Security-Bag: A Specification-based Intrusion Detection System Applied to Star Topology BLE Networks

Mohammad Beyrouti, Ahmed Lounis, Benjamin Lussier, Abdelmadjid Bouabdallah, Abed Ellatif Samhat

Security-Bag: A Specification-based Intrusion Detection System Applied to Star Topology BLE Networks

Mohammad Beyrouti*, Ahmed Lounis*, Benjamin Lussier*, Abdelmadjid Bouabdallah*
, Abed Ellatif Samhat†
*Université de Technologie de Compiègne, CNRS, Alliance Sorbonne Université, Heudiasyc, Compiègne, France
{mohammad.beyrouti, ahmed.lounis, benjamin.lussier, madjid.bouabdallah}@hds.utc.fr
†Faculty of Engineering-CRSI, Lebanese University
{samhat}@ul.edu.lb

***Abstract*—Bluetooth Low Energy (BLE), a protocol widely used in IoT applications, enables efficient data exchange with low-cost, resource-constrained devices. Unfortunately, BLE's numerous vulnerabilities and poor vendor patching policies, combined with the lack of seamless update mechanisms for BLE devices, expose these systems to various wireless attacks, jeopardizing the safety and security of IoT applications. This underscores the need for an Intrusion Detection System (IDS). However, IP-based IDS systems have limited BLE attack detection scope, and existing BLE-specific IDS solutions, particularly those using unreliable, Received Signal Strength Indication (RSSI)-dependent sniffers/probes and static thresholds in attack detection, suffer from limitations due to the dynamic aspects of BLE devices, such as device mobility and connection parameters update, in their designs. In this paper, we propose the *Security Bag* component: a specification-based IDS applied to star topology BLE networks that implements security rules derived from the BLE specifications. We validate this *Security Bag* by simulating spoofing attacks on a BLE sensor node using the Contiki Cooja simulator. The obtained results show that the *Security Bag* promptly detects the injected attack, opening avenues for applications on other wireless networks and deployment in real systems.**

***Index Terms*—IoT networks, Bluetooth Low Energy (BLE), Specification-based Intrusion Detection System (IDS), Security-Bag.**

I. Introduction

Internet of Things (IoT) networks connect everyday objects, from sensors to industrial machinery, enabling continuous data exchange for informed decision-making [1]. The Bluetooth Low Energy protocol (BLE) is one of the most popular wireless short range protocols for IoT devices, valued for a lightweight implementation that is invaluable for resource-constrained and mobile IoT components. A technical report by the SIG Bluetooth team reveals that 5 billion BLE devices were powered by 2023 [2].

However, the widespread use of BLE devices in sectors such as healthcare, smart grids, and smart homes has raised major security concerns. Recent discoveries of critical vulnerabilities have exposed these devices to various cyber attacks [3]. These vulnerabilities arise from both bad design and absence of appropriate security mechanisms, often due to the resource constraints in terms of energy, memory, and processing power of IoT devices [4]. The BLE specifications emphasize security mechanisms for proper encryption and authentication to prevent such vulnerabilities [5]. But, even with these security measures in place, a variety of exploitable vulnerabilities have still been identified [6]. In addition, manufacturers rarely adhere to these specifications, releasing products with proprietary BLE stacks that lack thorough security checks while prioritizing cost reduction and profit [7]. In addition, patching vulnerabilities in BLE devices, such as through firmware updates, is challenging due to resource constraints and the lack of seamless update mechanisms in BLE devices, which is exacerbated by the absence of patching policies by manufacturers. According to the IoT Security Foundation, only 10% of over 300 surveyed IoT companies have established vulnerability disclosure and patching policies [8].

This underscores the need for secondary defensive measures against threats to BLE devices. Traditional IT safeguard systems, such as firewalls or network-based IDS, are effective at analyzing IP-based network traffic but overlook internal communications of short-range wireless protocols [9]. For instance, an attacker could spoof a genuine BLE sensor to inject false readings in a hybrid network, which traditional IDS systems might consider legitimate traffic as it conforms to standard IP packet structures between the device and the rest of the network. Some work have proposed dedicated IDS for detecting BLE attacks, but these solutions often overlook crucial design and implementation considerations, leading to high false positives or negatives in attack detection. For example, they often include reliance on potentially wirelessly-vulnerable sniffers/probes components that pose challenges in key distribution and require complex bandwidth monitoring to adapt to BLE's dynamic channel hopping algorithms [10]. In addition, physical radio features often used for attack detection, such as the RSSI by these probes, can be influenced by various environmental dynamic conditions, such as noise and device mobility, and require complex design setup [11]. Detection methods that rely on BLE stack firmware modifications are also challenging, as the absence of seamless update mechanisms complicates the manual patching of proprietary BLE stacks, often requiring extensive reverse engineering.

Furthermore, the use of IDS with static timing and frequency thresholds to detect attacks are impeded by possible dynamic updates of the BLE connection parameters, such as connection interval, channel map, and latency [5]. Finally, detection strategies focused on specific attack implementations at a specific phase could be evaded by attackers, who may exploit other possible vulnerabilities or by imitating the fingerprint profile of the legitimate device [12]. Therefore, to effectively address these challenges, a carefully designed IDS attuned to the dynamics of BLE networks is essential.

In this paper, we propose *Security Bag*: a security solution inspired by the *Safety Bag* approach from the fault tolerance community [13]. It is a specification-based IDS, which we use to develop an efficient solution for the aforementioned challenges in the case of BLE star topology networks. The *Security Bag* comprises of local and global components: the local *Security Bag* component, deployed in the local BLE networks, conducts continuous diagnostic checks of BLE devices against potential attacks through either stream-based or interactive-based lightweight security rules derived by security experts from an in-depth analysis of BLE specifications. The global *Security Bag* component, located in the backend services, generates a global view of the whole network from notifications and alerts received from the local ones, informs system administrators of any security breaches and takes actions towards system recovery.

The rest of the paper is organized as follows: Section II provides an overview of the taxonomy of IDS detection methods in IoT, as well as BLE specifications, including the timing parameters for managing connection events in BLE star topology networks, and Generic Attribute Profile (GATT) services and characteristics. Section III reviews existing IDS solutions for BLE networks and identifies some of their limitations. Section IV introduces the proposed specification-based IDS security bag solution applied to BLE star topology networks. Section V details an authentication security rule to detect and tolerate spoofing attacks on BLE sensor nodes. The paper concludes with a validation of this security rule through simulations in Section VI, and then presents conclusions and future work in Section VII.

II. Background

In this section, we discuss the taxonomy of detection methods for IDS in IoT, as well as the timing and connection parameters in BLE star topology networks and the GATT layer services, based on BLE specifications.

A. Detection Strategies Classification for IDS in IoT Networks

Detection methods classification for IDS in IoT include [14], [15]:

- **Anomaly-based IDS:** This type of IDS uses analytical techniques, such as machine learning, to identify deviations from normal system behavior.
- **Signature-based IDS:** This type of IDS detects threats using a database of signatures for known malicious activities.
- **Specification-based IDS:** Also known as 'Rule-based' IDS, this type of IDS detects threats based on predefined rules designed by experts depending on the target protocol specifications.
- **Hybrid IDS:** This type of IDS can combine anomaly-based, signature-based, and specification-based methods.

B. BLE Link Layer Specifications

Figure 1 depicts a typical BLE packet exchange in a star network topology, illustrating message exchanges, connection parameters, and connection events between BLE *Slaves* and a *Master* with zero *Slave Latency* [5]. The pairing procedure is optional, based on protocol implementation.

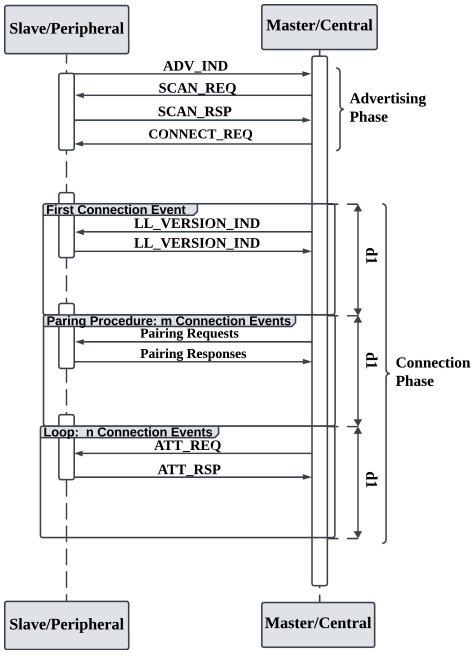

Fig. 1. Nominal Scenario Packet Exchange UML Diagram. **ADV_IND:** Advertising Packets, **SCAN_REQ/RSP:** Scan Request/Response, **CONNECT_REQ:** Connection Request, **LL_VERSION_IND:** Version Indication Packet, **ATT_REQ/RSP:** Attribute Request/Response, **d1:** Connection Interval.

Before the *Central* issues a connection request *(CONNECT_REQ)*, the *Peripherals* set no parameters. Subsequent message exchanges between the BLE *Peripherals* and the *Master* follow a timing schedule determined by the *CONNECT_REQ* Payload Data Unit (PDU) fields. The *CONNECT_REQ* PDU includes a *Header* and a *Payload*, with the latter containing fields such as the Initiator's Address (InitA), Advertiser's Address (AdvA), and the Link Layer Data field (LLData). The LLData consists of ten fields, three of which are crucial in designing the proposed security rule in this paper.

- *Interval field:* Indicates the *Connection Interval (**d1**)*, the time between two successive connection events ($7.5\,\text{ms} \leq d1 \leq 4\,\text{s}$).
- *Timeout field:* Specifies the *Connection Supervision Timeout (**d2**)*, the maximum time allowed between receiving two *Data Packet PDU* before considering the connection lost ($100\,\text{ms} \leq d2 \leq 32\,\text{s}$).

- *Latency field:* Defines the *Connection Slave Latency (**d3**)*, the number of connection events a peripheral device can skip ($0 \leq d3 < 500$).

C. BLE Generic Attribute Profile (GATT) Layer Services and Characteristics

In the BLE framework, a *Service* within the GATT server (peripheral) manages application data transfer with the GATT client (central) using the Attribute (ATT) protocol, and is categorized into standard and custom types. Standard services, predefined by the SIG Bluetooth team, support requests for generic data from sensors or commands to actuators in some common applications. Custom services can be developed by anyone designing a BLE application. Each service has a Universally Unique Identifier (UUID), with standard services using a 16-bit UUID and custom services a 128-bit UUID. Services consist of *Characteristics*, each one hold a UUID, a *Data Value*, a *Property*, *Permissions*, and possibly *Descriptors*. The *Data Value* specifies the type and length of data sent in the ATT responses, with types ranging from unsigned bytes to strings and arrays. The *Property* defines access methods—such as read, write, and notify—and *Permissions* outline access requirements, like encryption. *Descriptors* may also be included to provide additional details, helping GATT clients manage notifications and settings. The detailed configuration setup of custom services and characteristics in BLE is outlined in [16].

III. Related works

This section provides an overview of state-of-the-art IDS approaches proposed for star topology BLE networks, along with their limitations.

A. IDS for BLE IoT Networks

Salem et al. [17] proposed a specification-based detection mechanism for identifying spoofing attacks in IoT healthcare, using Locality-Sensitive Hashing (LSH) to generate energy-efficient data signatures. Roux et al. [18] proposed RadIoT, an anomaly-based machine learning IDS that detects intrusions in short-range wireless protocols, including BLE, by analyzing physical radio communication properties such as RSSI, frequencies, and reception time. Yurdagul et al. [19] proposed a specification-based IDS to detect spoofing attacks on BLE devices by monitoring the regularity of IoT devices' response time behaviors during read and write operations. Wu et al. [20] proposed BlueShield, a specification-based IDS designed to guard against spoofing attacks by monitoring cyber-physical features in BLE advertising packets to prevent attackers from mimicking legitimate devices. Gu et al. [21] proposed BF-IoT, an anomaly-based IDS framework using machine learning to detect BLE spoofing attacks through a two-phase authentication process. This process fingerprints IoT devices using unique feature sets extracted from advertising packet intervals at the link layer and response times at the ATT/GATT packet layers in the BLE stack. Yaseen et al. [22] proposed a signature-based IDS framework for BLE devices lacking Input/Output capabilities (NiNo devices), targeting spoofing attacks. This framework uses the RSSI level, advertisement packet time intervals, malicious device addresses, and malicious scan requests as metrics for detection. Cayre et al. [23] proposed OASIS, a hybrid (signature and specification-based) IDS framework for BLE, designed to generate and integrate detection software with BLE controllers to enhance protocol attack detection. This involves analyzing BLE stack firmware with a firmware analyzer and a build system to create and update detection software, providing a user-friendly tool for generating detection modules.

B. Limitations

The reviewed literature propose several interesting approaches for BLE attack detection, but suffer from some limitations. First, some IDS use probes and sniffers to collect packets, which could themselves be vulnerable to attacks. They also face challenges in key distribution during BLE pairing and in monitoring large networks to synchronize with channel hopping algorithms, especially when the network devices have small connection intervals ($L_{sniffers}$). Second, some IDS systems use RSSI for attack detection, but RSSI can be affected by environmental factors such as noise, interference with protocols in the same frequency bands, device mobility, battery level, and physical barriers, requiring complex device fingerprinting. These factors often result in numerous false positives, and particularly false negatives, by imitating the RSSI profile of the device (L_{RSSI}). Third, some proposed mechanisms require complex modifications to the BLE protocol stack, which are often hard to implement due to BLE devices' limited resources for flexible updates. Moreover, proprietary BLE stack firmware images often suffer from poor documentation, restricting patching tools application mainly to open-source BLE stacks ($L_{firmware\_modification}$). Fourth, some solutions, particularly those using device signatures or analyzing response times and frequency patterns based on static thresholds, fail to consider BLE dynamic connection parameter updates scenarios and extended supervision timeouts that permit peripherals to skip responses without disconnecting. For example, updates are initiated by the central BLE device via a *CONNECT_UPDATE_PDU* request to adjust parameters such as connection interval, channel map of the channel hopping algorithms, slave latency, and supervision timeout. Such oversights could result in high false positives or negatives ($L_{connection\_updates}$). Fifth, as new BLE vulnerabilities emerge, attackers can exploit those vulnerabilities to launch different variants of attacks, potentially evading the IDS, all while maintaining correct timing parameters [12]. This consideration, could lead to high false negatives ($L_{bypassing\_detection}$). Table I presents the limitations faced by the different approaches presented in subsection III-A.

IV. Security Bag: A Novel Specification-based IDS Applied to BLE Networks

In this section, we present our proposed specification-based IDS for BLE star topology, designed to detect active attacks using security rules designed by experts based on

TABLE I
COMPARISON OF THE DISCUSSED EXISTING IDS IN ADDRESSING THE AFOREMENTIONED LIMITATIONS WITH THE PROPOSED WORK. ✗: NOT ADDRESSED, ✓: ADDRESSED

Proposed Approach	**Against Attacks**	**Limitation in Design and Implementation**				
		$L_{sniffers}$	L_{RSSI}	$L_{firmware\_modification}$	$L_{connection\_updates}$	$L_{bypassing\_detection}$
Salem et al. [17]	Spoofing, DoS	✓	✗	✗	✓	✗
Roux et al. [18]	Spoofing, DoS	✗	✗	✓	✗	✗
Yurdagul et al. [19]	Spoofing	✗	✓	✓	✗	✗
Wu et al. [20]	Spoofing	✗	✓	✓	✗	✗
Gu et al. [21]	Spoofing	✗	✓	✓	✗	✗
Yaseen et al. [22]	Spoofing, DoS	✗	✗	✓	✗	✗
Cayre et al. [23]	Spoofing, DoS, Other Attacks	✓	✓	✗	✓	✓
Proposed Approach	Spoofing, DoS, Other Attacks	✓	✓	✓	✓	✓

BLE specifications. This IDS can also take recovery actions to ensure that the system remains safe and secure, either by interacting directly with the system or by sending notification alerts to the system's administrators. The IDS strategy employs centralized, independent components called *Security Bag*, which are divided into local and global categories and are inspired by the *Safety Bags* [13]. The local *Security Bag* is a trusted entity housed within a secure lockable enclosure to avoid physical tampering from an attacker. It establishes a secure wired connection with the local BLE gateway to detect threats that could emerge from wireless connections and physical tampering. Meanwhile, the global *Security Bag*, located within backend security services, receives notifications from the local *Security Bags* through a secure connection to alert users about attacks or to identify large-scale threats that could compromise business assets. Figure 2 depicts the high-level architecture integrating both local and global *Security Bags* based on the ISO IoT architecture [24].

The specification-based security rules that can be implemented by the local *Security Bag* are categorized into two types:

1) Stream-based Diagnostic Rules

These rules are implemented by the *Security Bag* to passively monitor network stream and detect deviations from the normal specification patterns of packet exchange without direct interaction with the system.

2) Interactive-based Diagnostic Rules

These rules are implemented by the *Security Bag* to detect attacks through direct interaction with the IoT system.

A. Security Rules Design

The design approach for security rule type requires a comprehensive investigation into the BLE specifications such as the one discussed in section II and the targeted attacks detection requirements by security experts.

1) Stream-based Diagnostic Security Rules: Design Requirements

Developing stream-based diagnostic security rules in our proposed IDS requires a detailed analysis of the targeted IoT protocol's timing and connection parameters, such as

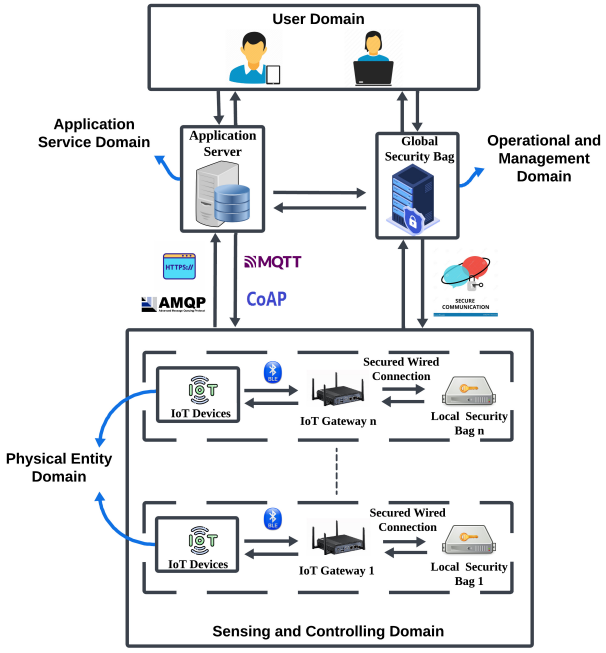

Fig. 2. Security Bag Integration with BLE Network Based on ISO IoT Architecture

the connection interval, supervision timeout, and latency, as explained in subsection II-B. These parameters determine the communication behavior between the IoT device and the gateway: under normal conditions, protocol packets are transmitted at consistent intervals depending on these parameters. Stream-based rules are designed to detect deviations from the expected timing patterns in packet exchanges, flagging any unexpected flow patterns as potential security threats based on the target attack detection requirements.

2) Interactive-based Diagnostic Security Rules: Design Requirements

Developing interactive diagnostic rules in our IDS involves designing custom service feature profiles at the GATT layer of the BLE protocol or potentially incorporating other built-in services. This method leverages the capability to configure

and add new custom service profiles and characteristics, as discussed in subsection II-C, without necessitating firmware stack modifications. This way, it allows us to directly incorporate security mechanisms into the custom-designed GATT specifications at both the IoT gateway and IoT device levels, based on target attack detection requirements.

Such integration enables the definition of security rules at the *Security Bag* level, offering several advantages. Firstly, the *Security Bag's* reliance on specification-based IDS can obtain a low false positive rate through a thorough study and observance of the BLE specifications. Secondly, the *Security Bag* rules can use a lightweight implementation of security mechanisms (like encryption or timed acknowledgments), suitable for resource-constrained IoT devices, and can be carried out by custom services at the GATT layer with no direct modifications of the BLE stack (addressing $L_{firmware\_modifications}$). Thirdly, our IDS operates on a secure wired connection with the gateway, thus not being vulnerable to wireless attacks (addressing $L_{sniffers}$) and using direct reports from the gateway without needing to listen to the radio traffic for anomaly detection (addressing L_{RSSI}). It can also accommodate dynamic thresholds update to connection parameters through security rule design (addressing $L_{connection\_updates}$). Finally, its rules can be generic to apply to known attacks and possible new variants (addressing $L_{detection\_bypass}$), as well as specific to target a particular attack.

V. Interactive-based Diagnostic Security Rule Design for BLE Sensor Nodes Spoofing

In this section, we propose an authentication diagnostic mechanism for our *Security Bag* using a custom GATT service (*AuthenticationDiagnostic*). This service includes two custom-designed characteristics, *EncryptedNonce (C1)* and *EncryptedHashNonce (C2)*, embedded with security features in sensors nodes (peripheral devices). These characteristics enable the local *Security Bag* to perform periodic authentication checks with these nodes, defending against different variants of spoofing attacks and potentially other types of attacks using the ATT protocol.

A. Authentication Diagnostic Check Custom Service Design

Figure 3 shows the configured custom service and its specific characteristics at each sensor node, along with the standard service the node offers. Characteristic *C1* has a "Write" property for handling encrypted nonces sent by the *Security Bag* via the gateway (central), while *C2* with a "Read" property, allows the sensor node to send back the encrypted hash of the nonce as requested by the gateway to the *Security Bag*. Each characteristic uses a byte string data type, with lengths determined by the encryption and hashing algorithms used. We modified the gateway's application logic to send the necessary ATT requests and handle the ATT responses as directed by the *Security Bag*, ensuring accurate implementation of authentication diagnostic rules. Characteristic permissions depend on the BLE stack's support for link layer encryption, encrypting all ATT packets and applying double encryption to nonces within custom services. If encryption is disabled (possibly due to resource constraints), permissions allow unencrypted access, but authentication checks remain effective through a resilient, lightweight encryption and hashing algorithm deployed at the application layer, securing the authentication payload independently of the stack's encryption capabilities while keeping the required resources low.

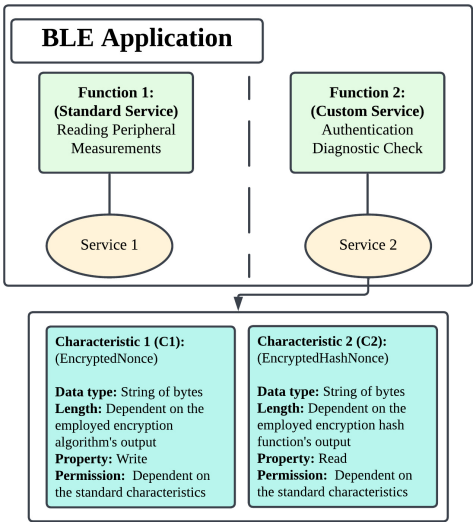

Fig. 3. Custom-Designed Authentication Diagnostic Service and its Characteristics on Each BLE Sensor Node.

B. Periodic Nonce-Based Authentication Mechanism by Security Bag

Algorithm 1 details a singular implementation of the authentication diagnostic security rule by the *Security Bag* that consists of two main phases executed successively and a third independent phase highlighted in the algorithm, aimed at verifying the authentication of an individual IoT sensor against potential spoofing attempts. This instance can be replicated to authenticate multiple nodes within the same BLE network.

1) Phase 1: Authentication Check Activation

In the first phase, the security bag activates the authentication mechanism near a desired frequency of $freq_{auth}$, triggered every $\frac{1}{k_i}$ standard service connection events, k_i being a value between 0 and 1 calculated from the connection parameters as explained in phase 3. The *Security Bag* generates an encrypted nonce using a pre-deployed resilient Lightweight Cryptographic Algorithm (LCA_i) and a pre-shared key K_i (deployed manually or possibly via a secure, custom-designed key exchange service) and sends it to the gateway. This nonce is then hashed using a Lightweight Cryptographic Hashing algorithm ($LCHA_i$), storing it to verify sensor authenticity and maintain data integrity, considering the gateway as untrusted entity for nonce generation and encryption.

2) Phase 2: Response Handling

In the second phase, the gateway sends the encrypted nonce to the BLE node via an *ATT Write Request* targeting the *C1* characteristic of the custom GATT service at the next *Connection Event n*. The BLE node then retrieves the nonce from

Algorithm 1 Periodic Authentication Diagnostic Security Rule for IoT Sensor Node$_i$ Implemented by the Local Security Bag

Input: $d1_i$, $d2_i$, $d3_i$, network parameters update
Output: Alerts/Notifications, Abort connection command
Initialize: $k_i = f(d1_i, d3_i), 0 < k_i < 1$
Initialize: Counter $i = 0$
loop
  if a message is received **then**
    Counter $i \leftarrow i + 1$
  end if
  if Counter $i \geq \frac{1}{k_i}$ {Corresponding to $freq_m$ duration} **then**
    {***Phase 1: Authentication Check Activation***}
    Activate Authentication Check Mechanism
    Generate a random nonce N
    SecurityBagHash $\leftarrow$ Hash nonce N using $LCHA_i$
    Store $SecurityBagHash$ {for later}
    Encrypt the nonce with pre-shared key K_i using LCA_i
    send *EncryptedNonce* to gateway (Master)
    {***Phase 2: Response Handling***}
    wait for the encrypted *SensorHash* from $node_i$
    Decrypt the message using LCA_i
    Compare the *SecurityBagHash* with the received *SensorHash*
    if they do not match **then**
      Trigger an alert for $node_i$; Spoofing
      Inform the gateway to terminate the connection of $node_i$
    else
      $node_i$ is authenticated
    end if
    Counter $i = 0$
  end if
  {***Phase 3:*** $freq_{auth}$ ***Optimization***}
  if connection parameters update received **then**
    Adjust $k_i = f(d1_{i,new}, d3_{i,new})$
  end if
end loop

C1, decrypts it using the pre-shared key as per LCA_i, hashes it using $LCHA_i$, re-encrypts it with LCA_i, and stores it in the *C2* characteristic. This process safeguards the data in the node's custom services against unauthorized data extraction. At the following *Connection Event n+1*, the gateway issues an *ATT Read Request* to retrieve the encrypted hash from *C2*, which is then relayed to the *Security Bag* for decryption and hash verification. A mismatch between this hash and the *Security Bag's* calculated hash indicates a potential spoofing attack, prompting the gateway to abort the connection and sending alerts to the global *Security Bag*. If the hashes match, the node is authenticated, allowing the *Security bag* to continue receiving responses at subsequent *Connection Events* and to perform periodic rule-based authentication checks.

3) Phase 3: $freq_{auth}$ Optimization

The value of $freq_{auth}$, determining the activation frequency of the authentication check after a certain number of connection events, is dependent on the value k_i, which is adjusted every time that the connection parameters change. The value k_i needs to be determined by security experts based on the application's security requirements and the node's standard service connection events frequency ($freq_m$). Two BLE parameters, the connection interval (*d1*) and slave latency (*d3*), influence $freq_m$, given by $freq_m = \frac{1}{d1 \times (1+d3)}$ Hz, where the sensor node is expected to respond with one ATT response every $d1 + d1 \times d3$. $freq_{auth}$ is established as a proportional factor less than $freq_m$, inversely related to $d1$ and $d3$: $freq_{auth} = k_i \times freq_m = \frac{k_i}{d1 \times (1+d3)}$ Hz $\quad, 0 < k_i < 1$.

Security experts must thus develop a mathematical model, either linear or non-linear, to dynamically adjust the scaling factor k_i based on $d1$, and $d3$ connection parameters, expressed as $k_i = f(d1, d3)$. This could be defined by the formula: $k_i = \frac{\alpha}{d1^{\gamma} \cdot (1+d3)^{\beta}} + \delta$, where, α, β, γ, and δ are constants determined from empirical data within BLE's specified $d1$ and $d3$ ranges (subsection II-B). This configuration ensures that increases in $freq_m$ result correspond to decreases in $freq_{auth}$, and vice versa, to meet security requirements. Hence, the fraction $(\frac{1}{k_i})$ represents the frequency of authentication checks in terms of the number of standard GATT service connection events. This approach enables the security bag to adjust thresholds in real-time, particularly when the *Master* requests an update to connection parameters via *CONNECT_UPDATE_PDU*, thereby aligning $freq_m$ and $freq_{auth}$ to optimize energy consumption.

VI. Validation Through Simulations

We validated the authentication diagnostic security rule presented in section V using the Contiki Cooja network simulator [25] on a IoT healthcare management system. This system includes wireless health indicator sensors (temperature, heart rate and blood pressure) and actuators (insulin pumps). In the simulation, the key components, including a temperature sensor (peripheral), the IoT gateway (master), the local *Security Bag*, and the attacker devices, were modeled using Cooja motes. We emulated an abstract BLE protocol between the sensor node and the gateway as depicted in Figure 1, without pairing. In addition, we integrated the custom GATT services and characteristics as presented in section V. We set up a TCP/IP-based client-server architecture between the gateway and the local *Security Bag* for secure communication. We then implemented algorithm 1 in the *Security Bag*, integrated with simple designed lightweight encryption and hashing algorithms for continuous sensor node diagnosis. For the attack simulations, we used two motes: one to sniff and inject malicious data into the temperature sensor node, and another for the IoT gateway. We assume that the local *Security Bag* is connected to a global *Security Bag* to send the necessary notifications upon attack detection, which is excluded here due to its minor role in the security rule's validation. Figure 4 depicts the network configurations including the *Security Bag* under both normal and attack conditions.

During both nominal and attack scenarios, the mote output window of the simulator displays acknowledgements for sending, forwarding, receiving, or sniffing messages by each node. Each packet in this window is identified by a PDU type and payload, which detail the packet's name and content. Simulation parameters, including those for managing connection events discussed in subsection II-B, are selected: $d1 = 250$ ms, $d2 = 600$ ms, and $d3 = 0$. To simplify the simulations for demonstration purposes, we do not update BLE connection parameters during a scenario; thus, $freq_{auth}$ is set to 0.26 Hz and $\frac{1}{k} = 15$. Therefore, the authentication mechanism will activate every 15 connection events.

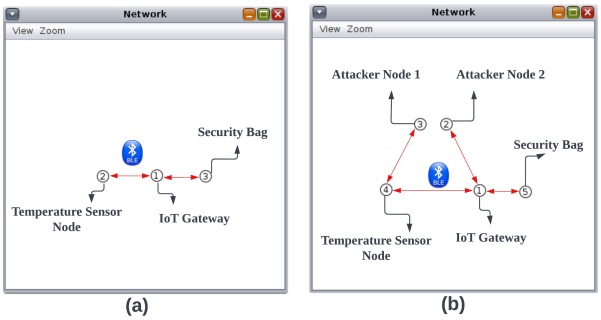

Fig. 4. Network Configurations with Security Bag in Nominal and Attack Scenarios

A. Nominal Scenario Simulation With Security Bag

Figure 5.(a)-(b) depicts the simulation of the nominal scenario packet exchange between the sensor node, the gateway, and the security bag, corresponding to the network configuration (a) in Figure 4. We noticed that the security bag performs the diagnostic check of sensor's authenticity periodically using the proposed algorithm in section V.

B. Attack Injection Simulation With Security Bag

We simulated a spoofing attack on the sensor node during a patient emergency (average temperature 39°C), based on exploiting CVE-2021-31615 and CVE-2020-15486 vulnerabilities from the NVD[1], using emulated attack nodes (nodes 2 and 3) to hijack the connection and inject false temperature readings. The successful attack, shown in Figure 5.(c), aligns with the network configuration in Figure 4.(b). During the connection phase, the attacker sniffed communication between the sensor node and the gateway. At connection event #1000, node 3 injected an *LL_TERMINATE_IND* packet at t=471.184 s (07:51.184) into the sensor, preempting the gateway's *ATT_REQ* and causing the sensor to exit the communication without sending a temperature response. With a *Supervision Timeout (d4)* interval of 600 ms, the IoT gateway initiated another request at connection event #1001. Here, attacker node 2 sniffed the *ATT_REQ*, spoofed its MAC address to that of the genuine temperature sensor, and injected a *Malicious ATT_RSP* with a falsified normal 37°C temperature reading, synchronizing with the gateway for subsequent events. In Figure 5.(d), the *Security Bag* detected the spoofing attempt at t=472.742 s (07:52.742), about 1.5 seconds post-injection, prompting the gateway to abort the connection and alert medical staff through the global *Security Bag*. In a worst-case scenario test, the attack was injected immediately after the authentication check. The *Security Bag's* diagnostics activates every 15 measurement events (roughly every 4.234 s) with an additional 500 ms execution time, and thus detected the attack in about $\approx$ 4.734 s.

VII. Conclusion and Future Work

In this paper, we proposed the *Security Bag*, an independent specification-based IDS for detecting and tolerating attacks in BLE star topology IoT networks. The *Security Bag* consists of both local and global components. The local *Security Bag*, implemented on local BLE networks, detects attacks by monitoring BLE stream or interacting directly with the system using diagnostic security rules derived from BLE protocol specification. The global *Security Bag*, implemented at the backend services, receives notifications from the local ones to generate actions on the whole network and alert the system's administrators of any security breaches in the BLE networks. As a practical example, we designed an interactive-based security rule to detect spoofing attacks on BLE sensor nodes. This is achieved through a periodic diagnostic authentication mechanism by a local *Security Bag*, which checks sensor nodes authenticity by interacting with them using a GATT custom-designed service. It should be noted that the design of security rules greatly depends on the experts' skills, requiring in-depth analysis of BLE specifications—a limitation of specification-based IDS. Nonetheless, initial tests conducted using Cooja simulations show promising results for future deployment of the *Security Bag* in BLE networks. For future perspectives, there is interest in comparing the *Security Bag* with other approaches presented in related works through experiments. Additionally, there is an intent to develop more security rules to detect various BLE attacks. Moreover, plans are underway to adapt the *Security Bag* to other short-range wireless IoT protocols. Finally, more simulations and experiments are planned to further validate the efficacy and trustworthiness of the *Security Bag* in real-world applications.

[1]https://nvd.nist.gov/

Acknowledgment

This work was funded by the International Research Project Adonis and the French region Hauts-de-France.

References

[1] A. Sadeghi-Niaraki, "Internet of thing (iot) review of review: Bibliometric overview since its foundation," *Future Generation Computer Systems*, 2023.

[2] B. SIG, "Bluetooth market update," Bluetooth Special Interest Group (SIG), Tech. Rep., 2019, https://www.bluetooth.com/bluetooth-resources/2019-bluetooth-market-update/.

[3] M. E. Garbelini, C. Wang, S. Chattopadhyay, S. Sumei, and E. Kurniawan, "{SweynTooth}: Unleashing mayhem over bluetooth low energy," in *2020 USENIX Annual Technical Conference (USENIX ATC 20)*, 2020, pp. 911–925.

[4] M. Beyrouti, A. Lounis, B. Lussier, A. Bouadallah, and A. E. Samhat, "Vulnerability and threat assessment framework for internet of things systems," in *2023 6th Conference on Cloud and Internet of Things (CIoT)*. IEEE, 2023, pp. 62–69.

[5] B. SIG, "Bluetooth core specification," Bluetooth Special Interest Group (SIG), Tech. Rep., 2023, version 5.4.

[6] P. Sivakumaran and J. 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 *28th USENIX Security Symposium (USENIX Security 19)*, 2019, pp. 1–18.

[7] J. Li, J. Jin, L. Lyu, D. Yuan, Y. Yang, L. Gao, and C. Shen, "A fast and scalable authentication scheme in iot for smart living," *Future Generation Computer Systems*, vol. 117, pp. 125–137, 2021.

[8] I. S. Foundation, "Understanding the contemporary use of vulnerability disclosure in consumer internet of things product companies," IoT Security Foundation, Tech. Rep., 2018.

[9] C. K. Nkuba, S. Woo, H. Lee, and S. Dietrich, "Zmad: Lightweight model-based anomaly detection for the structured z-wave protocol," *IEEE Access*, 2023.

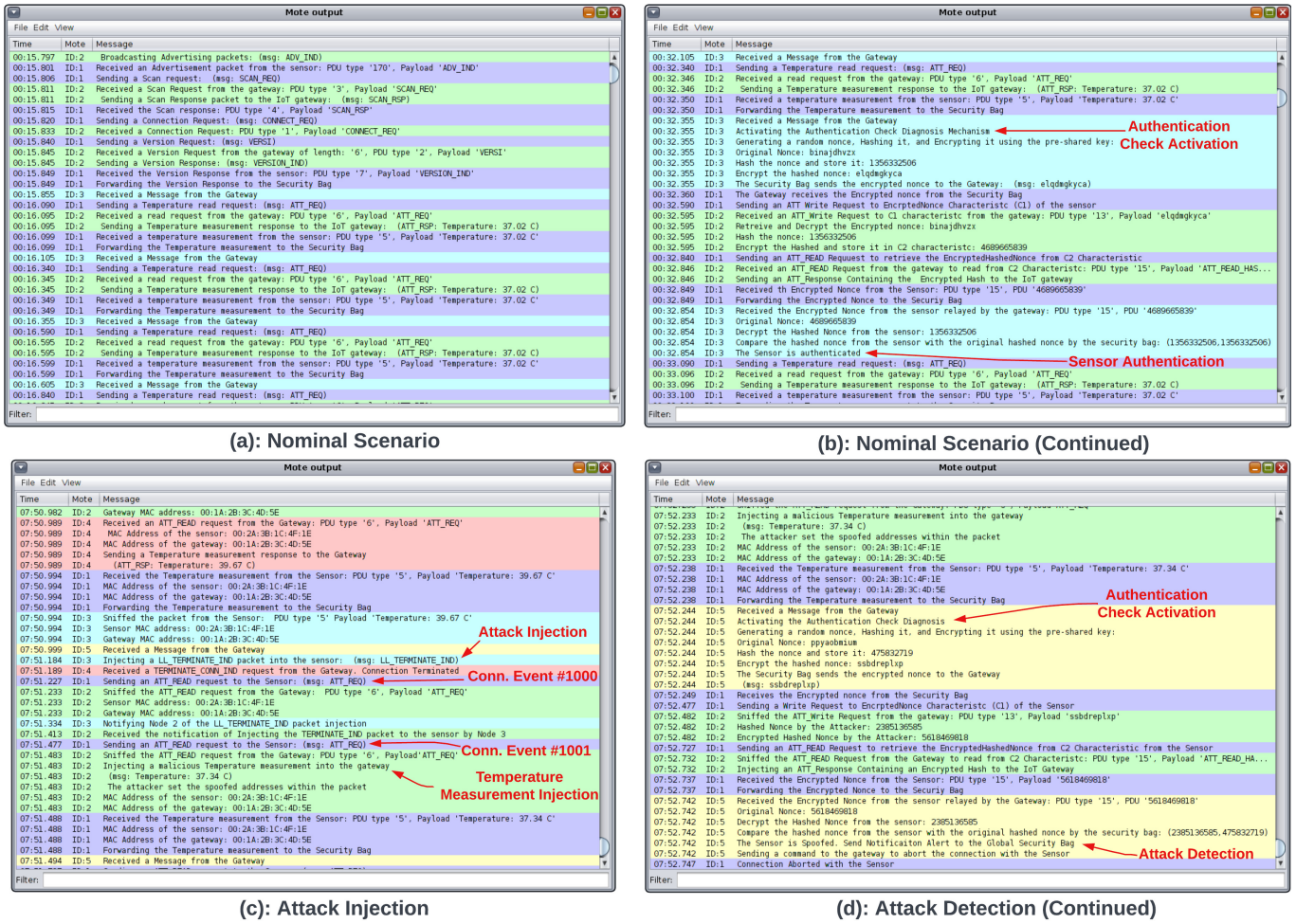

Fig. 5. Validation Results for Nominal and Attack Detection Scenarios with Security Bag Deployment

[10] S. Sarkar, J. Liu, and E. Jovanov, "A robust algorithm for sniffing ble long-lived connections in real-time," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[11] A. A. Ghany, B. Uguen, and D. Lemur, "A robustness comparison of measured narrowband csi vs rssi for iot localization," in *2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*. IEEE, 2020, pp. 1–5.

[12] R. Cayre, F. Galtier, G. Auriol, V. Nicomette, M. Kaâniche, and G. Marconato, "Injectable: Injecting malicious traffic into established bluetooth low energy connections," in *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2021, pp. 388–399.

[13] C. Paul, L. Benjamin, S. Walter, and M. Brini, "Validation of safety necessities for a safety-bag component in experimental autonomous vehicles," in *2018 14th European Dependable Computing Conference (EDCC)*. IEEE, 2018, pp. 33–40.

[14] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE communications surveys & tutorials*, vol. 16, no. 1, pp. 266–282, 2013.

[15] A. A. Anitha and L. Arockiam, "A review on intrusion detection systems to secure iot networks," *International Journal of Computer Networks and Applications*, vol. 9, no. 1, pp. 38–50, 2022.

[16] R. Kumar, A. P. Family, and C. CY8C4XXX-BL, "Creating a ble custom profile," *Cypress Semiconductors. Dosegljivo: http://www. cypress. com/file/140826/download.[Dostopano: februar 2016]*.

[17] O. Salem, K. Alsubhi, A. Shaafi, M. Gheryani, A. Mehaoua, and R. Boutaba, "Man-in-the-middle attack mitigation in internet of medical things," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 2053–2062, 2021.

[18] J. Roux, E. Alata, G. Auriol, M. Kaâniche, V. Nicomette, and R. Cayre, "Radiot: Radio communications intrusion detection for iot-a protocol independent approach," in *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*. IEEE, 2018, pp. 1–8.

[19] M. A. Yurdagul and H. T. Sencar, "Blekeeper: Response time behavior based man-in-the-middle attack detection," in *2021 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2021, pp. 214–220.

[20] J. Wu, Y. Nan, V. Kumar, M. Payer, and D. Xu, "{BlueShield}: Detecting spoofing attacks in bluetooth low energy networks," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020, pp. 397–411.

[21] T. Gu and P. Mohapatra, "Bf-iot: Securing the iot networks via fingerprinting-based device authentication," in *2018 IEEE 15Th international conference on mobile ad hoc and sensor systems (MASS)*. IEEE, 2018, pp. 254–262.

[22] M. Yaseen, W. Iqbal, I. Rashid, H. Abbas, M. Mohsin, K. Saleem, and Y. A. Bangash, "Marc: A novel framework for detecting mitm attacks in ehealthcare ble systems," *Journal of medical systems*, vol. 43, pp. 1–18, 2019.

[23] R. Cayre, V. Nicomette, G. Auriol, M. Kaâniche, and A. Francillon, "Oasis: An intrusion detection system embedded in bluetooth low energy controllers," 2024.

[24] "ISO/IEC 30141:2018, Internet of Things (IoT) — Reference architecture," International Organization for Standardization and International Electrotechnical Commission, Tech. Rep. ISO/IEC 30141:2018, 2018.

[25] G. Oikonomou, S. Duquennoy, A. Elsts, J. Eriksson, Y. Tanaka, and N. Tsiftes, "The Contiki-NG open source operating system for next generation IoT devices," *SoftwareX*, vol. 18, p. 101089, 2022.