceeds with this inspection mechanism for each received advertising packet from the BLE device. Recall that by definition, the INT between any two advertising packets must always be more than the *lower bound* of INT. Hence, if the runtime computed INT value is less than the lower bound of INT, the monitor considers it an anomaly and raises an alarm.

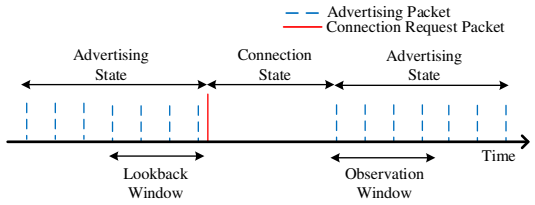

Figure 7: Illustration of the lookback and observation windows in CFO and RSSI inspection.

CFO and RSSI Inspection. BlueShield ceaselessly records the CFO and RSSI values of the advertising packets. When the CFO and RSSI inspection is triggered, BlueShield utilizes these values and proceeds through the following steps. As shown in Figure 7, for a BLE device with an intermittent advertising pattern, we define the *lookback window* as the duration of time T_l (with N_l packets) before the advertising-to-connection state transition, and the *observation window* as the duration of time T_o (with N_o packets) after the connection-to-advertising state transition. In BlueShield, after observing a connection request packet, the monitor triggers the CFO and RSSI inspection for advertising packets received from each collector in each of the three advertising channels. The monitor first utilizes CFO and RSSI values of advertising packets in the lookback window to compute their valid ranges, then it inspects these values of advertising packets in the observation window. If the monitor detects an anomaly in either of these two features, it raises an alarm. Below we elaborate on the process of CFO inspection. The RSSI inspection follows similar steps as shown in Appendix A.

The CFO values corresponding to a BLE device follow a Gaussian distribution [43]. Hence, when the mean and standard deviation of CFO values are denoted by μ_0 and σ_0, respectively, the probability distribution function of CFO values can be computed as $f_c(x_i) = \frac{1}{\sigma_0\sqrt{2\pi}} \cdot e^{-(x_i-\mu_0)^2/2\sigma_0^2}$, where x_i denotes a CFO sample. In BlueShield, using the N_l CFO values of advertising packets in the lookback window, the monitor computes μ_0 and σ_0, and then sets their values in the above function. Now if the advertising packets in the lookback and observation windows are generated by the same BLE device, the CFO values of advertising packets in the observation window must statistically follow the above distribution. To verify this, the monitor computes the negative log-likelihood of the CFO values in the observation window, i.e., $L_c = \frac{1}{N_o}\sum_{i=1}^{N_o} -\log f_c(x_i)$. If the log-likelihood is below a CFO inspection threshold denoted by τ_c, i.e., $L_c \leq \tau_c$, the CFO values are considered to belong to the benign BLE device. Here, τ_c is a design parameter in BlueShield which determines the valid range of CFO values in the observation window. However, if the log-likelihood exceeds the CFO inspection threshold, i.e., $L_c > \tau_c$, the monitor considers it an anomaly and triggers an alarm indicating a spoofing attack. The impact of this parameter is further discussed in Section 6.

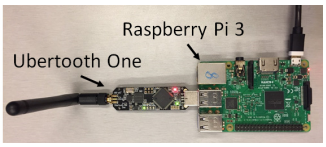

Figure 8: Prototype of a collector utilized in BlueShield.

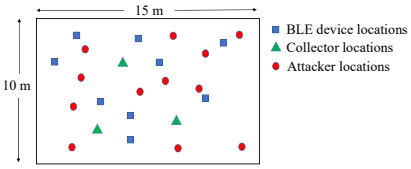

Figure 9: Locations of collectors, BLE devices and attackers within the office.

Figure 10: BLE devices used in our experiments.

5 Implementation

BlueShield can be readily implemented using low-cost, off-the-shelf platforms. We implemented the collector using an Ubertooth One radio [41] connected to a Raspberry Pi running Linux 4.14 (Figure 8). The total cost for such a collector is around $100. We note that when a collector is deployed on a custom-designed platform, the per-unit cost could be less than $5 for a BLE module [40]. The Ubertooth first captures the packets on advertising channels. Then, to retrieve the physical features, we modified the Ubertooth firmware to provide CFO and RSSI values for each received packet. Such customization is feasible as Ubertooth is an open platform for Bluetooth research and development. Finally, the Raspberry Pi communicates the packets along with their relevant features to the monitor. We implemented the monitor on an Ubuntu 18.04 Desktop PC. At the monitor, all processes including interacting with collectors, parsing the received information from collectors, and runtime inspection mechanisms were implemented with $\approx$ 3 K lines of Python code.

User Notification. The design of BlueShield supports notification of detected spoofing attacks. To demonstrate this use case, we have developed an Android application that employs a push-based mechanism to receive notifications from BlueShield's monitor through a secure HTTPS connection (Appendix B). The user can register to BlueShield's notification service by installing and configuring the app. Then, whenever BlueShield detects a spoofing attack, the user receives the notification with relevant information, such as the targeted BLE device's name and MAC address.

6 Evaluation

6.1 Experiment Setup

Deployment Environment. We evaluated the detection performance of BlueShield in a real-world environment: a 15m $\times$ 10m office hosting multiple graduate students in 20 cubicles. We divided the office space into grids of 1m $\times$ 1m. We deployed BlueShield by placing the three collectors at selected grid locations within the office as shown in Figure 9. The office presents a typically noisy and challenging indoor environment for determining the detection performance of BlueShield. By recording RF signals within the reception range of the collectors, we discovered significant channel interference from 30 other Bluetooth/BLE-equipped devices (sensors, headsets, smartphones, and laptops), dozens of Wi-Fi access points and a microwave[2]. We also observed that abrupt movements of students within the office significantly altered channel conditions in the monitored environment.

Device Selection. To exhaustively evaluate BlueShield, we utilized nine different BLE devices which are shown in Figure 10. These BLE devices cover the mainstream BLE applications (e.g., temperature sensor, lock, and smoke detector) and popular manufacturers (e.g., Nest, August, and Eve) with a variety of Bluetooth chips (e.g., DA14580 and nRF51822). As shown in Figure 9, we randomly chose nine different locations within the office to place these BLE devices.

Attack Simulation. To carry out different types of attacks, we utilized four attacker platforms, a Dell Latitude 5480 laptop [15], a CSR 4.0 BT dongle [11], an HM-10 developmental board [25], and a CYW920735 developmental board [12]. These platforms were selected because they provide ease of access and programmability, and they utilized different transmit power values. Besides, to thoroughly evaluate the performance of BlueShield, we launched a variety of spoofing attacks from 12 different locations, some at the center and some at edges of the office (Figure 9). Further, to enrich the evaluation of the effectiveness of the CFO inspection, we utilized different copies of the same BLE device as attackers (in addition to the four attacker platforms). For the unbiased evaluation of the RSSI inspection, we utilized the same BLE device as the benign BLE device and the attacker, and collected its advertising packets by placing it at different locations within the office environment.

Experimental Data. For each BLE device, benign advertising packets were collected for 48 hours (throughout day and night). For each attacker platform placed at each location, spoofed advertising packets were collected for around 15 minutes. In total, we collected 5,507,978 advertising packets which are comprised of 80.7% benign advertising packets and 19.3% spoofed advertising packets. This data was utilized as the ground truth for our evaluation.

[2]We only saved and analyzed the data of BLE devices deployed by us. We did not record any data from other devices within the office environment.

Table 2: BlueShield's detection performance against spoofing attacks (FP and FN are presented in %).

Device ID	Device Name	Advertising Period (s)	Observation Window (s)	INT		CFO		RSSI		Overall	
				FP	FN	FP	FN	FP	FN	FP	FN
1	Nest Protect Smoke Detector	1.28	3.84	0.00	0.00	0.80	0.00	0.97	5.84	**1.76**	**0.00**
2	Nest Cam Indoor Camera	0.15	0.45	0.00	0.00	1.38	17.74	3.59	21.15	**4.92**	**3.69**
3	SensorPush Temperature Sensor	1.28	3.84	0.00	0.00	0.56	4.46	1.43	5.22	**1.98**	**0.23**
4	Tahmo Tempi Temperature Sensor	2.00	6.00	0.00	0.00	0.64	0.00	1.32	22.94	**1.95**	**0.00**
5	August Smart Lock	0.30	0.90	0.00	0.00	1.12	4.85	1.26	1.60	**2.37**	**0.08**
6	Eve Door&Window Sensor	1.28	3.84	0.00	0.00	0.77	8.17	1.64	1.46	**2.40**	**0.12**
7	Eve Button Remote Control	1.28	3.84	0.00	0.00	0.98	1.41	1.18	3.00	**2.15**	**0.04**
8	Eve Energy Socket	0.15	0.45	0.00	0.00	0.60	1.67	0.85	1.55	**1.44**	**0.03**
9	Ilumi Smart Light Bulb	0.10	0.30	0.00	0.00	0.88	14.28	1.48	15.73	**2.35**	**2.25**
	Average	0.87	2.61	0.00	0.00	0.86	5.84	1.52	8.72	**2.37**	**0.72**

6.2 Overall Effectiveness

The detection performance of BlueShield is evaluated in terms of two metrics: false positive (FP) and false negative (FN) misclassifications. A false positive refers to the error in which BlueShield generates a false alarm for a spoofing attack after inspecting benign advertising packets generated by the benign BLE device. A false negative refers to the error in which BlueShield fails to detect a spoofing attack after analyzing spoofed advertising packets from the attacker.

We highlight that the INT inspection, and the CFO and RSSI inspection are exclusively employed to detect spoofing attacks under different scenarios (Figure 6). The INT inspection readily uncovers an attack in which the attacker transmits spoofed advertising packets while the BLE device is broadcasting benign advertising packets. In such a scenario, the INT inspection does not introduce any FP and causes negligible FN in detecting attacks (Appendix C). However, the INT inspection cannot be used to detect an advanced attack where the attacker first suppresses the broadcast of benign advertising packets by connecting to the BLE device whose advertising pattern is intermittent, and then transmits spoofed advertising packets with the same advertising period. BlueShield employs the CFO and RSSI inspection to detect this type of attack. Since BlueShield raises an alarm for a spoofing attack when either CFO inspection or RSSI inspection detects an anomaly, BlueShield significantly reduces FN at the cost of a slight increase in FP. Theoretically, since these inspection mechanisms employ independent features, the overall FN of BlueShield can be calculated as $FN_{BlueShield} = FN_{CFO} \times FN_{RSSI}$. Also, the overall FP generated by BlueShield can be calculated as $FP_{BlueShield} = FP_{CFO} + FP_{RSSI} - FP_{CFO} \times FP_{RSSI}$.

6.2.1 Summary

As shown in Table 2, BlueShield achieves an average of 2.37% FP rate and 0.72% FN rate on our tested BLE devices. The results are obtained when the inspection thresholds τ_c and τ_r in the CFO and RSSI inspection are set to 3 and 5 respectively. The numbers of advertising packets in the lookback window (N_l) and observation window (N_o) are set to 100 and 3, respectively. This means BlueShield is set to report a potential spoofing attack within three advertising periods, resulting in an average detection time of 2.61s based on the corresponding advertising period (see columns 3 and 4 in Table 2). Note that, we present our results with these reasonable values of parameters to illustrate a setting where BlueShield provides low FP values while detecting spoofing attacks against all nine BLE devices. BlueShield can also be employed with device-specific parameters with minimal effort to achieve an appropriate trade-off between FP and FN in a real-world deployment. For example, as shown in the receiver operating characteristic (ROC) curve in Figure 12, when the true positive decreases (i.e., FN increases), FP also decreases.

False Positive. As shown in Table 2, while the INT inspection does not introduce any FP, the average FP triggered by the CFO and RSSI inspection are 0.86% and 1.52% respectively. Overall, BlueShield achieves an average FP of 2.37%. To better illustrate the real-world impact of the FP rate, we consider a user device that makes five connections per day with the light bulb with Device $ID = 9$. We note that this is already a heavy usage scenario based on an IoT device usage frequency report [44]. As shown in Table 2, BlueShield achieves an average FP of 2.35% for this BLE device within an observation window of 0.3s. Hence, BlueShield will only cause an average of $5 \times 2.35/100 = 0.1175$ false alarm per day, which implies that there will be less than one false alarm during a one-week deployment. However, we point out that if we ceaselessly monitor the physical features of all packets transmitted by the BLE device, we would encounter $\frac{60\times60\times24\times2.35/100}{0.3} = 6768$ false alarms per day. Fortunately, BlueShield employs the selective inspection mechanism so that the CFO and RSSI inspection are triggered only when the BLE device makes an advertising-to-connection state transition. This mechanism drastically reduces such false alarms. Figure 11 further illustrates the impact of the selective inspection mechanism on the number of false alarms at different FP rates.

False Negative. In Table 2, we observe that while the INT inspection detects spoofing attacks quite robustly, the CFO and RSSI inspection achieve an average FN of 5.84% and 8.72% in detecting spoofing attacks. Overall, BlueShield achieves

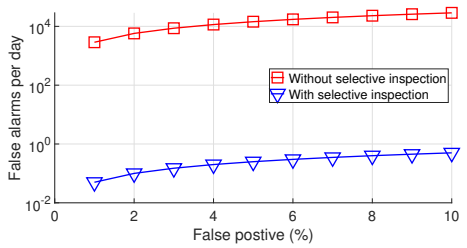

Figure 11: Number of false alarms per day with and without selective inspection mechanism in BlueShield.

an average FN of 0.72% which implies a detection rate of 99.28%. Further, we point out that most BLE devices force a disconnection after a certain time-out (e.g., 30 s). Therefore, to incessantly suppress benign advertising packets and replace them with spoofed advertising packets, an attacker may need to trigger multiple connections with the BLE device. As a result, the attacker is exposed to multiple CFO and RSSI inspections by BlueShield. For instance, if the attacker triggers three connections during a spoofing attack, BlueShield can fail to detect the attack with an average FN of only $3.7 \times 10^{-7}\%$ $(= 0.0072^3)$.

6.2.2 Robustness against Advanced Attacker

As mentioned earlier, BlueShield's randomized channel switching (Section 4.3.1) ensures that even for an advanced attacker which can precisely mimic the values of all physical features, there is barely any chance to evade detection. We evaluate the detection performance of BlueShield against such attacks as follows: In the randomized channel switching mechanism, the number of ways in which the monitor can assign N_c collectors the task to record and cover all the three advertising channels is given by $N_c! \binom{N_c-1}{3-1} = \frac{N_c!(N_c-1)(N_c-2)}{2}$. Hence, the probability with which an attacker can correctly guess the channel assignments over N_o advertising intervals within the observation window is given by $(\frac{2}{N_c!(N_c-1)(N_c-2)})^{N_o}$. For the three collectors (i.e., $N_c = 3$) recording the feature values over three advertising intervals (i.e., $N_o = 3$ as shown in Table 2), the probability of correct guesses by the attacker can be calculated as $(\frac{1}{6})^3 \approx 0.46\%$. As a result, the average detection rate of a spoofing attack launched by this type of attack over *one observation window* can be calculated as 99.54%. To this end, we conclude that the randomized channel switching enables BlueShield to effectively detect attackers that try to mimic all physical features used for detection.

6.3 Effectiveness of Individual Features

INT Feature. In the INT inspection, an anomaly is detected when the observed INT is less than the lower bound of INT. Since each of the benign BLE device's INT is always larger than the calculated lower bound of INT, by definition, there cannot be any FP in the INT inspection. Regarding the FN, recall that every broadcast of a spoofed advertising packet by the attacker results in an observed INT value that is smaller than the lower bound of INT. Hence, by noticing such an anomaly, the INT inspection detects the spoofing attack with negligible FN. Due to space limitation, we leave the detailed theoretical analysis about the FN in Appendix C.

Table 3: Effectiveness of the CFO inspection in differentiating packets transmitted by different platforms.

Device ID	FP	FN				
		Dev-1[1]	Dev-2[2]	Dongle[3]	Laptop[4]	Average
1	0.80	0.00	0.00	0.00	0.00	0.00
2	1.38	0.00	5.41	64.46	0.00	17.74
3	0.56	0.00	0.00	17.84	0.00	4.46
4	0.64	0.00	0.00	0.00	0.00	0.00
5	1.12	0.00	18.97	0.41	0.00	4.85
6	0.77	0.00	15.17	17.50	0.00	8.17
7	0.98	0.00	0.00	5.64	0.00	1.41
8	0.60	0.00	0.02	6.67	0.00	1.67
9	0.88	0.00	50.80	0.00	6.33	14.28
Average	**0.86**	**0.00**	**10.04**	**12.50**	**0.70**	**5.84**

1. HM-10 Bluetooth chip with expansion shield.
2. CYW920735Q60EVB-01 evaluation kit.
3. CSR 4.0 Bluetooth USB adapter.
4. Dell Latitude 5480 with built-in Qualcomm Bluetooth chip.

Table 4: CFO inspection results while using different copies of Nest Protect smoke detector as the BLE device and attacker.

	FP	FN		
		Copy-1	Copy-2	Copy-3
Copy-1	0.00	N/A	16.41	0.00
Copy-2	3.85	12.32	N/A	0.06
Copy-3	0.96	0.00	0.00	N/A

CFO Feature. Through our extensive experiments, we validate two critical characteristics of the CFO feature. (1) The CFO value of a BLE device is not affected by changing its relative location to collectors. This is because CFO is a device-specific feature, i.e., it is related to the device's RF circuit. (2) The CFO value is robust against interference from human movements as it only depends on RF signals' frequency values rather than their amplitude values. Table 3 shows the results obtained by launching spoofing attacks on the nine BLE devices by the four attacker platforms. We observe that spoofing attacks can be robustly detected using the CFO inspection in most cases. Moreover, Table 4 shows the performance of CFO inspection when using different copies of the same BLE device as the benign BLE device and the attack device. We observe that even though the different copies have the same Bluetooth chip from the same manufacturer with the same version, the copy used as the BLE device can be readily differentiated from the copy used as the attacker.

In a few cases, BlueShield has relatively high FN values (i.e., higher than 15%). Our further analysis shows that such high FN values are mainly caused by the limited CFO reso-

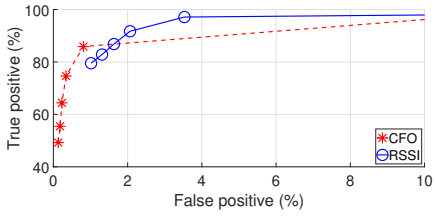

Figure 12: Receiver operating curve generated by changing the inspection threshold (Device $ID = 9$, $T_o = 0.3$ s.).

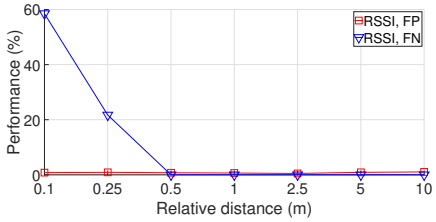

Figure 13: Detection performance vs. distance between BLE device and attacker (Device $ID = 9$, $\tau_r = 5$, $T_o = 0.3$ s.).

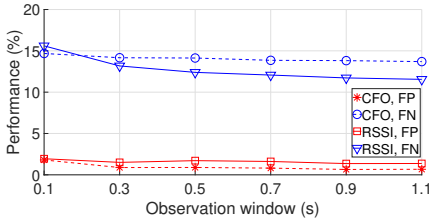

Figure 14: Detection performance vs. the duration of observation window (Device $ID = 9$, $\tau_c = 3$, $\tau_r = 5$).

Table 5: Effectiveness of the RSSI inspection in detecting packets transmitted from different locations.

Device ID	Night-time (8 pm-8 am)		Daytime (8 am-8 pm)		Average	
	FP	FN	FP	FN	FP	FN
1	1.25	0.00	0.69	9.44	0.97	5.84
2	3.00	5.82	4.18	33.62	3.59	21.15
3	0.11	4.19	2.75	4.18	1.43	5.22
4	0.57	3.36	2.06	37.07	1.32	22.94
5	1.68	0.46	0.84	2.12	1.26	1.60
6	1.22	0.41	2.05	2.32	1.64	1.46
7	1.34	0.02	1.02	5.86	1.18	3.00
8	0.40	2.88	1.29	0.05	0.85	1.55
9	1.10	5.92	1.85	20.47	1.48	15.73
Average	**1.19**	**2.56**	**1.86**	**12.79**	**1.52**	**8.72**

lution captured by Ubertooth One (the platform we used as the collector). More specifically, CFO in Ubertooth One is estimated with a resolution of 5.2KHz (p. 42 in [23]). This implies that any difference below 5.2KHz between two BLE devices cannot be detected by Ubertooth One. As such, a customized collector with a finer resolution will significantly enhance the detection performance. Nevertheless, even in these cases, the RSSI inspection alleviates such limitations, and the BlueShield's overall FN rate is significantly lower than the CFO-only inspection.

RSSI Feature. We evaluate the effectiveness of RSSI inspection by analyzing the RSSI values of the attacker's advertising packets from different locations. Figure 13 illustrates that spoofing attacks can be readily detected with low FN using the RSSI inspection when the attacker and the BLE device are placed at more than 0.25m. Further, we evaluate the RSSI inspection with (during daytime) and without (during night-time) significant interference from other wireless devices and human movements. In Table 5, we observe that while there is no significant impact of interference on the FP, the FN increases notably with the increase in interference. For instance, the FNs in detecting spoofing attacks on BLE devices with $ID = 2$ and $ID = 4$ are more than 30% during the daytime. This is because these BLE devices transmit advertising packets with lower signal power than other BLE devices, and the high channel interference conceals small differences in RSSI values of these BLE devices and attackers. We note that for these cases, the CFO inspection naturally complements the RSSI inspection in effectively bringing down the BlueShield's overall FN as shown in Table 2.

6.4 Responsiveness

The responsiveness of BlueShield is measured by the duration of the observation window T_o. Figure 14 illustrates that with increasing T_o, FP values of CFO and RSSI inspection do not change significantly, but their FN values decrease. Recall that BlueShield is implemented to report a spoofing attack by inspecting CFO and RSSI values of N_o advertising packets in the observation window T_o (Figure 7). As such, the specific values of N_o and T_o can be determined based on the required detection performance. We also highlight that since different BLE devices have different advertising periods, if we set N_o to a fixed value for all BLE devices, then T_o is different for different BLE devices, and vice versa.

For the results presented in Table 2, we configure BlueShield to have the same N_o but different T_o for different BLE devices. In the worst case, the observation window is limited to 6 s because of the requirement to achieve good detection performance for the BLE device ($ID = 4$) with the advertising period of 2 s. We point out that when BlueShield is deployed in real-world usage scenarios to monitor BLE devices with lower advertising periods, it can be optimized to have shorter observation windows. Overall, the fast responsiveness of BlueShield enables it to effectively detect the presence of the attacker before it can potentially spoof the user device. Such in advance alarms also enable BlueShield to quickly take other necessary actions (e.g., notify the network administrator) to *prevent* any harm to users.

6.5 Monitoring of Multiple BLE Devices

The number of BLE devices which BlueShield can monitor at the same time relies on the computation and communication capabilities of collectors and the monitor, and the channel interference in the monitored environment. We observe that with an increase in the number of monitored BLE devices, while the computational overhead at the monitor increases linearly, the computational burden at collectors does not change.

This is because collectors first need to record all packets on the three advertising channels and forward them to the monitor. The monitor is then responsible for classifying advertising packets belonging to different BLE devices and detecting spoofing attacks for each monitored BLE device. Our experiments also validate that BlueShield can monitor at least 30 BLE devices at the same time without any degradation in the detection performance corresponding to the monitored BLE devices. More detailed results are shown in Appendix D.

7 Discussion

Physical Features. BlueShield exploits the physical features of BLE devices' RF signals for device characterization. While some of these features (e.g., RSSI) have been previously employed to detect spoofing attacks in other wireless networks (e.g., Wi-Fi and ZigBee [10, 16, 35]), BlueShield augments three novel traits to enhance their effectiveness: (1) BlueShield employs a randomized channel switching mechanism that reflects the moving target defense and ensures a robust detection of spoofing attacks launched by an SDR-enabled attacker that can mimic the selected physical features. (2) BlueShield leverages cyber features (i.e., characteristics of the BLE protocol) to trigger the RSSI and CFO inspection at appropriate time instants. Such selective inspection allows BlueShield to significantly reduce false alarms compared to prior schemes. (3) Different from prior research which requires customized hardware to provide high-accuracy values of these features, BlueShield can be implemented using low-cost, off-the-shelf platforms without high-resolution and high-consistency RSSI and CFO values.

Deployment Considerations. The monitoring infrastructure required for implementing BlueShield (i.e., the collectors and the monitor) can be conveniently deployed on edge devices such as Wi-Fi/Bluetooth routers [9] which are widely utilized in indoor environments. These edge devices offer the natural choice to deploy BlueShield in a non-intrusive and economical way as they may already be equipped with BLE transceivers. For profiling a given BLE device, BlueShield only needs to make a one-time effort and collect a few advertising packets for determining the BLE device's relevant characteristics. Hence, the time needed to execute the profiling phase and add a new BLE device to the monitored environment is short (within a few seconds). Also, the one-time profiling process conveniently supports dynamic addition/removal of BLE devices to/from a BLE network monitored by BlueShield, without affecting other BLE devices in the network. In other words, there's no need to re-profile the monitored BLE devices. Besides, since the protocol characteristics of a BLE device are location-independent, they can also be securely profiled at an isolated location before deployment in the production environment.

8 Related work

Spoofing attacks against traditional wired and wireless networks have been widely studied in the existing literature which broadly includes GPS spoofing [42], DNS spoofing [38] and ARP spoofing [45]. In the context of BLE networks, the usage of encryption and authentication mechanisms provided in the BLE specification largely depends on the application-specific requirements determined by BLE device manufacturers. As such, the BLE devices which do not employ the recommended security mechanisms are vulnerable to a variety of attacks [3, 8, 13, 34] which can be trivially launched against their users. Recent studies [2, 49] have also revealed the design and implementation flaws of the BLE stack, which can be easily exploited by attackers to perform spoofing attacks against the BLE devices and their user devices which employ the specified authentication mechanisms. The detrimental implications of spoofing attacks on user devices have also been partly identified in prior research [26, 31].

To defend against spoofing attacks, the existing schemes largely rely on modifying the BLE protocol or updating the firmware/hardware of BLE devices [21, 37], on a vulnerability-by-vulnerability basis. Unfortunately, these approaches are impractical for wide adoption and deployment in the real world, especially for the millions of legacy BLE devices – many of which do not even support firmware updates – produced by many device vendors. One prior approach towards an off-the-shelf solution for protecting the privacy of BLE devices is BLE-Guardian [20], which mainly broadcasts jamming signals to corrupt the BLE device's advertising packets that contain privacy-sensitive information. Unfortunately, BLE-Guardian cannot defend against spoofing attacks. Instead, BlueShield is a practical, device-agnostic spoofing detection framework, which protects user devices against spoofing attacks without any interference or modification to the conventional BLE devices and user devices.

9 Conclusion

In this paper, we propose BlueShield, an out-of-the-box defense that provides a device-agnostic, legacy-friendly monitoring framework for detecting spoofing attacks in BLE networks. We demonstrate that BlueShield is robust against an advanced attacker with the ability to spoof the monitored features of a BLE device. BlueShield can be implemented using low-cost, off-the-shelf components; and its operation remains transparent to the communications between the user device and the BLE device. Our evaluation results illustrate that BlueShield can effectively and robustly detect spoofing attacks with very low false positive and false negative rates.

Acknowledgments

We thank the anonymous reviewers for their valuable comments and suggestions. This work was supported in part by ONR under Grant N00014-18-1-2674. Any opinions, findings, and conclusions in this paper are those of the authors and do not necessarily reflect the views of the ONR.

References

[1] Daniele Antonioli, Nils Tippenhauer, and Kasper Bonne Rasmussen. Key negotiation downgrade attacks on bluetooth and bluetooth low energy. *ACM Trans. Inf. Syst. Secur.*, 0(ja).

[2] Daniele Antonioli, Nils Ole Tippenhauer, and Kasper Rasmussen. The KNOB is broken: Exploiting low entropy in the encryption key negotiation of Bluetooth BR/EDR. In *28th USENIX Security Symposium*, pages 1047–1061, August 2019.

[3] Vaibhav Bedi. Exploiting BLE smart bulb security using BtleJuice: A step-by-step guide. https://blog.attify.com/btlejuice-mitm-attack-smart-bulb/, 2018. Accessed: August 1, 2019.

[4] Bluetooth SIG. Bluetooth Market Update. https://www.bluetooth.com/bluetooth-resources/2019-bluetooth-market-update/, 2019. Accessed: August 1, 2019.

[5] Bluetooth SIG. Core Specifications 5.1. https://www.bluetooth.com/specifications/bluetooth-core-specification, 2019. Accessed: August 1, 2019.

[6] Bluetooth SIG. Smart industry. https://www.bluetooth.com/markets/smart-industry, 2019. Accessed: August 1, 2019.

[7] Vladimir Brik, Suman Banerjee, Marco Gruteser, and Sangho Oh. Wireless device identification with radiometric signatures. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, pages 116–127, 2008.

[8] Victor Casares. Mimo baby hack. https://medium.com/@victor_14768/mimo-baby-hack-ac7fa0ae3bfb, 2018. Accessed: August 1, 2019.

[9] Cassia Networks. Cassia hub: The Bluetooth router. https://www.cassianetworks.com/blog/cassia-hub-bluetooth-router/, 2019. Accessed: August 1, 2019.

[10] Yingying Chen, Jie Yang, Wade Trappe, and Richard P. Martin. Detecting and localizing identity-based attacks in wireless and sensor networks. *IEEE Transactions on Vehicular Technology*, 59(5):2418–2434, 2010.

[11] CSR. CSR 4.0 Bluetooth USB adapter. https://www.amazon.com/Bluetooth-Adapter-Songway-Computer-Keyboard/dp/B07KWVXBKZ/ref=sr_1_46?keywords=bluetooth+adapter+car+4.0&qid=1563227361&s=electronics&sr=1-46. Accessed: August 1, 2019.

[12] Cypress. CYW920735Q60EVB-01 Evaluation Kit. https://www.cypress.com/documentation/development-kitsboards/cyw920735q60evb-01-evaluation-kit. Accessed: August 1, 2019.

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

[14] National Vulnerability Database. CVE-2020-9970. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-9770. Accessed: June 26, 2020.

[15] Dell. Dell Latitude 5480. https://www.dell.com/en-us/work/shop/dell-laptops-and-notebooks/latitude-5480/spd/latitude-14-5480-laptop. Accessed: August 1, 2019.

[16] Murat Demirbas and Youngwhan Song. An RSSI-based scheme for sybil attack detection in wireless sensor networks. In *Proceedings of the International Symposium on on World of Wireless, Mobile and Multimedia Networks*, pages 564–570, 2006.

[17] Developex. BLE in smart home devices. https://developex.com/blog/ble-in-smart-home-devices/, 2017. Accessed: Aug 1, 2019.

[18] Medical Device and Diagnostic Industry. 3 reasons bluetooth is perfect in healthcare settings. https://www.mddionline.com/3-reasons-bluetooth-perfect-healthcare-settings, 2019. Accessed: August 1, 2019.

[19] Ramsey Faragher and Robert Harle. Location fingerprinting with Bluetooth low energy beacons. *IEEE Journal on Selected Areas in Communications*, 33(11):2418–2428, 2015.

[20] Kassem Fawaz, Kyu-Han Kim, and Kang G Shin. Protecting privacy of BLE device users. In *25th USENIX Security Symposium*, pages 1205–1221, 2016.

[21] Avinatan Hassidim, Yossi Matias, Moti Yung, and Alon Ziv. Ephemeral identifiers: Mitigating tracking & spoofing threats to BLE beacons. https://developers.google.com/beacons/eddystone-eidpreprint.pdf, 2016. Accessed: August 1, 2019.

[22] Software Testing Help. 18 most popular iot devices in 2020. https://www.softwaretestinghelp.com/iot-devices/. Accessed: August 1, 2019.

[23] Texas Instruments. CC2400 data sheet. http://www.ti.com/lit/ds/symlink/cc2400.pdf, 2006. Accessed: August 1, 2019.

[24] Zhu Jianyong, Luo Haiyong, Chen Zili, and Li Zhaohui. RSSI based Bluetooth low energy indoor positioning. In *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 526–533, 2014.

[25] Keyestudio. Ks0255 keyestudio bluetooth 4.0 expansion shield. https://wiki.keyestudio.com/Ks0255_keyestudio_Bluetooth_4.0_Expansion_Shield, 2019. Accessed: August 1, 2019.

[26] Constantinos Kolias, Lucas Copi, Fengwei Zhang, and Angelos Stavrou. Breaking BLE beacons for fun but mostly profit. In *Proceedings of the 10th European Workshop on Systems Security (EuroSec)*, pages 4:1–4:6, 2017.

[27] Kontakt.io. Beacons in healthcare. https://goto.kontakt.io/beacons-in-healthcare, 2019. Accessed: August 1, 2019.

[28] Angela Lonzetta, Peter Cope, Joseph Campbell, Bassam Mohd, and Thaier Hayajneh. Security vulnerabilities in Bluetooth technology as used in IoT. *Journal of Sensor and Actuator Networks*, 7(3):28, 2018.

[29] Geoffrey McLachlan and David Peel. *Finite mixture models*. John Wiley & Sons, 2004.

[30] Tal Melamed. BLE Application Hacking. https://www.owasp.org/images/archive/6/6f/20170811005623%21OWASP2017_HackingBLEApplications_TalMelamed.pdf, 2017. Accessed: August 1, 2019.

[31] Tal Melamed. An active man-in-the-middle attack on bluetooth smart devices. *Safety and Security Studies (2018)*, 15, 2018.

[32] Rajani Muraleedharan and Lisa Ann Osadciw. Jamming attack detection and countermeasures in wireless sensor network using ant system. In *Wireless Sensing and Processing*, volume 6248, page 62480G. International Society for Optics and Photonics, 2006.

[33] Muhammad Naveed, Xiao-yong Zhou, Soteris Demetriou, XiaoFeng Wang, and Carl A Gunter. Inside job: Understanding and mitigating the threat of external device mis-binding on android. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, pages 1–14, 2014.

[34] Mike Ryan. Bluetooth: With low energy comes low security. In *Proceedings of the 7th USENIX Conference on Offensive Technologies*, pages 1–7, 2013.

[35] Yong Sheng, Keren Tan, Guanling Chen, David Kotz, and Andrew Campbell. Detecting 802.11 MAC layer spoofing using received signal strength. In *IEEE International Conference on Computer Communications (INFOCOM)*, pages 1768–1776, 2008.

[36] GARY SIMS. Bluetooth mesh positioned to provide de-facto protocol for smart homes. https://www.androidauthority.com/bluetooth-mesh-for-smart-homes-940886/, 2019. Accessed: August 1, 2019.

[37] Pallavi Sivakumaran and Jorge Blasco. A study of the feasibility of co-located app attacks against BLE and a large-scale analysis of the current application-layer security landscape. In *USENIX Security Symposium*, pages 1–18, 2019.

[38] U Steinhoff, A Wiesmaier, and R Araújo. The state of the art in dns spoofing. In *Proc. 4th Intl. Conf. Applied Cryptography and Network Security (ACNS)*, 2006.

[39] Weiping Sun, Jeongyeup Paek, and Sunghyun Choi. CV-track: Leveraging carrier frequency offset variation for BLE signal detection. In *Proceedings of the 4th ACM Workshop on Hot Topics in Wireless (HotWireless)*, pages 1–5, 2017.

[40] John Teel. How to develop a sellable bluetooth low-energy (BLE) product. https://makezine.com/2016/08/01/develop-sellable-bluetooth-low-energy-ble-product/. Accessed: April 6, 2020.

[41] Ubertooth Developers. Ubertooth One. https://github.com/greatscottgadgets/ubertooth/wiki, 2019. Accessed: August 1, 2019.

[42] E. Vattapparamban, İ. Güvenç, A. İ. Yurekli, K. Akkaya, and S. Uluağaç. Drones for smart cities: Issues in cybersecurity, privacy, and public safety. In *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 216–221, 2016.

[43] Tien Dang Vo-Huu, Triet Dang Vo-Huu, and Guevara Noubir. Fingerprinting Wi-Fi devices using software defined radios. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pages 3–14, 2016.

[44] Voicebot. Smart speaker consumer adoption report 2018. https://voicebot.ai/2018/04/02/smart-speaker-owners-use-voice-assistants-nearly-3-times-per-day/. Accessed: August 1, 2019.

[45] Sean Whalen. An introduction to arp spoofing. *Node99 [Online Document], April*, 2001.

[46] Jianliang Wu, Yuhong Nan, Vireshwar Kumar, Antonio Tian, Dave (Jing)and Bianchi, Mathias Payer, and Dongyan Xu. Blesa: Spoofing attacks against reconnections in bluetooth low energy. In *Proceeding of the 14th USENIX Workshop on Offensive Technologies (WOOT'20)*, 2020.

[47] Fenghao Xu, Wenrui Diao, Zhou Li, Jiongyi Chen, and Kehuan Zhang. Badbluetooth: Breaking android security mechanisms via malicious Bluetooth peripherals. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS)*, pages 1–15, 2019.

[48] Wenyuan Xu, Ke Ma, Wade Trappe, and Yanyong Zhang. Jamming sensor networks: attack and defense strategies. *IEEE network*, 20(3):41–47, 2006.

[49] Yue Zhang, Jian Weng, Rajib Dey, Yier Jin, Zhiqiang Lin, and Xinwen Fu. On the (in)security of bluetooth low energy one-way secure connections only mode, 2019.

[50] Yuan Zhuang, Jun Yang, You Li, Longning Qi, and Naser El-Sheimy. Smartphone-based indoor localization with Bluetooth low energy beacons. *Sensors*, 16(5):596, 2016.

A RSSI Inspection

Inspection mechanism. To detect anomalies in RSSI values under the presence of strong signal reflections in BLE networks, we employ the two-component Gaussian mixture model. This is because the distribution of RSSI values in noise-rich environments can be modeled using two normal distributions [35]. The probability distribution function of RSSI values can be represented by

$$f_r(y_i) = w \cdot \frac{1}{\sigma_1\sqrt{2\pi}} \cdot e^{-\frac{(y_i-\mu_1)^2}{2\sigma_1^2}} + (1-w) \cdot \frac{1}{\sigma_2\sqrt{2\pi}} \cdot e^{-\frac{(y_i-\mu_2)^2}{2\sigma_2^2}}, \quad (1)$$

where μ_1 and μ_2 represent means of the two components, σ_1 and σ_2 represent standard deviations of the two components, w represents a weight parameter, and y_i denotes an RSSI sample. In BlueShield, the monitor utilizes N_l RSSI values of advertising packets in the lookback window to learn values of μ_1, μ_2, σ_1, σ_2 and w using a conventional expectation–maximization (EM) algorithm [29]. Then, the monitor computes the negative log-likelihood that RSSI values (y_i, $\forall i \in [1, N_o]$) of advertising packets in the observation window come from the model given by equation (1), i.e.,

$$L_r = \frac{1}{N_o} \sum_{i=1}^{N_o} -\log f_r(y_i), \quad (2)$$

Finally, the monitor detects an anomaly if the negative log-likelihood is larger than an RSSI inspection threshold denoted by τ_r, i.e., $L_r > \tau_r$. Here, τ_r is a design parameter in BlueShield which is further discussed in Section 6.

B BlueShield App

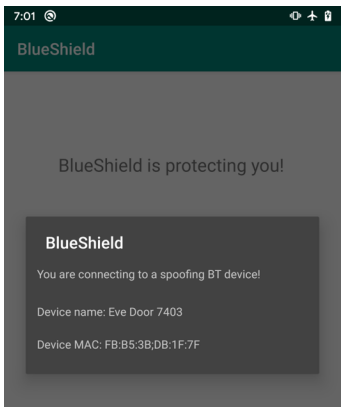

Figure 15: Sample of a notification of a spoofing attack shown in BlueShield's Android app.

We have developed an Android app which communicates with BlueShield's monitor through a secure HTTPS connection. If the user registers to BlueShield's notification service, the user can receive the notification about the spoofing attack with relevant information, such as the targeted BLE device's name and MAC address (Figure 15).

C Analysis of INT inspection

Here, by theoretically analyzing the INT inspection from the perspective of the lower bound of INT, we illustrate that the INT inspection mechanism encounters negligible false negative in detecting spoofing attacks.

In BlueShield, the lower bound of INT, T_{lb}, is calculated such that $T_{lb} \in [T_p - \Delta, T_p]$, where T_p is the advertising period

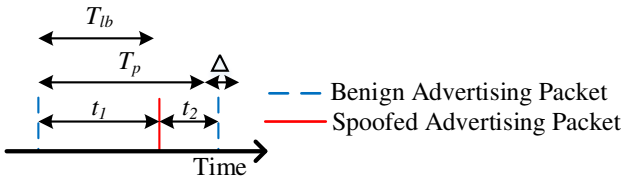

Figure 16: Illustration for the INT inspection analysis.

of the device and $\Delta = 10$ms. In the spoofing attack, when the device and the attacker broadcast advertising packets simultaneously, the monitor can observe a spoofed advertising packet between two consecutive benign advertising packets (Figure 16). We note that the original INT is divided into two observed INT values, t_1 and t_2. This spoofing attack can be detected if at least one of t_1 and t_2 is smaller than T_{lb}. Clearly, when $t_1 < T_{lb}$, the attack can be detected. When $t_1 \geq T_{lb}$, the INT inspection can miss the attack if $t_2 \geq T_{lb}$. Such a false negative can be produced under the following condition.

$$
\begin{aligned}
& t_1 + t_2 \leq T_p + \Delta \\
\Longrightarrow \quad & 2T_{lb} \leq T_p + \Delta \quad (\text{since } T_{lb} \leq t_1 \text{ and } T_{lb} \leq t_2) \\
\Longrightarrow \quad & 2T_p - 2\Delta \leq T_p + \Delta \quad (\text{since } T_p - \Delta \leq T_{lb}) \\
\Longrightarrow \quad & T_p \leq 3\Delta
\end{aligned}
$$

The above expression implies that when $\Delta = 10$ ms, the attacker can bypass the INT inspection if $T_p \leq 30$ms. We highlight that as per the BLE specification (p.1322 in [5]), $T_p \geq 20$ms. In our experiments (Table 2), since the minimum value of T_p among the nine test devices is 100ms, we do not observe such a false negative.

However, the packet loss rate in the BLE network might affect the false negative of the INT inspection in BlueShield. Specifically, the INT inspection may fail to detect an anomaly in the advertising interval when the collector does not receive the benign advertising packet due to extensive interference, but it receives the spoofed advertising packet from the attacker. In this scenario, when the packet loss rate of the benign advertising packet is denoted by ρ, the probability for the attacker to transmit n spoofed advertising packets without being detected by the INT inspection is ρ^n. Hence, the resulting FN value is negligible. For instance, when $\rho = 10\%$ (which is the typical packet loss rate observed in our experimental setup as shown in Table 6), the probability to transmit $n = 6$ spoofed advertising packets (in two advertising events) without being detected is 0.0001%. We highlight that six is an extremely small number of packets for the spoofing attack to succeed in a real-world environment.

Table 6: Comparison of BlueShield's performance when monitoring 1 BLE device and 9 BLE devices.

Device ID	Packet Loss (%)		Packet Delay (ms)	
	1 device	9 devices	1 device	9 devices
1	7.50	9.35	11.43	10.42
2	20.46	15.79	11.95	11.25
3	6.00	5.79	11.38	10.71
4	6.35	8.81	10.32	11.16
5	15.12	14.03	11.11	10.66
6	8.07	8.93	6.86	8.57
7	4.57	6.29	10.15	11.51
8	8.70	12.73	10.73	11.29
9	8.00	10.15	11.10	10.69
Average	**9.42**	**10.21**	**10.56**	**10.70**

D Impact of Multiple Device Monitoring

We evaluate the effectiveness of BlueShield in monitoring multiple BLE devices in terms of two metrics: (1) *packet loss rate* defined as the ratio of the number of packets received by collectors and the number of packets transmitted by the BLE device, and (2) *packet delay* defined as the time required by the monitor to retrieve a packet from a collector and inspect its features. As can be seen from Table 6, our results indicate that there is no significant difference in the average packet loss rate (9.42% vs. 10.21%) and the average packet delay (10.56ms vs. 10.70ms) between monitoring one BLE device and 9 BLE devices, respectively. Further, recall that in our experiments, BlueShield readily discovered 30 BLE/Bluetooth equipped devices in our office environment. This indicates that BlueShield can effectively monitor at least 30 BLE devices, which covers most BLE usage scenarios.