Securing Billion Bluetooth Low Energy Devices Using Cyber-Physical Analysis and Deep Learning Techniques

Hanlin Cai[1,3*], Yuchen Fang[1*], Jiacheng Huang[1], Honglin Liao[1], Meng Yuan[2], Zhezhuang Xu[3†]
[1]National University of Ireland, Maynooth, [2]Chalmers University of Technology, [3]Fuzhou University
{hanlin.cai, yuchen.fang, jiacheng.huang, honglin.liao}.2021@mumail.ie, meng.yuan@chalmers.se, zzxu@fzu.edu.cn

ABSTRACT

Bluetooth Low Energy (BLE) serves as a critical protocol for low-energy communication, playing a vital role in various sectors including industry, healthcare, and home automation. Despite its widespread adoption, inherent security limitations and firmware vulnerabilities expose BLE to significant risks, notably from spoofing attacks that threaten device integrity and data privacy. Addressing this challenge, this paper introduces ***BLEGuard***, a hybrid detection mechanism specifically designed to identify spoofing attacks within BLE networks. BLEGuard integrates pre-detection scheme, reconstruction techniques, and classification models to effectively detect advanced spoofing threats. To refine and validate BLEGuard system, this paper established a physical Bluetooth testbed to simulate attacks and generated a large-scale BLE Spoofing Attack Dataset (***BLE-SAD***) with over 1.3 million network packets. The experimental results demonstrate a high detection accuracy rate of 99.02%, with a false alarm rate of 2.04% and an un-detection rate of 0.37%. These findings highlight BLEGuard's effectiveness in enhancing the security of BLE networks, proving its potential as a robust solution to safeguard against sophisticated cyber threats in real-world applications.

CCS CONCEPTS

• **Security and privacy** → *Mobile and wireless security*; *Security services*; • **Computing methodologies** → *Machine learning*.

KEYWORDS

Bluetooth Low Energy, Cyber-physical Systems, Security and Privacy, Time Series Anomaly Detection, Data Mining

ACM Reference Format:
Hanlin Cai[1,3*], Yuchen Fang[1*], Jiacheng Huang[1], Honglin Liao[1], Meng Yuan[2], Zhezhuang Xu[3†]. 2024. Securing Billion Bluetooth Low Energy Devices Using Cyber-Physical Analysis and Deep Learning Techniques. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (KDD-UC'24)*. ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

KDD-UC'24, Aug 25-29, 2024, Barcelona, Spain

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Named after the Viking King Harald Bluetooth, who was known for his role in unifying Danish tribes, Bluetooth technology has become a ubiquitous standard for short-range wireless communications. Since its inception, Bluetooth has revolutionized the way devices interact in close proximity [7]. The advent of the Bluetooth Low Energy (BLE) standard has further solidified its dominance, especially in the burgeoning era of the Internet of Things (IoT) and the emerging technologies of 6G communications [14]. BLE's low power requirements and high functionality make it an ideal choice for a multitude of IoT applications ranging from industrial automation to health monitoring, ensuring seamless connectivity between billions of devices. By 2027, the deployment of BLE devices is anticipated to burgeon to an astonishing 7.5 billion [2].

This exponential adoption, however, is overshadowed by significant security challenges within the BLE networks. BLE-enabled devices are prone to a diverse array of sophisticated attacks due to inherent I/O limitations and firmware vulnerabilities. These threats include zero-day exploits, where attackers exploit undisclosed vulnerabilities [16], DDoS (Distributed Denial of Service) attacks that cripple network services [6], and particularly spoofing attacks [18].

Spoofing attacks are alarmingly prevalent and concerning due to their low initiation costs and minimal hardware requirements, making them a preferred tactic among attackers. In these attacks, perpetrators impersonate legitimate devices, misleading network participants to intercept or manipulate sensitive data [22]. This undermines the integrity and confidentiality of BLE systems, facilitating unauthorized access and data breaches. The ease and low cost of initiating these attacks underscore the urgent need for the development of advanced detection mechanisms. These mechanisms must be capable of identifying and mitigating the sophisticated tactics used in spoofing attacks, thereby enhancing the security posture of BLE networks against these pervasive threats [19].

To combat these security threats, an out-of-the-box monitoring system has been introduced, leveraging BLE's cyber-physical features to fortify defenses against spoofing attackers [17]. Additionally, various research initiatives employ machine learning techniques to detect anomalous patterns within BLE network traffic. A particularly promising learning framework that integrates reconstruction and classification models has been developed to identify network packets as either benign or malicious with remarkable precision [10].

Unfortunately, most existing methods grapple with the significant challenge of harmonizing detection accuracy, false positive rates, and resource utilization. This delicate balance severely restricts their applicability across a broader spectrum of real-world

[*]Both authors contributed equally to this research.
[†]Corresponding author.

```
Packet Number: 8942
Timestamp: 2023-04-18 17:45:30.654321
Channel: 39 (Used Channel Number)
Source MAC: 1a:2b:3c:4d:5e:6f (Device MAC Address)
Destination MAC: 6f:5e:4d:3c:2b:1a (Central Device MAC Address)
Advertising Interval: 400ms (Time between consecutive advertising packets)
RSSI: -54 dBm (Received signal strength indicator)
Carrier Frequency Offset: +2 KHz (Difference from the carrier frequency)
PDU Length: 31 bytes (Length of the protocol data unit)
Payload Data:
  Opcode: 0x1c (ATT Read Request)
  Handle: 0x0040 (Characteristic handle for Battery Level)
  Value: 85% (Battery Level measurement value)
  CRC: 0xDEADBEEF (Cyclic Redundancy Check for error-checking)
```

Figure 1: Sample data of a typical BLE network packet.

scenarios [18, 23]. There is a pressing need for a more adaptable and efficient solution, which can uphold stringent detection standards while effectively managing resource constraints. Such an innovation would significantly broaden the utility of security frameworks, extending their deployment across a wider variety of environments and devices. This expansion is crucial for bolstering defenses against spoofing attacks in increasingly diverse and resource-constrained settings [20].

Therefore, this work aims to introduce a novel detection mechanism that leverages cyber-physical analysis and deep learning techniques. Specifically engineered to detect sophisticated spoofing attacks, this mechanism combines extensive offline training with critical real-time online analysis. In pursuit of this goal, we established a tangible BLE network system for conducting attack simulations and compiling a large-scale network dataset. This broad and verifiable dataset is crucial for advancing research within the domain and ensuring the robustness of our findings. A series of experiments utilizing diverse datasets will be conducted to test the viability of the detection mechanism proposed. Subsequent to these tests, a meticulous assessment of the experimental results will be performed, and their profound implications for real-world applications will be analyzed. Overall, our contributions are threefold:

- Development of the ***BLE-SAD*** dataset, which includes around 906,000 packets, tailored specifically for the training and evaluation of our models.
- Design and empirical validation of ***BLEGuard***, which is proposed for effective detection of spoofing attacks.
- Integration capabilities of ***BLEGuard*** within BLE networks, designed to ensure effective detection without disrupting existing network operations or depleting network resources.

2 PRELIMINARIES

2.1 Basics of Bluetooth Low Energy

Bluetooth Low Energy (BLE) is often the technology of choice for networks where energy-efficient and cost-effective communication is paramount. This is especially common with low-cost, energy-constrained devices like temperature sensors that capture specific data attributes and wirelessly transmit this information to user devices, like smartphones. BLE operates using three dedicated radio frequency channels (37, 38, and 39) for advertising, which is the process of broadcasting the presence of a BLE device to initiate a connection [7]. These are known as the advertising channels. Once a connection is established, the remaining channels, known as data channels, are used for ongoing communication between devices.

The typical communication protocol in a BLE network encompasses four main stages: advertising, connecting, pairing, and data accessing [17]. The advertising stage is where the BLE device announces its availability to connect. In the connecting phase, a user device responds to this advertisement, establishing a bidirectional link. Pairing is the next crucial step, where security credentials are exchanged, forming the foundation for a secure communication. Finally, in the data accessing stage, the authenticated user device is able to read or write the data from or to the BLE device. **Figure 1** shows a typical network packet during the BLE communication, which includes data with time-series features such as packet number, timestamp, advertising interval and payload data.

2.2 Spoofing Attacks in BLE Networks

The spoofing attack is a type of cybersecurity attack wherein an attacker impersonates a legitimate BLE device or network entity [22]. In such attacks, the perpetrator typically masquerades as a trusted BLE device using forged information, such as a spoofed MAC address or other identifying details, as illustrated in **Figure 2 (a)**. In the context of a spoofing attack, the cyber-physical features of the BLE network are notably impacted, leading to significant deviations from typical benign scenarios. For instance, an anomalous shift in the RSSI (Received Signal Strength Indicator) values of the advertising packets can signal the presence of a spoofing attack, as depicted in **Figure 2 (b)**. These deviations provide critical indicators that can be used to effectively identify potential malicious activity [4].

Given the unique characteristics of BLE networks, this paper has identified and utilized four key cyber-physical features to enhance the detection algorithm and to facilitate the training of learning models:

- Used Channel Numbers (UCN): These denote the specific data channels employed during the transmission of BLE packets, crucial for analyzing communication patterns.
- Advertising Interval (INT): This measures the temporal interval between consecutive packets transmitted on the same advertising channel, vital for detecting timing anomalies.
- Received Signal Strength Indicator (RSSI): This feature represents the signal-to-noise ratio gleaned from packet exchanges, providing insights into the physical layer connectivity.
- Carrier Frequency Offset (CFO): Refers to the discrepancy between the expected and the actual carrier frequencies used in BLE communications, indicating potential frequency drifts or unauthorized channel usage.

2.3 Current Security Challenges

The BLE specifications [1] provide a range of authentication mechanisms theoretically designed to prevent spoofing attacks. However, these mechanisms often fail to achieve their intended purpose in practice due to three main reasons:

(1) **Limited Device I/O Capabilities:** A significant number of BLE devices have limited I/O capabilities, which precludes them from utilizing any robust authentication mechanisms. It is not surprising that recent research has shown that over 80% of current BLE

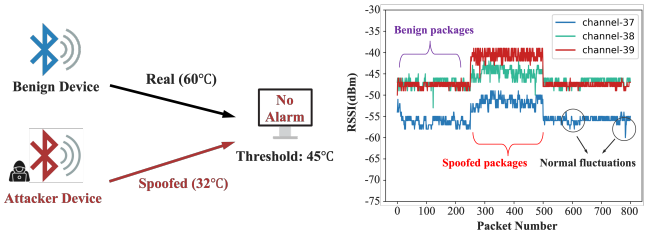

Figure 2: (a) Spoofing attack in BLE sensor network and (b) observed RSSI values during attack simulation.

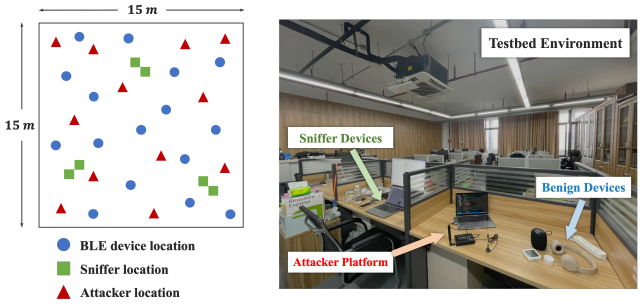

Figure 3: Locations of devices in proposed BLE testbed.

devices communicate with user devices in plaintext without any form of authentication [3].

(2) **Persistent Security Vulnerabilities:** For BLE devices that do implement various security measures, there are still numerous attack vectors at both the protocol level and application level that malicious actors can exploit to conduct spoofing attacks [17].

(3) **Insufficient User Awareness:** Users of BLE devices may lack awareness or the technical knowledge required to enable and configure security features properly, leading to increased susceptibility to spoofing attacks [5].

Additionally, the challenge of implementing software-based solutions (i.e., firmware updates for BLE devices or software patches on user devices) to these security vulnerabilities is compounded by four major practical challenges:

(1) **Ineffectiveness Against Zero-Day Exploits:** The nature of software patches does not allow them to preemptively protect against zero-day vulnerabilities, which can be immediately exploited by attackers upon discovery [16].

(2) **Fragmented Update Ecosystem:** The diversity in BLE device manufacturers leads to a fragmented ecosystem for firmware updates, which complicates the process of applying uniform security patches across devices.

(3) **Legacy Device Constraints:** A considerable number of legacy BLE devices in use are incapable of being updated due to outdated I/O capabilities, leaving them vulnerable to new exploits.

(4) **Resource Constraints for Update Dissemination:** Many manufacturers of BLE devices may face resource constraints that impede the timely development and distribution of necessary firmware updates, further exacerbating security challenges.

3 DATASETS BUILDING

3.1 Testbed Implementation

The testbed environment can be categorized into four parts: (i) BLE devices, (ii) user devices, (iii) attacker platforms, and (iv) network sniffers. **Table 1** comprehensively illustrates all the components utilized in the network testbed. The testbed was strategically deployed within a physical environment: a $15m \times 15m$ office space configured with 18 cubicles, as illustrated in **Figure 3**. The office was methodically partitioned into $1m \times 1m$ grids. This setting typifies a complex and acoustically active indoor environment, presenting significant challenges for evaluating the detection efficiency.

Table 1: Components of proposed BLE network testbed.

Component	Description
BLE devices	To build the BLE network testbed.
User devices	To simulate normal usage scenarios.
Attacker platforms	To launch spoofing attacks.
Network sniffers	To capture network packets.

3.2 BLE-SAD Dataset

Regarding the building of our dataset, we collected normal advertising packets from each BLE device over a period of approximately eight hours (five hours during daytime and three hours at nighttime). Additionally, for each attacker platform situated in various positions, malicious packets were collected for about 20 minutes. Our BLE Spoofing Attack Dataset (***BLE-SAD***) contains 1,304,000 advertising packets, of which 82.4% are benign and 16.6% are malicious. The open-source data can be accessed at the appendix.

4 DETECTION MECHANISM

In this section, we will discuss our proposed BLEGuard system, a hybrid detection mechanism combined cyber-physical analysis with deep learning techniques.

4.1 Pre-detection Scheme

In BLEGuard, suspicious activities are identified through detection of atypical fluctuations in cyber-physical features such as Used Channel Numbers (UCN), Advertising Interval (INT), Carrier Frequency Offset (CFO), and Received Signal Strength Indicator (RSSI). Abrupt changes in UCN and INT indicate potential security threats, while RSSI and CFO are crucial for a continuous pre-detection mechanism that anticipates advanced spoofing attacks.

To effectively monitor these indicators, BLEGuard employs three network sniffers that capture the values of these features within a *lookback window*. The lookback window refers to a predefined period prior to the current analysis point, during which data is collected to establish a baseline for normal behavior. This historical data is essential for understanding typical network conditions and variations. Subsequently, the system evaluates the current network activity by examining the values from an *observation window*, which is the period immediately following the lookback window. This

approach allows BLEGuard to compare present data against the baseline to spot any irregularities or deviations.

An alarm is triggered if there are deviations from the established norms in any of the monitored features, indicating a potential security breach. This method can be seamlessly integrated into existing BLE networks without causing disruption or significant resource consumption. Detailed detection schemes for each feature are outlined as follows:

• Metric 1: Used Channel Numbers

In BLE networks, Used Channel Numbers (UCN) designates the sequence of radio channels that BLE devices utilize for transmission, adhering to a preconfigured pattern to enhance connectivity and reduce noise interference. The stability of UCN patterns can be compromised during spoofing attacks, as attackers may instigate an irregular shift in the communication channels, thus disrupting the network's harmonious channel utilization. To quantify such fluctuations, we introduce the metric UCN_{change}, which represents the cumulative measure of channel switching activity:

$$UCN_{\text{change}} = \sum_{i=1}^{N_{\text{obs}}} |UCN_i - UCN_{i-1}|, \tag{1}$$

where N_{obs} is the count of observed transmission packets and UCN_i corresponds to the utilized channel for the i^{th} packet transmission. An elevated UCN_{change} value is indicative of more frequent channel alternations, potentially signaling an ongoing spoofing attack. For operational integrity in BLE networks, an acceptable threshold for UCN_{change}, denoted by $\Delta UCN_{\text{normal}}$, is set at 2.8. This threshold indicates the maximum allowable frequency of channel changes within a defined observation period. A breach of this threshold is symptomatic of anomalous behavior:

$$\text{If } UCN_{\text{change}} > \Delta UCN_{\text{normal}}, \text{ activate further detection.} \tag{2}$$

Employing UCN_{change} as a heuristic enables a robust security framework capable of detecting and responding to potential spoofing threats, thereby fortifying the BLE network's defenses.

• Metric 2: Advertising Interval The Advertising Interval (INT) is also a key parameter in BLE communications, defining the time gap between consecutive advertising packets. This interval is crucial for maintaining the orderly transmission of broadcast information in BLE networks. By definition, the INT between any two consecutive advertising packets should never fall below a predefined lower bound, which is set based on the specifications of the BLE device and the operational requirements of the network. This lower bound is denoted as L_{int}. The formula used to compute the runtime INT value, INT, for the interval between two packets is given by:

$$INT = T_{\text{current}} - T_{\text{previous}}, \tag{3}$$

where T_{current} is the timestamp of the current advertising packet, and T_{previous} is the timestamp of the immediately preceding advertising packet. The Advertising Interval (INT) is calculated as the difference between these two timestamps. If INT is found to be less than the predefined lower limit L_{int}, the monitor identifies this condition as anomalous. Such a scenario indicates a potential operational fault or a security breach, such as a spoofing attack that attempts to flood the network with frequent, unauthorized advertising packets. Upon detecting such an anomaly, the monitor triggers an alarm, alerting the system to the potential threat. Typically, L_{int} is set to a threshold value of 10 milliseconds to detect rapid, unscheduled transmissions [17]. The corresponding condition can be mathematically expressed as:

$$\text{If } INT < L_{\text{int}}, \text{ activate further detection.} \tag{4}$$

This monitoring mechanism ensures the integrity and correct functioning of the BLE network by verifying that the advertising packets are transmitted within the expected intervals, adhering to the designed operational parameters.

• Metric 3: CFO level

BLEGuard continuously monitors the CFO (Carrier Frequency Offset) and RSSI (Received Signal Strength Indicator) values from advertising packets. Upon activation of the CFO and RSSI inspection, BLEGuard analyzes these values through the following procedure. For a BLE device exhibiting intermittent advertising patterns, we define the lookback window as the time period T_l (with N_l packets) before the transition from advertising to connection state, and the observation window as the time period T_o (with N_o packets) after the transition from connection back to advertising state. In BLEGuard, following the reception of a connection request packet, the monitoring system initiates the CFO and RSSI inspections for advertising packets collected from each device across the three advertising channels (37, 38 and 39). The system first calculates the acceptable ranges for CFO and RSSI values using data from the lookback window. It then evaluates these metrics in the advertising packets during the observation window. If an anomaly is detected in either the CFO or RSSI readings, an alarm is triggered.

The CFO values observed from BLE networks are expected to conform to a Gaussian distribution [17]. Consequently, when μ_0 and σ_0 represent the mean and standard deviation of these CFO values, the probability distribution function for the CFO can be articulated as:

$$F_{cfo}(x_i) = \frac{1}{\sigma_0\sqrt{2\pi}} \cdot e^{-\frac{(x_i-\mu_0)^2}{2\sigma_0^2}} \tag{5}$$

where x_i denotes a sample CFO value. In BLEGuard, the monitor employs the CFO values from advertising packets within a lookback window, comprising N_l packets, to calculate μ_0 and σ_0. These parameters are then integrated into the probability function previously mentioned. If the advertising packets from both the lookback and subsequent observation windows originate from the same BLE device, the CFO values from the observation window's advertising packets should statistically align with the given distribution. This is verified by the monitoring system calculating the negative log-likelihood of the CFO values from the observation window packets, defined as:

$$L_{cfo} = \frac{1}{N_o} \sum_{i=1}^{N_o} -\log F_{cfo}(x_i) \tag{6}$$

If the log-likelihood value is less than a predetermined CFO inspection threshold, denoted by β_{cfo} (i.e., $L_{cfo} < \beta_{cfo}$), the CFO values are considered to be within the normal range for the BLE device. This threshold β_{cfo} is a tunable parameter within BLEGuard that dictates the permissible range of CFO values during the observation window. In contrast, if the log-likelihood value exceeds β_{cfo} (i.e., $L_{cfo} > \beta_{cfo}$), an anomaly is recognized, and an alarm is

activated, signaling a possible spoofing attack. In our real-world tests, the optimal setting for parameter β_{cfo} is 3.0 [17].

• **Metric 4: RSSI level**

To detect anomalies in RSSI values amid strong signal reflections in BLE networks, we utilize a two-component Gaussian mixture model. This approach is chosen because RSSI values in environments with high noise can be effectively modeled using two normal distributions [15]. The probability distribution function for RSSI values is given by:

$$F_{rssi}(y_i) = w \cdot \frac{1}{\sigma_1\sqrt{2\pi}} \cdot e^{-\frac{(y_i-\mu_1)^2}{2\sigma_1^2}} + (1-w) \cdot \frac{1}{\sigma_2\sqrt{2\pi}} \cdot e^{-\frac{(y_i-\mu_2)^2}{2\sigma_2^2}} \quad (7)$$

In this equation, μ_1 and μ_2 are the means of the two components, σ_1 and σ_2 are their standard deviations, w is a weight parameter that balances the two components (usually set to 0.56), and y_i is a RSSI sample data. Using the BLEGuard system, N_l of RSSI values from advertising packets within a lookback window are analyzed to estimate the parameters $\mu_1, \mu_2, \sigma_1, \sigma_2$, and w through a conventional expectation-maximization (EM) algorithm [8]. Following this, the monitor calculates the negative log-likelihood that the RSSI values $(y_i, \forall i \in [1, N_o])$ from the observation window conform to the model specified by Equation (7) to:

$$L_{rssi} = \frac{1}{N_o} \sum_{i=1}^{N_o} -\log F_{rssi}(y_i) \quad (8)$$

An anomaly is detected when the negative log-likelihood surpasses a predefined RSSI inspection threshold, denoted as δ_{rssi} (i.e., $L_{\text{rssi}} > \delta_{\text{rssi}}$). The threshold δ_{rssi} is a crucial parameter in BLEGuard, calibrated to optimize detection sensitivity and specificity. Typically, δ_{rssi} is set to 5.0. The details of model parameter setting will be discussed in Section 5.2.

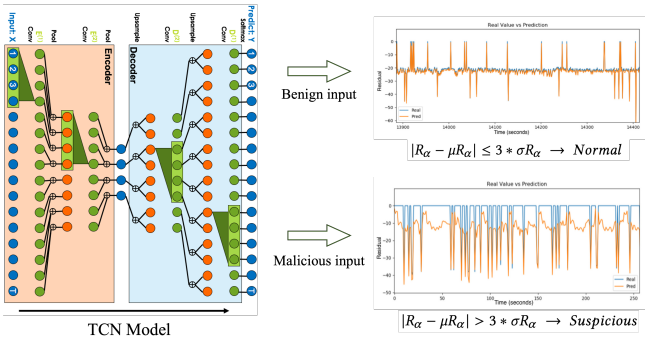

Figure 4: The operational mechanism of temporal convolutional network.

4.2 Reconstruction Model

Upon identifying suspicious activities, a thorough analysis is initiated on anomalous data batches. To facilitate this, a Temporal Convolutional Network (TCN) [11], as illustrated in **Figure 4**, is employed to reconstruct traffic patterns. This approach helps isolate aberrant data through comparative analysis. A TCN is a type of neural network specifically designed for sequence modeling that combines convolutional layers with causal connections to ensure that predictions for a specific time step can only depend on past data. This structure makes TCNs particularly effective for time series predictions where past context is crucial.

During the offline training phase, the objective is to minimize the discrepancy between the learned data D_L and the original dataset D_T. In the online testing phase, the presence of malicious packets in the input data triggers an increase in the reconstruction error, indicative of potential spoofing threats. The residual, defined as $R(D_T, D_L) = |D_T - D_L|$ with $D_L = f(D_T)$, where f represents the transformation function employed by the TCN auto-encoder, serves as a critical metric. This residual is assessed to calculate the anomaly score α [10] for each data batch, as depicted in Equation (9). Here, R_α denotes the calculated residual, μ is the mean value of the residual, and σ is its standard deviation.

$$\alpha = \begin{cases} 0, \ when \ |R_\alpha - \mu R_\alpha| \leq 3 * \sigma R_\alpha \ \rightarrow \ Normal \\ 1, \ when \ |R_\alpha - \mu R_\alpha| > 3 * \sigma R_\alpha \ \rightarrow \ Suspicious \end{cases} \quad (9)$$

4.3 Classification Models

Following the identification of suspicious data batches, the next step involves classifying these packets into two categories: benign or malicious. In this research, a text-convolutional neural network (text-CNN) [13] is employed for the extraction of traffic features. Text-CNNs are specialized types of convolutional neural networks designed to handle text data. They apply convolutional layers to extract higher-level features from text data structured as input vectors, making them highly effective for tasks involving natural language processing and text analysis.

For packet classification, this work employs four cost-efficient classifiers: Support Vector Machine (SVM) [9], K-Nearest Neighbors (KNN) [10], Random Forest (RF) [13], and NaÃŕve Bayes [12]. This multi-classifier approach helps to mitigate potential biases in text analysis by diversifying the analytical perspectives. Network payload-based features are generated by converting the payload bytes into low-dimensional vectors using *Word2Vec* techniques, which effectively capture the semantic relationships within the data. These vectors form the input for the text-CNN, where key traffic features are extracted. The features extracted by the text-CNN are then concatenated with statistical features to create a comprehensive feature set for the final classification models.

4.4 System Overview

BLEGuard is designed to optimize the balance between detection accuracy and power consumption. As depicted in **Figure 5**, the system employs a flexible approach where the pre-detection algorithm is utilized to maintain efficiency under computing resource constraints, minimizing power and computational overhead. In scenarios where high detection accuracy is paramount, the reconstruction model is activated to enhance analytical precision. Moreover, the classification models within BLEGuard are adept at precisely identifying malicious advertising packets, providing targeted feedback that significantly augments the efficacy of the detection modules. This versatile framework ensures that BLEGuard can adapt to varying operational demands, thereby maintaining robust security measures without compromising on network performance.

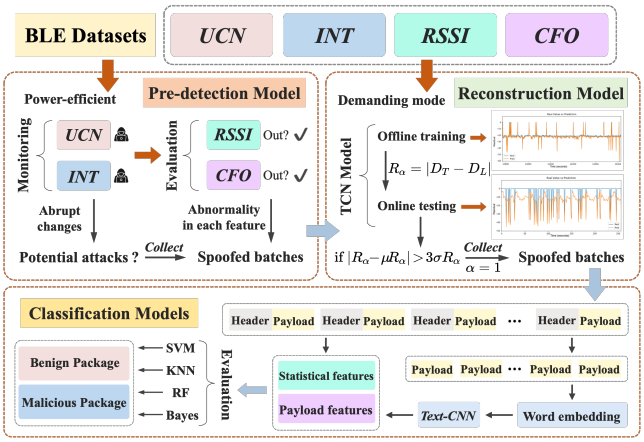

Figure 5: Overall workflow of detection mechanism.

5 EXPERIMENTAL RESULTS

This section describes the experimental framework used to evaluate the efficacy of the BLEGuard system. The settings are carefully designed to mimic realistic scenarios in which BLE networks operate, ensuring that the results are both robust and applicable to real-world applications.

5.1 Experimental Details

The BLE-SAD dataset was compiled from nine distinct BLE devices, with their information detailed in the appendix. The dataset was divided into training and testing sets at a ratio of 8.5 to 1.5, comprising 962,850 effective BLE network packets for training and 169,920 for testing, respectively. The model training was conducted using an Intel Core i5-13600 CPU processor (3.50 GHz) with 32GB of RAM and an NVIDIA GeForce RTX 4060 Ti GPU equipped with 24GB of memory. The algorithms were implemented in Python 3.8, utilizing the PyTorch 1.8.1 framework.

5.2 Parameter Settings

In BLEGuard's Pre-detection Scheme, four crucial parameters (L_{int}, $\Delta UCN_{\text{normal}}$, β_{cfo}, and δ_{rssi}) are carefully configured within specific ranges to maximize detection accuracy, as summarized in **Table 2**. These settings are the result of comprehensive testing and fine-tuning, ensuring that BLEGuard efficiently and reliably identifies spoofing attacks within BLE networks. Addtionally, the hyperparameter of learning models are given in the appendix.

5.3 Overall Performance Evaluation

During the evaluation phase, three key metrics are employed to assess the effectiveness of our proposed methods. Accuracy, defined as the overall proportion of correctly classified instances, serves as a fundamental measure of the model's capability to accurately differentiate between benign and malicious packets. This metric is critical in evaluating the overall efficacy of the detection system. The False Alarm Rate (FAR) quantifies how often BLEGuard erroneously activates an alert when processing benign advertising packets from legitimate BLE devices, reflecting the model's precision. Conversely, the Un-detection Rate (UND) measures the frequency with which BLEGuard fails to identify a spoofing attack, highlighting potential vulnerabilities in detecting sophisticated threats.

BLEGuard's performance evaluation is conducted on a robust and imbalanced dataset collected from nine different BLE devices, such as Xiaomi sensors, Apple HomePod, and Dell speakers. The devices and the corresponding evaluation results are comprehensively detailed in **Table 3**. The table presents the performance metrics for each device, including the accuracy, FAR, and UND, thus providing a granular view of the system's effectiveness across varied hardware configurations. The experimental data reveals BLEGuard's impressive detection capabilities, achieving an exemplary average accuracy of 99.02%, complemented by a low false alarm rate of 2.04% and an un-detection rate of 0.37%. These statistics not only validate the robustness of BLEGuard but also illustrate its adaptability and reliability in diverse operational environments.

As shown in **Table 4**, compared with MARC framework [21] and BlueShield system [17], BLEGuard offers higher accuracy in identifying spoofing attack while maintaining faster response times. By integrating cyber-physical analysis with deep learning techniques, BLEGuard achieves approximately a 15% improvement in false alarm rate and nearly a 50% improvement in un-detection rate, thereby enhancing overall accuracy. In addition, we aim to achieve improved detection performance and better response time through adjustments to model hyperparameters.

6 CONCLUSION AND ONGOING WORK

In this paper, we proposed the BLEGuard system, a novel hybrid detection mechanism designed to safeguard Bluetooth Low Energy (BLE) networks against sophisticated spoofing attacks. BLEGuard's unique integration of a pre-detection scheme, reconstruction techniques, and classification models enables it to effectively identify and neutralize threats, thereby enhancing network security. The system's high detection accuracy, combined with a low false alarm rate and un-detection rate, underscores its potential not only as a specialized tool for BLE security but also for broader applications in industry, healthcare, and smart home sectors. The practical application of BLEGuard in these sectors can significantly mitigate risks associated with the inherent security vulnerabilities of BLE technologies, providing a reliable security solution that aligns with the needs of modern connected environments.

The project will incorporate additional datasets encompassing real-world low-power Bluetooth usage scenarios, thereby expanding the scale of the BLE-SAD dataset. Furthermore, we will explore the application of more advanced models for data extraction and threat assessment, aiming to enhance assessment speed and reduce system power consumption. These advancements will not only strengthen the security of BLE networks but also pave the way for next-generation protection methodology in the evolving landscape of digital communication technologies.

ACKNOWLEDGMENTS

This project was supported by the Chinese National Undergraduate Innovation Training Program (No. 202310386056) and the AAAI 2024 Undergraduate Consortium Scholarship. We extend our sincere gratitude to Dr. Tozammel Hossain, Dr. Jason Grant, Dr. Patricia OrdÃşÃśez, and Ms. Linshi Li for their insightful suggestions.

Table 2: Optimal Parameter Settings for the Pre-detection Scheme.

Network Features	Parameter	Testing Range	Optimal Setting
Used Channel Numbers (UCN)	$\Delta UCN_{\text{normal}}$	(2.0, 5.0)	**2.8**
Advertising Interval (INT)	L_{int}	(5.0, 20.0) ms	**10.0 ms**
Carrier Frequency Offset (CFO)	β_{cfo}	(1.0, 5.0)	**3.0**
Received Signal Strength Indicator (RSSI)	δ_{rssi}	(3.0, 10.0)	**5.0**

Table 3: Detection performance of *BLEGuard* mechanism.

ID	Device (Number)	Accuracy	FAR	UND	Response Time (s)
1	Xiaomi Sensor (*3)	98.92%	2.23%	0.43%	1.19
2	Xiaomi Locker (*2)	99.11%	2.04%	0.32%	1.37
3	Xiaomi Speaker (*2)	98.93%	1.84%	0.36%	2.49
4	Apple HomePod (*1)	99.04%	2.11%	0.34%	2.54
5	Dell Speaker (*1)	99.21%	2.51%	0.17%	1.91
6	Lenovo Speaker (*1)	98.71%	1.81%	0.76%	2.89
7	August Smart Lock (*2)	99.00%	2.43%	0.19%	2.63
8	Nutale Key Finder (*2)	99.05%	1.45%	0.52%	2.11
9	Nordic nRF52 DK (*2)	99.20%	1.96%	0.22%	1.59
Overall		**99.02%**	**2.04%**	**0.37%**	**2.08**

Table 4: Performance comparison.

Method	FAR	UND	Accuracy	Response Time
MARC [21]	7.28%	5.71%	92.64%	8.79s
BlueShield [17]	2.37%	0.73%	98.67%	3.46s
BLEGuard (us)	**2.04%**	**0.37%**	**99.02%**	**2.08s**

REFERENCES

[1] Bluetooth-SIG. Accessed on: 20 April 2024. Bluetooth Core Specification 5.4. https://www.bluetooth.com/specifications/.

[2] Bluetooth-SIG. Accessed on: 20 April 2024. Bluetooth Market Update. https://bluetooth.com/2024-market-update/.

[3] Bluetooth-SIG. Accessed on: 20 April 2024. Security in Bluetooth Specifications. https://www.bluetooth.com/learn-about-bluetooth/key-attributes/bluetooth-security/.

[4] Hanlin Cai. 2024. Securing Billion Bluetooth Devices Leveraging Learning-Based Techniques. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 23731–23732.

[5] Matthias Cäsar, Tobias Pawelke, Jan Steffan, and Gabriel Terhorst. 2022. A survey on Bluetooth Low Energy security and privacy. *Computer Networks* 205 (2022), 108712.

[6] Shane Ditton, Ali Tekeoglu, Korkut Bekiroglu, and Seshadhri Srinivasan. 2020. A proof of concept denial of service attack against bluetooth iot devices. In *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 1–6.

[7] Carles Gomez, Joaquim Oller, and Josep Paradells. 2012. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *sensors* 12, 9 (2012), 11734–11753.

[8] Xiansheng Guo, Lin Li, Feng Xu, and Nirwan Ansari. 2018. Expectation maximization indoor localization utilizing supporting set for Internet of Things. *IEEE Internet of Things Journal* 6, 2 (2018), 2573–2582.

[9] Christiana Ioannou and Vasos Vassiliou. 2021. Network attack classification in IoT using support vector machines. *Journal of sensor and actuator networks* 10, 3 (2021), 58.

[10] Abdelkader Lahmadi, Alexis Duque, Nathan Heraief, and Julien Francq. 2020. MitM attack detection in BLE networks using reconstruction and classification machine learning techniques. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 149–164.

[11] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. 2017. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

[12] Amjad Mehmood, Mithun Mukherjee, Syed Hassan Ahmed, Houbing Song, and Khalid Mahmood Malik. 2018. NBC-MAIDS: Naive Bayesian classification technique in multi-agent system-enriched IDS for securing IoT against DDoS attacks. *The Journal of Supercomputing* 74 (2018), 5156–5170.

[13] Erxue Min, Jun Long, Qiang Liu, Jianjing Cui, and Wei Chen. 2018. TR-IDS: Anomaly-based intrusion detection through text-convolutional neural network and random forest. *Security and Communication Networks* (2018).

[14] Sushant Kumar Pattnaik, Soumya Ranjan Samal, Shuvabrata Bandopadhaya, Kaliprasanna Swain, Subhashree Choudhury, Jitendra Kumar Das, Albena Mihovska, and Vladimir Poulkov. 2022. Future wireless communication technology towards 6G IoT: An application-based analysis of IoT in real-time location monitoring of employees inside underground mines by using BLE. *Sensors* 22, 9 (2022), 3438.

[15] Yong Sheng, Keren Tan, Guanling Chen, David Kotz, and Andrew Campbell. 2008. Detecting 802.11 MAC layer spoofing using received signal strength. In *IEEE INFOCOM 2008-The 27th Conference on Computer Communications*. IEEE, 1768–1776.

[16] Ioannis Stellios, Panayiotis Kotzanikolaou, and Mihalis Psarakis. 2019. Advanced persistent threats and zero-day exploits in industrial Internet of Things. *Security and Privacy Trends in the Industrial Internet of Things* (2019), 47–68.

[17] Jianliang Wu, Yuhong Nan, Vireshwar Kumar, Mathias Payer, and Dongyan Xu. 2020. {BlueShield}: Detecting spoofing attacks in bluetooth low energy networks. In *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*.

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

[19] Jianliang Wu, Ruoyu Wu, Dongyan Xu, Dave Tian, and Antonio Bianchi. 2023. SoK: The Long Journey of Exploiting and Defending the Legacy of King Harald Bluetooth. In *2024 IEEE Symposium on Security and Privacy (S&P)*.

[20] Han Xu, Yaxin Li, Wei Jin, and Jiliang Tang. 2020. Adversarial attacks and defenses: Frontiers, advances and practice. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3541–3542.

[21] Muhammad Yaseen, Waseem Iqbal, Imran Rashid, Haider Abbas, Mujahid Mohsin, Kashif Saleem, and Yawar Abbas Bangash. 2019. Marc: A novel framework for detecting mitm attacks in ehealthcare ble systems. *Journal of medical systems* 43 (2019), 1–18.

[22] Pengfei Zhang, Sai Ganesh Nagarajan, and Ido Nevat. 2017. Secure location of things (SLOT): Mitigating localization spoofing attacks in the Internet of Things. *IEEE Internet of Things Journal* 4, 6 (2017), 2199–2206.

[23] Zaixi Zhang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 2022. Fldetector: Defending federated learning against model poisoning attacks via detecting malicious clients. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2545–2555.

A REPRODUCIBILITY

Here we describe the setup and implementation details of our experiments presented in Section 5. Implementations of ***BLEGuard*** algorithm, along with ***BLE-SAD*** dataset for reproducing experiments, can be found at https://github.com/BLEGuard/supplement

A.1 Testbed Details

The BLE-SAD is established in the following components,

- **BLE Devices:** A variety of commercial BLE devices, such as sensors, locks, and beacons, which represent a cross-section of typical endpoints found in BLE networks. These devices are instrumental in generating the benign traffic patterns for our datasets.
- **User Devices:** Smartphones, tablets, and computers used by end-users to interact with BLE devices. These devices are equipped with BLE capabilities to emulate regular user operations and activities within the network (Table 5).
- **Attacker Platforms:** These include custom-built software and modified hardware designed to simulate various security attacks on the BLE network, such as spoofing and denial of service (DoS) attacks. Tools in this category help test the robustness of the network's security measures (Table 6).
- **Network Sniffers:** Devices and software used to capture and analyze the traffic flowing through the BLE network. Examples include Wireshark for packet analysis and Ubertooth for specific BLE monitoring (Table 7).
- **Data Acquisition Systems:** These systems are configured to automatically record all network traffic, capturing essential metrics such as packet size, timing, and payload data. They are critical for gathering the raw data needed for further analysis.
- **Simulation Software:** Software tools that simulate network conditions and behaviors, which help in predicting network performance under various scenarios and in understanding potential network failures before they occur.

Table 5: User devices used in BLE testbed.

Device Name	Operation System
Google Pixel 7	Andriod 13
iPhone 13	iOS 16
Surface Laptop 5	Windows 11
MacBook Pro M1	MacOS 13.1
Lenovo V15-IIL	Windows 10 Pro
Dell 7050 PC	Windows 10 Pro

Table 6: Attacker platform used in BLE testbed.

Device Name	Operating Platform
Lenovo 15IIL Laptop	Mirage Software
CSR 4.0 BT dongle	Mirage Software
HM-10 development board	Ostinato Software
CYW920735 development board	Ostinato Software

Table 7: Network sniffer used in BLE testbed.

Communication Platform	Network Capture Tool
Raspberry Pi (Linux 5.4)	BLE-Analyzer-PRO
Raspberry Pi (Linux 5.4)	Ubertooth One
Google Pixel 7 (Anroid 13)	nRF Connect Software
Apple MacBook (MacOS 13.1)	nRF Connect Software

A.2 Baseline Machine Learning Models

SVM. The Support Vector Machine (SVM) is a widely used supervised learning model for classification and regression tasks. The primary objective of SVM is to find an optimal hyperplane that separates the data, clearly distinguishing between attack packets and normal packets.

KNN. K-Nearest Neighbors (KNN), a non-parametric regression method, was implemented with a leaf size of 50 to balance computational efficiency and prediction accuracy. This parameter optimizes the trade-off between resource use and predictive performance.

Random Forest. The Random Forest ensemble learning method was employed with a maximum tree depth of 50 and 250 estimators. This configuration was selected to achieve an optimal balance between model complexity and computational feasibility.

NaÃŕve Bayes. The NaÃŕve Bayes classifier was utilized for its efficiency and robust performance in high-dimensional datasets, assuming conditional independence between features.

A.3 Hyperparameters of Deep Learning Models

The hyperparameters of temporal convolutional network (TCN) and text-convolutional neural network (Text-CNN) are as follows:

Table 8: Hyperparameters of TCN model.

Hyperparameters	Value
Optimizer	RMSprop
Batch size	50
Epoch number	50
Loss function	Binary cross-entropy
Validation metric	Accuracy
Validation split	0.2
Deep learning framework	PyTorch 1.8.1 Gensim (WordVec) 3.7.1

Table 9: Hyperparameters of Text-CNN model.

Hyperparameters	Value
Optimizer	RMSprop
Learning rate	5e-4
Kernel size	8
Number of filters	9
Loss function	MSE
Hidden units	10
Dropout rate	0.05
Gradient clipping	1