Computer Networks 211 (2022) 108953

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Survey paper

Securing Bluetooth Low Energy networking: An overview of security procedures and threats

Andrea Lacava [a,b,*], Valerio Zottola [a], Alessio Bonaldo [a], Francesca Cuomo [a], Stefano Basagni [b]

[a] "Sapienza" University of Rome, 00184 Rome, Italy
[b] Institute for the Wireless Internet of Things, Northeastern University, Boston, MA, USA

ARTICLE INFO

Keywords:
Bluetooth Low Energy
Bluetooth mesh networking
Data and security
Intrusion Detection Systems
Attack prevention

ABSTRACT

Bluetooth Low Energy (BLE) is rapidly becoming the de-facto standard for short-range wireless communications among resource-constrained wireless devices. Securing this technology and its networking capabilities is paramount, as their widespread use by Internet of Things (IoT) applications demands protection from malicious users. While its security features have remarkably improved over the years, the BLE technology is still prone to severe threats, creating a gap between the standard theoretical design and its implementation. Particularly, the BLE Mesh Profile (*Bluetooth Mesh*), which enables many-to-many communication, prompts an overall analysis of its security, to ensure that its use preserves the integrity and privacy of end users. This work surveys the state-of-the-art of the security of BLE with an emphasis on Bluetooth Mesh, highlighting the threats that can still hinder their usage. We review the latest specifications in terms of link set up and authentication and describe attacks to both point-to-point and multicast networking. Our work also discusses solutions to mitigate and prevent attacks to the current standard, such as Intrusion Detection Systems, thus improving the general level of security of BLE systems.

1. Introduction

Since its creation in the late 1990s the Bluetooth technology has become the *de facto* standard for simple and secure short range wireless connectivity. Initially conceived for cable replacement, Bluetooth has mostly been used for point-to-point wireless communication in the 2.4 GHz ISM band, sharing the spectrum with other technologies for local area networking under the IEEE 802.x umbrella, including WiFi (802.11xx). The standard is publicly available from the Bluetooth Special Interest Group (aka, the SIG Alliance) [1]. Over the years three main distinct Bluetooth physical layers have emerged: Basic Rate and Enhanced Data Rate (BR/EDR), High Speed (HS) and Low Energy (BLE). Starting with the fourth version of Bluetooth, the first two physical layers go by the name of *Bluetooth Classic* (BT), while the Low Energy variant is commonly referred to as BLE. Even though all three physical layers are currently used in the mobile market, BLE appears to be the most widely used version of Bluetooth in smartphones and IoT devices as it satisfies their energy saving demands and supports multiple types of communication.

Fig. 1, inspired by the Bluetooth Technology Overview [1], depicts the differences between the communication modes available in BT and BLE. While the earlier Bluetooth Classic offers point-to-point communication in support of audio streaming and data transfer, BLE adds broadcast communications (one-to-all) that are particularly suited to serve indoor localization and contact tracing applications. BLE also provides functionalities to configure mesh networks (many-to-many communications) to realize complex solutions for IoT. For all communication modes, BLE introduce new jargon. For instance, in the realm of point-to-point communication, a device can be a *Central* device, actively scanning for other BLE devices to connect to, or a *Peripheral* device, which establishes a connection with a central device. In the broadcast communication mode, devices can be either *Broadcasters*, sending connection-less advertisements, or *Observers*, which scan the advertisements sent by the Broadcaster. The mesh communication mode, which enables many-to-many communications, is composed of BLE *broadcasters* and *observers* working together to create a flooding-based network that interconnects all devices. New names for devices in mesh mode include *provisioner* and *unprovisioned device*, *node* and *relay node*, and others names (listed in details in the following sections).

BLE is designed with security in mind, building on the primitives of BT and expanding them to address the security issues of the new communication modes [2]. In this work we focus on the security features of BLE and in particular on those presented in the 5.1 Core Specification [3]. We place particular emphasis on the security aspects

* Corresponding author at: "Sapienza" University of Rome, 00184 Rome, Italy.
E-mail addresses: lacava.a@northeastern.edu (A. Lacava), zottola.1701177@studenti.uniroma1.it (V. Zottola), bonaldo.1645694@studenti.uniroma1.it (A. Bonaldo), francesca.cuomo@uniroma1.it (F. Cuomo), s.basagni@northeastern.edu (S. Basagni).

https://doi.org/10.1016/j.comnet.2022.108953
Received 4 September 2021; Received in revised form 13 March 2022; Accepted 6 April 2022
Available online 26 April 2022

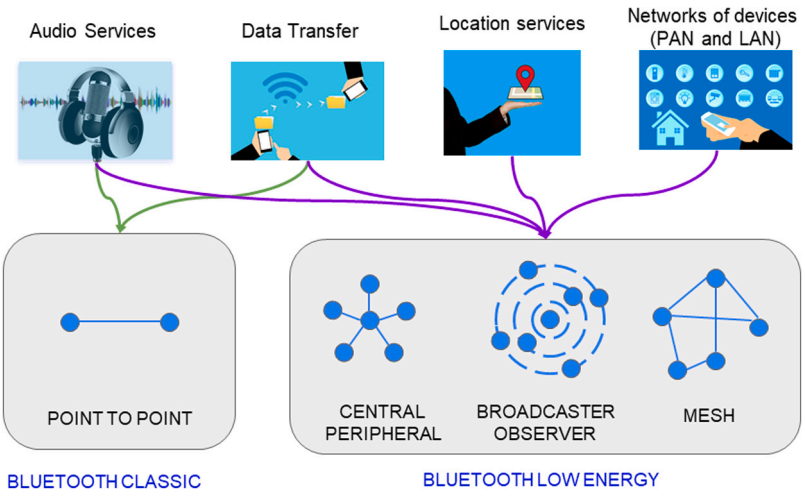

Fig. 1. Bluetooth Classic (BT) vs. Bluetooth Low Energy (BLE): Communication modes [1].

of the BLE Mesh Profile (BLE Mesh or Bluetooth Mesh in the following), which is relevant to the implementation of a vast range of IoT applications. We review the current status of BLE security and of BLE Mesh in terms of available features and threats, indicating current and desirable defense mechanisms. We consider Bluetooth version 5.1 as it pays particular attention to relevant security features and because its distribution on the market is rapidly growing. We will also present some of the new security features introduced in the 5.2 Core Specification [4], describing where in the standard they have been introduced and for what purpose, thus providing a more complete picture of the security of the current standard.

A handful of previous works have described the vulnerabilities that compromise the general security of the BLE technology. Many of these vulnerabilities have been addressed by newer versions of the standard, in that the attacks that exposed them were mitigated or obviated by new security features. For instance, Pammi provides a comprehensive description of BLE and of the type of attacks that could occur in a BLE-based IoT setting [5]. However, with the newer versions of the core specifications these vulnerabilities have been patched and the attacks are not reproducible anymore. Other works analyze attacks that are specific to given architectures or communication modes, and do not provide a general review of BLE security. Examples include DoS attacks against Bluetooth devices that cause battery exhaustion [6], and various security threats to Bluetooth communications in e-Health systems [7]. Tay et al. provide no general description of BLE security, focusing mainly on the IBeacon protocol and on attacks to devices using only the broadcast communication mode [8]. Adomnicai et al. consider the vulnerabilities of the physical layer for BLE Mesh networks [9]. Works that, like ours, overview security aspects of BLE fall short of the generality we aim at achieving. For instance, Zhang et al. provide a solid introduction to the BLE standard that however focuses primarily on the *Secure Connection method* (Section 3). In addition, they perform an analysis of security design flaws mainly on android devices where the Secure Connection method is not enforced during pairing. Ghori et al. present a general overview of BLE Mesh security [10]. However, their review does not consider newer specific threats affecting the latest versions of the protocol.

The aim of our work is to provide an up-to-date survey of the security features of the current BLE standard, and a comprehensive account of the latest known attacks to devices implementing both BLE and BLE Mesh. An additional contribution of our survey is a new taxonomy of known attacks that include effective mitigation remedies for each attack. To the best of our knowledge, no comprehensive review of BLE and BLE Mesh security is available, which makes our contribution the first to highlight how secure is BLE networking.[1]

Our review is organized as follows. Section 2 provides an introduction to BLE, highlighting the layers of the BLE stack and the characteristics of its Mesh Profile. In Section 3 we overview the security features of the Bluetooth Core Specification 5.1 for BLE [3] and of version 1.0.1 of the Mesh Profile [11]. Current known attacks and their proposed countermeasures are described in Section 4. Here we propose a categorization of known attacks based on the BLE version, the attacked layer of the BLE stack, and the compromised cybersecurity metric (e.g., confidentiality, integrity or availability). In Section 5 we describe attack prevention methodologies to counter the attacks described in Section 4. Conclusions are drawn in Section 6.

2. BLE and the Mesh Profile

BLE devices have three different modes for data exchange: *1. central-peripheral* communication, where *peripheral devices* are connected to one coordinator called the *central unit*; *2. broadcaster-observer* communication, in which the exchange happens in a connection-less mode and it is led by devices called *broadcasters* that transmit messages to listening devices called *observers* and, finally, *3. many-to-many communication*, where the devices use their broadcasting–observing capabilities to enable enhanced service models and complex networking. The latter mode is specified in the *Mesh Profile* [11].

In this section we describe the protocol stack of BLE and of its mesh profile, respectively. The two stacks are shown side-by-side in Fig. 2. The BLE stack, on the left, implements the first two communication modes while the stack on the right side is used for mesh networking.

[1] Our review work focuses on the security of BLE and of its mesh profile. Security aspects of Bluetooth Classic or of other technologies that are alternative to BLE were excluded from our sources, as we believe that BLE security has already generated interest and works to grant surveying and a new taxonomy. The methodology of our search for sources on BLE security include queries to the digital libraries of reputable professional associations such as the Association for Computer Machinery (ACM) and of the IEEE, using keywords such as *BLE Security*, *BLE Mesh Profile Security*, or more specific keyword reflecting particular security aspects of BLE (e.g., "KNOB attack on BLE"). We also searched the Internet at large. Most of our references, however, came from the References section of the many papers we surveyed.

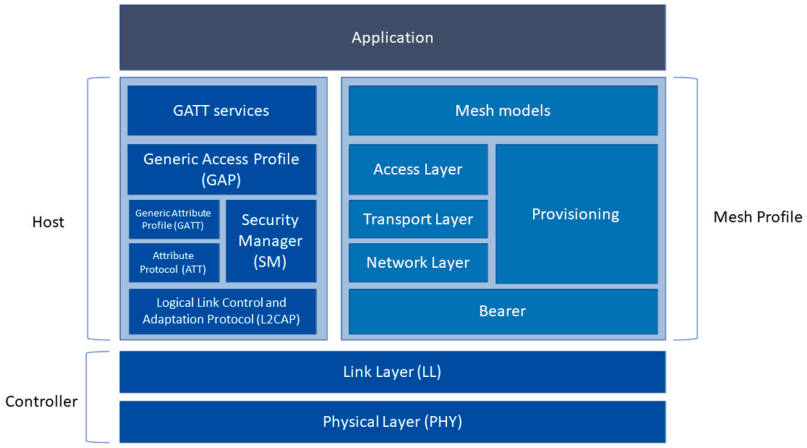

Fig. 2. The Bluetooth Low Energy (left) and Bluetooth Mesh (right) protocol stacks [12].

2.1. Bluetooth Low Energy

The protocol stack of BLE (right side of Fig. 2) is organized into two main parts, the *controller* (bottom) and the *host* (on top). The controller is in turn made up of two stacked layers:

- The Physical Layer (PHY), designed for very low power operation, transmits data over 40 channels of 2 MHz each in the 2.4 GHz unlicensed ISM frequency band. The channel is accessed using Frequency-Hopping Spread Spectrum (FHSS) and is modulated with Gaussian frequency-shift keying (GFSK).
- The Link Layer (LL) interfaces directly to the PHY. LL deals with advertising, scanning, creating and managing connections.

Controller and host are interfaced by an "hidden layer" called the Host Controller Interface (HCI), implementing communication between controller and host through a set of commands and actions.

The host part of BLE is organized into the following stacked layers.

- The Logic Link Control and Adaptation Protocol (L2CAP) is the first interface between upper layer protocols and the controller. L2CAP is in charge of channel multiplexing, i.e., to deliver the packets from the LL to the correct upper layer protocol during the setup of a channel, distinguishing also among different upper layer entities using the same protocol. L2CAP supports segmentation and reassembly of transport frames (L2CAP Service Data Unit or SDU). Starting from the Bluetooth Core Specifications Version 5.2 [13], L2CAP has also control over the Protocol Data Unit (PDU) size, which affords benefits including optimizing the interleaving of different data units to minimize latency. Another useful feature of L2CAP is the possibility to support controllers with limited transmission capabilities thanks to the fragmentation and the reassembly of the L2CAP PDU. Finally, L2CAP also implements tasks such as error control and support of QoS requirements. All these features are implemented through a non-negligible amount of code. This can be source of vulnerabilities due to bugs or bad programming habits (Section 4).
- The Security Manager Protocol (SMP) provides a framework for generating and distributing security keys between two devices and defines security requirements and capabilities through its own specific PDU fields (Section 3).
- The Attribute Protocol (ATT) is used by the Generic ATT profile (GATT) as a transport mechanism and as a mean of organizing data. As a generic data protocol ATT is used to store services, characteristics and related data via a lookup table using 16-bit IDs for each entry.
- The Generic Access Profile (GAP) manages device access methods and procedures, including device discovery, connection creation and termination, initiation of security functions, and device configuration.
- The Generic Attribute Profile (GATT) manages data exchange over a connection using ATT to exchange data. It also consists of services, features and descriptors that are collected and organized in the attribute table.

2.2. Mesh profile

In 2019, the SIG Alliance formally decided to standardize the support of mesh networking using the BLE LL and PHY as the foundation of the Mesh Profile (right side of Fig. 2). The new mesh capability implements many-to-many communication and is optimized for creating large-scale multi-hop networks of BLE devices. This kind of topology is suited for building automation, sensor networks and any solution that requires a large number of devices to reliably and securely communicate with one another.

According to the Mesh Profile specifications, BLE nodes can assume multiple roles at the same time. These roles correspond to different functions within the mesh profile layers. The roles are usually identified by the mesh features that a node can supports as defined by the mesh core specifications. In this survey we discuss particularly those roles that are possible targets of known attacks. For a description of all roles and details on the Bluetooth Mesh we refer the reader to the specifications [11]. The roles most relevant to this survey are:

- **Unprovisioned device**. This is one of two possible initial roles of every device. In this role the device is not a member of any mesh network.
- **Provisioner**. This is the device that is responsible of forming and managing a mesh network. Every BLE mesh network must have at least one provisioner. A schematic representation of how provisioning works is shown in Fig. 3. Provisioning messages are encrypted by the elliptic curve Diffie–Hellman (ECDH) key exchange algorithm over the FIPS-P256 curve [14]. To prevent man-in-the-middle (MITM) attacks the device can be authenticated during the provisioning process with a standard defined range of Out Of Band (OOB) authentication methods, such as passphrases or predefined secrets [9].
- A **Node** is a generic provisioned device. Nodes (e.g., lighting or temperature control devices, manufacturing equipment and electric doors) are devices that host a *Mesh Service Model*. These models represent the application layer of the network. According to their capabilities, they can be resource consumers or producers, namely *Mesh Clients* or *Mesh Servers*.

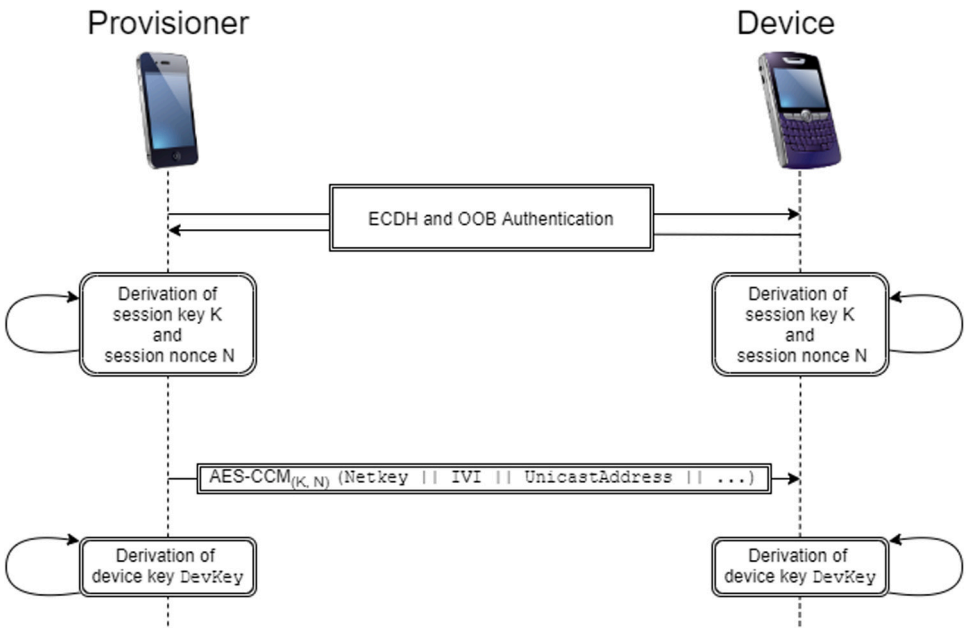

Fig. 3. Provisioning process in BLE mesh networks. This is the first step for each device to be part of a mesh network [9].

- **Relay Node**. Relay nodes can receive and forward messages, so the messages can be transferred further. Messages can be relayed multiple times, and each relay is considered a *hop*.
  Messages can hop up to 126 times as the standard stipulates that only 7 bits of the BLE Mesh PDU can carry hop information. This number is considered sufficient to transfer messages through a wide area.
- **Proxy Node**. Proxy nodes receive messages from an advertising or GATT bearer and resend it to another one. Their main purpose is to connect communication equipment that only support GATT bearer to a BLE mesh network, e.g., a BLE device without the mesh profile.
- **Low Power Node**. These nodes are characterized by reduced computational capacity and a limited battery. To minimize energy costs, low-power (LP) nodes do not remain always active and connected to the network. They follow a duty cycle, remaining inactive (i.e., in sleep mode) for most of the time and turning on their antenna only to send messages in a given time interval [15]. Every message to LP nodes, including security updates, is usually delivered to another special class of nodes, called *friends*.
- **Friend Node**. Friend nodes can store messages and security updates that are sent to LP nodes. The stored information will be transferred to LP nodes during its online phase. LP nodes and friends nodes must be paired through a process called *friendship* (discussed from a security standpoint in Section 3). Cooperation between LP and friend nodes optimizes the use of the radio and of the battery because LP nodes can receive messages without keeping listening. When on, an LP node polls its friend node to see if there are new messages. This concept of friendship suffers from design issues that may be exploited to disrupt networks functionalities (Section 4).

The mesh profile stack is topped by the *Mesh Models* that define and implement the functionality and the behavior of the features of a device. Models also define interfaces for the applications to use those features. Models are grouped into *elements* that indicate a set of functionalities of a device. Each node has at least one element (the *primary element*) and may have additional elements.

Messages must be sent to and from a specified address (point-to-point messaging). The mesh profile prescribes three different types of addresses, each 16 bit long.

- The **Unicast Address** is a unique address allocated to each device, assigned by the provisioner when a device joins the mesh network.
- A **Group Address** is a multicast address used for delivering messages to a specific set of devices. These addresses are used to identify a physical grouping of nodes such as all those within a specific zone (e.g., all lights in a room). There are two different kinds of group addresses: (i) *SIG-Fixed* group addresses are defined by the SIG, based on node roles (e.g., all-proxies or all-relays); (ii) dynamic group addresses are defined by the user via a configuration application.
- A **Virtual Address** is also a multicast address. BLE uses virtual addresses to support Universally Unique Identifiers (UUIDs), namely, globally unique 128-bit numbers that are used to identify profiles, services and data types. A virtual address is the hash of a UUID, which makes UUIDs more efficiently manageable. Virtual addresses may be assigned to one or more elements spanning multiple nodes. These addresses are usually configured during manufacturing.

Bluetooth mesh networks communicate via a client–server architecture using the aforementioned addresses. The function of a server is to expose the state of a mesh model whose value can be read or modified by the client. For further details on the Mesh Profile the reader is referred to the BLE specifications [11] and to dedicated surveys [16].

3. Security review of the BLE standard

From a security standpoint BLE features functionalities that can be enabled and combined on demand according to application requirements [3]. The degree of security of a specific BLE-enabled product is therefore case-dependent, being highly influenced by both intrinsic features of the product, like its usability and flexibility in a particular setting, and by extrinsic factors such as compliance with local standards, policies and implementation costs. In this section we provide an overview of the security of the version 5.1 of the BLE standard, which is arguably the prevailing version of BLE. We will also review some security features of the 5.2 Core Specification of BLE [4], so to provide the reader with an up-to-date, more complete picture of the current security state of the technology.

Table 1
Bird's eye view of the three different BLE security modes and their features.

Mode	Level	No security	Authenticated	Encrypted	Signed	Version	Description
1	**1**	Yes	No	No	No	4.0	Used mainly in central–peripheral communication.
	2	No	No	Yes	No	4.0	
	3	No	Yes	Yes	No	4.0	
	4	No	Yes	Yes	No	4.2	
2	**1**	No	No	No	Yes	4.0	Used for connection based data signing when a device or service needs fast and efficient data transfer.
	2	No	Yes	No	Yes	4.0	
3	**1**	Yes	No	No	No	5.2	Used in broadcast communications. In Level 3, communications are encrypted.
	2	No	No	Yes	No	5.2	
	3	No	Yes	Yes	No	5.2	

One of the main features of BLE is the possibility for devices to connect in ways that reduce energy consumption. Communications among devices can be *connection oriented*, which involves setting up persistent connections that include an initial pairing process, and *connection-less*, with no pairing between devices. The first type of communication is secured by exchanging keys that will be used to encrypt and authenticate the exchanged data and to authenticate the devices. Due to its intrinsic low power nature, connection-less communication enables lower levels of security and can be a possible target from malicious users that can perform impersonation attacks.

In a central–peripheral connection **security keys** are established and distributed during the pairing process. This enables security features such as *link encryption*, *privacy*, and *data signing*. There are three different keys in BLE:

1. Long Term Keys (LTK) are used for link encryption.
2. Connection Signature Resolving Keys (CSRK) are used to sign data over an unencrypted link.
3. Identity Resolving Keys (IRK) are used for privacy.

Once the keys have been distributed, BLE uses an authenticating encryption algorithm called AES-CCM so that packets exchanged over an encrypted link have their authenticity too. Such authentication helps preventing attacks like Man-in-the-Middle (MitM) and tampering.

Each BLE device can decide its own security requirements, such as encryption, authentication and data signing for both connection oriented and connection-less communications. For this purpose, the BLE core specifications define three different **Security Modes** for securing communications: *LE security mode 1*, *LE security mode 2* and *LE security mode 3* (the latter introduced in version 5.2). Any mode is a combination of security attributes and requirements that define the *Security Level* of a device, service or service request. Achievable levels are: *1. No security*, requiring no authentication, encryption or data signing; *2. Authenticated pairing*, which requires pairing with authentication of the two devices; *3. Encryption* of the connection using the keys exchanged during pairing, and *4. Data signing*, used for transferring authenticated data between two devices in an unencrypted connection. Table 1 summarizes the security modes and their levels.

Every physical connection between two devices can only adopt one security mode at time. Security Mode 1 is normally used in point-to-point communications. The others have been introduced for specific cases. Particularly, Security Mode 2 is only used for connection oriented data signing when a device or service needs fast and efficient data transfer. For this reason, Security Mode 2 is particularly suitable for devices with low computational power such as No-input No-output devices (NiNo). Security Mode 3 is used if the network needs to broadcast messages, which it does by forming a *Broadcast Isochronous Group (BIG)*. In this case, an *Isochronous Broadcaster* creates a BIG for one or more *Broadcast Isochronous Streams (BIS)*, namely, one-to-many data transport streams. The BIS logical transport is used to transport one or more isochronous data streams to all devices for a BIS within range. Data may be fixed or variable size, framed or unframed. Moreover, a BIS supports the transmission of multiple new isochronous data packets in every BIS event and each BIS can be received by an unlimited number of Synchronized Receivers [4]. Each BIS is part of a BIG. In fact, a BIG may have one or more BISes, all with a common timing reference based on the broadcaster (i.e., they are synchronized). Examples include the left and right channels of an audio stereo stream that are received by separate devices and need to be reproduced at the same time. Isochronous channels have been one of the main additions to the Bluetooth Core Specification 5.2 for enabling broadcast audio for applications such as audio sharing. Audio sharing enables an audio source device to broadcast one or more audio streams to an unlimited number of audio sink devices. Since audio sharing is an important use case of LE Audio, it needs to be secured. Security Mode 3 was introduced for this purpose.

Depending on the selected mode, pairing will be performed by choosing the association model that meets the security requirements of the security mode. **Pairing** is the most fundamental feature of BLE and of its security. Pairing allows two devices to associate to create a link or a network. It enables different security features. Two devices that store the resources exchanged with pairing, including those related to security, are said to be **bonded**. If two bonded devices want to establish a new connection, there are two possibilities for encryption and authentication: The first option, called *proactive authentication*, implies that the central device enables encryption and authentication before sending any request to the server, using the Long Term Key established during pairing. The other option, called *reactive authentication*, can happen immediately after the re-connection where the central device sends an attribute access request (such as Mode 1 Level 1) to the peripheral and accessing the attribute requires an higher encryption and authentication level. In this case the peripheral will send back an error message asking to the central device the authentication and encryption level required.

Pairing can be done with different degrees of security. The fundamental points of the process are:

1. Generation and distribution of keys that enable the use of required Bluetooth security features such as encryption, data signing and privacy.
2. The distribution of up to three types of keys between the two devices must be protected from passive eavesdroppers who may attempt to steal the keys being distributed.
3. Authentication of devices during pairing to protect against attacks such as MitM attacks and tampering.

The maximum level of security is obtained when network devices support all security features defined by pairing. The version 4.2 of the standard introduced the LE Secure Connection method in addition to the already existing LE Legacy Pairing method.

Each method is made up of the so called *association models*. These models decide how pairing should proceed and, where possible, how authentication should be done depending mainly on which security mode is required by the peripheral. These models are: Just Works, Passkey Entry and OOB for LE Legacy Pairing and Just Works, Passkey Entry, Numeric Comparison, and OOB for LE Secure Connection. Consider, for instance, the scenario where a central device would like to

Table 2
SMP PDU fields.

SMP PDU fields	
Secure Connection (SC)	Bit indicating if the device supports LE Secure Connection. If SC is set the device must use LE Secure Connection. If both devices support LE Secure Connection, this method must be used.
Man in the Middle (MitM)	If this bit is set to 1 then authentication is required (by both devices or neither).
Out Of Band (OOB)	Out of Band data used for authentication by mechanisms that do not use Bluetooth between two devices (e.g., QR codes).
Maximum Encryption Key Size	Denotes the maximum encryption key size that a device can support. A key can range from 7 to 16 octets (56 to 128 bits). Each of the two devices communicates the value of the Maximum Encryption Key Size to the other device and the smaller value will be chosen (since both devices must use the same key length).
Initiator Key Distribution and Responder Key Distribution	They indicate the types of key that the Initiator would like to provide and the types of key required by the other device.

connect to a peripheral device that requires Mode 1 Level 3 for encrypted and authenticated communication. The central device connects and pairs with the peripheral device choosing the pairing method to use between LE Legacy Pairing and LE Secure Connection. Subsequently, an association model that generates an LTK to encrypt the connection and authenticates the central device is chosen (i.e., Passkey Entry). After the pairing process is successfully completed, the central device uses peripheral services and immediately starts encrypting the connection with the previously generated LTK. LE Secure Connection is the safest of the two and differs from LE Legacy Pairing in two ways. LE Legacy Pairing uses a simple method to exchange secret data with which to derive the key that will encrypt the link during the key distribution phase. LE Secure Connection, instead, uses elliptic curve public key cryptography to allow a secure derivation of the secret key that will always be used to encrypt the link during the key distribution phase. Moreover, LE Secure Connection has an additional association model, namely, Numeric Comparison.

To handle pairing and key distribution, the specifications define the *Security Manager Protocol (SMP)*. This protocol uses specific PDUs with fields for different types of security requirements and capabilities. The SMP PDU fields are described in Table 2. SMP PDUs are not used for broadcast purposes since this type of communication is connection-less and enables lower levels of security requirements.

The rest of this section is devoted to describe the phases of the pairing process in details (Section 3.1, Section 3.2 and Section 3.3) and an overview of the security features of the BLE mesh profile (Section 3.4).

3.1. Pairing feature exchange

The pairing process always starts with the Pairing Feature Exchange procedure, started by a device called the *initiator* looking for *responders*. The procedure consists of the exchange of an SMP Pairing Request (sent by the initiator) and of an SMP Pairing Response (sent by a responder) whose fields include IO Capability, SC, MitM, Maximum Encryption Key Size, Initiator Key Distribution and Responder Key Distribution (Table 2). This exchange provides each device with information to decide:

1. The pairing method between *LE Legacy Pairing* and *LE Secure Connection*.
2. Whether to perform the authentication process between the two devices and, if so, what type of authentication method. In addition, they determine which keys should be generated and distributed.
3. The length of the Long Term Key. This is then distributed together with the CSRK and/or the IRK (if required).

The exchange also allows each device to inform other devices about their security capabilities.

Pairing then proceeds depending on which pairing method is selected between *LE Legacy Pairing* and *LE Secure Connection*. Both methods are described next.

3.2. LE Legacy Pairing

If either of the two devices has not set the SC (Secure Connection) flag, LE Legacy Pairing is chosen as pairing method. LE Legacy Pairing is a three phase process. Phase 1 concerns deciding the association model. The way in which an association model is chosen over another depends on the IO capabilities of the device and the other set flags described in the "Pairing Feature Exchange" section. The association models available for LE Legacy Pairing are:

1. **Just Works**: there is no user interaction. This model is used if the MitM flag is set to 0 and if neither device or one of the two has the SC flag not set, or if both devices have the OOB flag set to 0, one or both devices have the MitM flag set to 1 and the IO capabilities of the passkey entry are not supported.
2. **Passkey Entry**: unlike Just Works this model requires minimal user interaction and the IO capability of one of the devices to display a number of at least 6 digits. If one of the two devices has this capability, the user must necessarily manually enter the 6-digit number in the other device (this implies that the other device must be equipped with input capabilities).
3. **OOB**: consists in exchanging data through a channel that does not use Bluetooth. Each product has different OOB channels depending on the vendor. The latter must also evaluate the security of the channel based on security requirements (usually a chosen OOB channel is resistant to MitM attack).

The main purpose of Phase 2 is to generate the Short Term Key (STK) and authenticate the two devices to ensure security against MitM attacks. The STK is used to encrypt the communication for key exchange during Phase 3 and is not reused. This phase is divided into three steps:

1. Generation of the Temporary Key (TK), a 128-bit key created according to the association model chosen in Phase 1. For instance, if Passkey Entry was chosen then the value will be the 6 digits number exchanged between the two devices plus a padding of zeros to reach 128 bits overall. The TK is one of the three final elements to derive the STK.
2. Authentication between two devices to avoid MitM attacks by proving that both devices know the TK (Fig. 4).
3. Creation of the 128-bit STK key to encrypt the link layer.

From this moment on, the channel is encrypted with the STK which will be used as Session Key to secure each BLE link layer packet.

Phase 3 consists in sending all the parameters defined during Phase 1 by the pairing feature exchange. These include the length of the keys and one or all among the Long Term Key (LTK), the Connection Signature Resolving Key (CSRK) and the Identify Resolving key (IRK). These keys are sent over the link encrypted by the STK created in Phase 2. In addition, the so called values EDIV and RAND, which act as identifiers for the LTK in the security database, are distributed. The STK

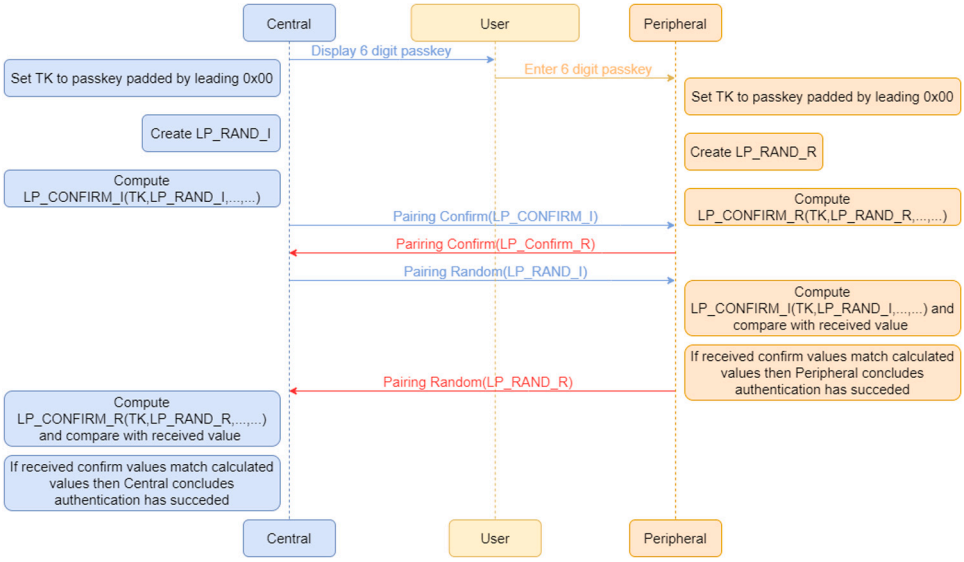

Fig. 4. LE Legacy Pairing authentication.

will be used until the time of a new reconnection. From that moment on the LTK will be used. The time between the first connection and the others can be longer or shorter. When the reconnection occurs after a long time, it involves a vulnerable channel due to the lack of robustness of the STK (with entropy 0 in the case of Just Works).

3.3. LE Secure Connection

LE Secure Connection is considered a much more secure method than LE Legacy Pairing. This method is used if both devices support it and if the SC flag is set. Similarly to LE Legacy Pairing, LE Secure Connection proceeds in three phases. Phase 1 is similar for both methods but for small differences concerning the association models. Particularly, this method adds a new association model called *numeric comparison*. In this model, a six digit random number is generated and displayed on both devices. The user performing the pairing is asked to confirm that the numbers displayed by both devices are the same by pressing a *yes* or *no* button. The LE Secure Connection Passkey Entry module has two variants:

1. The user inserts a six-digit passkey into both devices.
2. One device displays a six-digit passkey and the user inserts it into the other device (as in LE Legacy Pairing).

In Phase 2, LE Secure Connection uses elliptic-curve cryptography with the P-256 elliptic curve [14], allowing the secure exchange of data that will be used to derive a symmetric key called the Long Term Key (LTK). This is then used to encrypt the link that will be used for key distribution in Phase 3. Phase 2 happens in 4 steps: Public Key Exchange, Calculate DHKey, Authentication Stage 1 and Authentication Stage 2. Authentication stages 1 and 2 are used to authenticate the two devices where, unlike LE Legacy Pairing, the data entered by the user during authentication stage 1, are not used to derive the cryptography keys with the various levels of entropy (and therefore security), which this approach entails. Finally, both devices have the possibility to generate an ECDH public–private key pair. It is recommended practice to change the private key at each pairing, whether the pairing was successful or not, even if it is not mandatory [3].

- Step 1. The initiator of the pairing process sends his public key to the responder who in turn will respond with his own. To do this, the SMP Pairing Public Key PDU is used. When both devices receive the public key, they will validate it by checking that the two public keys are on the same elliptical curve.
- Step 2. The two devices, A and B, calculate the Diffie–Hellman key as follows:
  A: $DHKey = P256(SK_a, PK_b)$
  B: $DHKey = P256(SK_b, PK_a)$
  where SK_a and SK_b are the private keys of device A and device B respectively while PK indicate their respective public keys. The strength of this operation is that both devices have the same DH value (a shared secret) without having exchanged the private key in any way.
- Step 3. Authentication stage 1 allows the user to check if the device he is pairing with is really the device he wants to pair with. This process changes according to the association model chosen during Phase 1. Fig. 5 shows an example of authentication stage 1 for numeric comparison. Just works does not go beyond the Cb check while Passkey entry and OOB use slightly different modes and values. In case Passkey entry has been selected, both devices must enter the same passkey at the beginning of the procedure. While for OOB, the process is divided into an Out of Band part (information exchange procedure not performed by Bluetooth such as NFC, QR codes, or some other approach) and an In Band part (procedure performed through Security Manager Protocol PDUs).
- Step 4. Authentication stage 2 aims to confirm that all the values exchanged by the two devices in the previous stages match and are correct. Such stage is divided into 4 steps:

  1. **Calculation of LTK and MacKey**: each device calculates the LTK and MacKey using parameters that both devices have such as the DHKey computed in step 2, random values exchanged during authentication stage 1 and the devices addresses. If this process goes well, both devices will end up with the same LTK and MacKey;
  2. **Check** E_a: at this point, device A generates a check value called E_a using parameters common to both devices so that B can recalculate E_a on its side. In fact A after calculating E_a, sends it to B who will recalculate it and compare it with the one sent by A. If they do not match the pairing is aborted;
  3. **Check** E_b: if E_a's check was successful, the process is repeated by B who will send E_b to A who will recalculate and compare it with E_b sent by B. If the values do not match, the pairing is aborted;
  4. **Encryption with LTK**: finally, the link to create a session key can be encrypted by LTK;

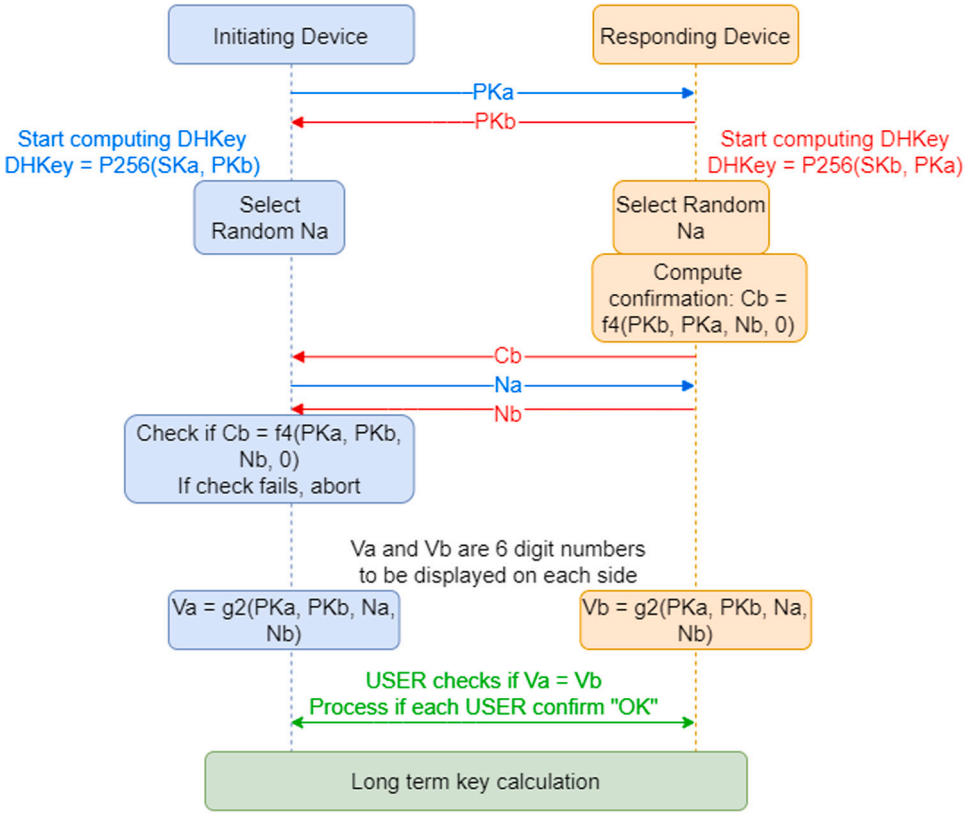

Fig. 5. LE Secure Connection authentication stage 1 for Numeric Comparison.

As for LE Legacy Pairing, Phase 3 is dedicated to sending all the parameters defined in Phase 1 and by the pairing feature exchange. IRK and CSRK can be sent during this phase and, unlike LE Legacy Pairing, the Bluetooth device address is used as the LTK identifier in the security database.

3.4. Mesh Profile security

As typical with the new generations of wireless protocols, securing message exchange in a BLE mesh network should be granted by design and by default [11]. As described before and as depicted in Fig. 3, new devices added to a BLE mesh network can be provisioned using 256-bit elliptic curves and out-of-band authentication, while the PDU of mesh packets is secured using AES-CCM with 128-bit keys, ensuring that all messages in the mesh are encrypted and authenticated.

Mesh operations and communications are secured by the following keys. The first one is the **Network Key (NetKey)**, which is used to secure messages across all the nodes in the same network. The second one, called **Device Key (DevKey)**, is a special application key that is unique to each device. This key is used to encrypt *Configuration Messages*, namely, the messages exchanged between the Provisioner and the device being provisioned and configured. The third key concerns securing the Transport Layer by the **Application Key (AppKey)** that is used for the encryption and decryption of application data. Application keys are bound to NetKeys. This means that application keys are only used in the context of the network keys to which they are bound. An application key can only be bound to a single network key. Finally, messages through the network are encrypted with a **Master Security Material** that is derived from the NetKey. Messages encrypted by the master security material can be decoded by any node in the same network.

When the end-of-life of a device occurs that prompts its removal from the network, the device is added to a black list and the **Key Refresh** procedure is started. This entails updating the network and application keys of all nodes with the exception of those in the black-list. Low-power nodes will receive the new keys from their friend. Because of duty cycling this process may take some time.

The Key Refresh procedure performs three steps to replace the entire set of keys. (1) Distribution of new keys to each node. Nodes can continue to broadcast using the old keys but they will be able to receive using both old and new keys. (2) In the second phase, a Secure Network beacon is transmitted that signals to the network that the new keys have been distributed. Nodes will use the new key to transmit but will continue to use both old and the new keys to receive. (3) The third phase concerns the transmission of another Secure Network beacon that signals to all nodes that they should revoke the old keys. From now on, the nodes will transmit and receive using only the new keys.

Therefore, during the initial stages of Key Refresh, nodes use two keys simultaneously. Once the old keys have been revoked the nodes return to normal operations using the new keys. This procedure is meant to avoid using the keys left in the device removed from the network to compromise the network, an attack known as the “trash-can attack” [17].

The security policies of the BLE mesh standard explain that nodes are equipped to countermeasure the most common attacks to mesh networks. For instance, there is a procedure to update the security keys in case one of nodes is leaving the network or is attacked and deprived of its keys. There is a black list that includes nodes whose keys are compromised. Such list is maintained by the Provisioner, which is also in charge of adding such nodes to the list and to initialize Key Refresh Procedures to force the replacement of the entire set of security keys. In this way, compromised devices will not get new credentials and will no longer be members of the network. However, the mesh profile specifications do not provide tools or procedures to make the network aware of an attack or about whether its security has been compromised.

Another simple yet powerful security procedure is the use of a sequence number *SEQ* to prevent *replay attacks*, where an attacker passively captures a message and sends it later without modification.

By doing so the attacker tricks the victim into accepting a duplicate message that could produce a malicious result. Every device increases the sequence number for each new message that it sends so that each mesh message is sent with a unique pair of sequence number and source address. When receiving a message, the recipient device stores the sequence number and makes sure that it is more recent than the last one that it received from the same source address. Otherwise, the message is discarded. However, sequence numbers can only takes $2^{24} = 16777216$ values. Given a device transmitting a new message at 2 Hz, its sequence number space would be exhausted after 97 days [9]. To overcome this limitation, a 32 bit **Initialization Vector (IV) Index** is used and must be keep updated through time. The IV Index value is related to the random number required for message encryption. To ensure that the value of the random number is not repeated, this value is incremented every time the *SEQ* number is exhausted. All the nodes in a mesh network share the same IV Index value. Starting from zero (in hexadecimal format), the IV Index increases during the IV Update procedure and is maintained by a specific process, ensuring that the IV Index shared in the mesh network is the same. This can be done when the node believes that it is at risk of exhausting its sequence numbers, or when it determines other nodes are nearly exhausting theirs. The standard mandates that the update time should not be less than 96 hours [11].

Concerning the security of the *friendship establishment* procedure, communication between Friend nodes and LP nodes takes advantage of two security credentials.

The first one is called *Managed Flooding security material* and is the *master security material* mentioned above. It is derived from NetKey and from a function called *k2* [11]. This function generates the master security material formed by instances of *EncryptionKey*, *PrivacyKey*, and Network ID. *k2* uses the MAC function AES-CMACT with a 128-bit key T. The values in input to the $k2$ function are: N (128 bits) and P (1 or more octets). The key T is computed as:

$$T = AES - CMAC_{SALT}(N),$$

where SALT is 128-bit value computed by the *s1* function defined in Section 8.1.1 of the Mesh Profile standard [11]. The output of the key generation is:

$$T0 = empty\ string(zero\ length)$$

$$T1 = AES - CMAC_T(T0 \parallel P \parallel 0 \times 01)$$

$$T2 = AES - CMAC_T(T1 \parallel P \parallel 0 \times 02)$$

$$T3 = AES - CMAC_T(T2 \parallel P \parallel 0 \times 03)$$

$$k2(N, P) = (T1 \parallel T2 \parallel T3)\ mod\ 2^{263}$$

Based on the functionality of $k2$, the master security material denoted by NID, Privacy Key and Encryption Key, is defined as:

$$NID \parallel EncryptionKey \parallel PrivacyKey = k2(NetKey,\ 0 \times 00)$$

In the communication between LP nodes and friend nodes some of the messages are encrypted and decrypted using the *Friend Poll* used to initialize the friendship and to notify that the LP node is ready to take updates from the friend, the *Friend Update* to notify that is possible to get new data from the LP node, and the *Friend Clear* to end the friendship between the nodes and its confirmation message (*Friend Clear Confirm*). For the remaining operations, the ***friendship security credential*** is used instead. It depends on both NetKey and the $k2$ function and on four additional parameters that are exchanged during the establishment of a friendship. These are the addresses and the counters of both LP node and friend node.

$$Friendship\ Security\ Material = NID \parallel Encryptionkey \parallel PrivacyKey = k2(NetKey,\ 0 \times 01 \parallel LPNAddress \parallel FriendAddress \parallel LPNCounter \parallel FriendCounter)$$

The LPNCounter is the value exchanged in the Request and the FriendCounter is the value exchanged in the Offer.

4. Threats analysis

After introducing the standard and its fundamental security aspects, in this section we analyze the main attacks that are affecting the BLE protocol. While Bluetooth is generally believed to be a secure protocol [18], during the years a lot of design vulnerabilities have emerged. One common misconception about this protocol which supports the idea of BLE as *perfectly secure*, is the poor presence of hardware in the wild that allows the traffic eavesdrop unlike WiFi protocol. Currently, even the most established network analysis tools such as Wireshark do not provide any support for parsing BLE packets. However, the work in [19] demonstrates how is possible to read and decode a BLE communication using dedicated hardware such as Ubertooth.[2] The authors, after some preliminary evaluation of the capability of intercepting BLE communication, they pointed out that the version of Bluetooth used by the devices determines the level of security of the communication. In fact, older versions of Bluetooth (especially those below version 4.2) do not provide the same security features that enable a secure communication. They also claim that a vulnerability affecting potentially all Bluetooth versions is the improper storage of connection keys which allows attackers to view and potentially alter them.

The vulnerabilities that affect BLE-based devices depend on various factors such as the version of BLE stack and the development choices of the manufacturer. In general, to avoid additional development costs, vendors prefer Just Works as the desirable pairing method, which makes the device susceptible to MitM attack.

As it has been pointed out in [2], the decision not to be compliant with the standard, and thus not to develop the security requirements discussed in Section 3, can be the reason of major security flaws.

Some of these vulnerabilities, presented in [20], affect mainly older BLE versions and they are unpatched in the most widespread of BLE commercial devices as referred by authors. Other vulnerabilities are presented in [21] and to extend this work, the NIST's document [22] presents many known security threats associated with Bluetooth. Additionally, the authors of [23] have identified a list of attacks applicable to BLE devices based on NIST's own documentation.

Despite these issues, Bluetooth Low Energy is an integral part of Internet of Things (IoT) environments which have a huge potential to simplify and improve everyday life.

Whether because of their ability to transfer data or their ease of access, IoT protocols tend to be more suitable target for criminals [24], raising the need for protocols like BLE to have a constant attention over their security features. In general, many vendors have developed plenty of frameworks to manage their BLE ecosystems which is most of the time challenging. In this cases, the possibility of security gaps being left open is very high. In this regards, [25] describes the security features of the various IoT platforms.

To effectively present all the major threats affecting BLE protocol, we propose a taxonomy shown in Fig. 6. The main motivation under this classification is to create a model that can efficiently identify and briefly describe every known attack, and furthermore, can be expandable over time.

The main characteristics used to categorize attacks are: Target entity, Layer, CIA elements, Version Affected and the Topology on which the technique can be performed.

[2] http://ubertooth.sourceforge.net.

Table 3
Categorization of known BLE attacks according to the proposed taxonomy. All the vendors patches are listed in [20]. * Mesh Profile Specification.

Name	Target	Location	Compromise	Version	Known mitigation	Topology
MitM	GATT	GATT	Confidentiality	4.0+	ML method in [26]	Single
Zero LTK Installation	Long Term Key	Link	Confidentiality	5.0, 5.1	Vendors Patches	Single
DHCheck Skip	DHCheck	Link	Confidentiality	5.0, 5.1	Vendors Patches	Single
BLE Co-located Application Attacks	GATT	GATT	Integrity	5.0	Security measures for data at higher layers [27]	Single
KNOB attack	Long Term Key	Link	Integrity	5.0	Min. 16 bytes of entropy [28]	Single
DoSL	Energy	Link	Availability	4.0	Mitigated in v.4.1	Single
Battery exhaustion	Energy	Link	Availability	4.0	Mitigation in [29]	Single
Silent Length Overflow	LL Length	Link	Availability	4.2	Vendors Patches	Single
Invalid Connection Request	Conn. Request Packet	Link	Availability	4.1, 5.0, 5.1	Vendors Patches	Single
Unexpected Public Key Crash	Public Key address	SMP	Availability	5.0, 5.1	Vendors Patches	Single
Sequential ATT Deadlock	Conn. Event	ATT	Availability	5.0, 5.1	Vendors Patches	Single
Key Size Overflow	Key size	Link	Availability, Integrity	5.0, 5.1	Vendors Patches	Single
HCI Desync Deadlock	Wrong packet numbers	HCI	Availability	5.0, 5.1	Vendors Patches	Single
Invalid Sequence Memory Corruption	NESN and SN bit	Link	Availability	5.0, 5.1	Vendors Patches	Single
Invalid Channel Map Crash/Deadlock	Channel Map Field	Link	Availability	5.0, 5.1	Vendors Patches	Single
Link Layer Length Overflow	LL Length	Link	Integrity	4.2, 5.0, 5.1	Vendors Patches	Single
Link Layer LLID deadlock	LLID field	Link	Availability	4.2, 5.0, 5.1	Vendors Patches	Single
Invalid L2CAP fragment	PDU size	L2CAP	Availability	4.1	Vendors Patches	Single
Jamming	Signal radio	Physical	Availability	All	Mitigation in [30]	Single
Btlejack attack	BLE supervision timeout	Link	Confidentiality, Integrity	4.0+	Secure connection	Single
Selective Jamming	Signal radio	Physical	Availability	All	Mitigation in [31]	Broadcast
Spoofing attacks	Pairing	Link	Confidentiality	5.2	Updating Implementation, Specification [32]	Broadcast
Simple Power Analysis	Keys	Physical	Confidentiality	1.0+*	No Mitigation	Mesh
Differential Power Analysis	Keys	Physical	Confidentiality	1.0+*	Masking	Mesh
Fault Attack	Keys	Physical	Confidentiality	1.0+*	No Mitigation	Mesh
Selective forwarding attack	Selected messages	Network	Availability	1.0+*	No Mitigation	Mesh
Black hole attack	All messages	Network	Availability	1.0+*	No Mitigation	Mesh
Malformed TTL attack	TTL value	Network	Availability	1.0+*	No Mitigation	Mesh
Collection materials	Friendship security materials	Network	Confidentiality	1.0+*	No Mitigation	Mesh
Exploit Pool message policies	Poll message PDU	Network	Availability	1.0+*	No Mitigation	Mesh
Clear message attack	Friendship	Application	Availability	1.0+*	No Mitigation	Mesh

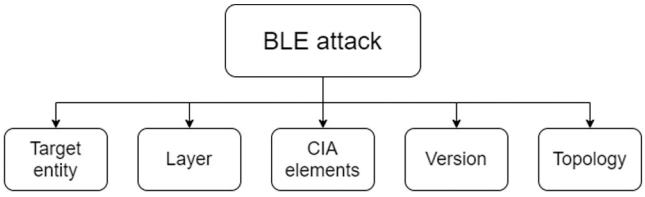

Fig. 6. Taxonomy of a BLE attack.

With **Target entity** we intend the standard specification that has been exploited to carry out the attack. A lot of targets have been identified as vulnerable in the past and having record of this can help hardware manufacturers and security experts to put the appropriate focus to them while evaluating the risks. An example of a target entity can be the pairing process or the data exchanged in the GATT.

The **Layer** section is used to identify in which layer of the BLE stack the attack is performed. While the majority of attacks take place in the Link Layer, the release of the Mesh Profile opened a lot of possibilities for malicious users and identifying which layer is affected is important to improve the fallback or mitigation procedures.

The **CIA elements** indicate which elements of the CIA triad, namely, *Confidentiality*, *Integrity* and *Availability*, have been compromised. Specifically:

- *Confidentiality* prevents sensitive data from disclosure to unauthorized individuals. Even if in a BLE connection the data can be encrypted, ensuring data confidentiality against attackers requires that two devices that are initially not connected must first authenticate their identity, encrypting the link, and then distribute the keys used for encryption.
- *Integrity* refers to methods of ensuring which provide that data is real, accurate and safeguarded from unauthorized user modification. In the context of this survey, Integrity ensures that the IoT devices and Information cannot be read and used by someone unauthorized.
- *Availability* means that the information and computing resources are readily available when required by a service. Whenever an IoT device detects a physical parameter, the system must store and process the data to function properly on the communication channels [33].

The triad is included in the taxonomy as it is an established tool in cybersecurity that is used to deal with a vast range of threats, from those affecting the first protocols of the Internet to the most recent technologies for IoT [34].

Topology identifies the architecture of the network with which the target devices communicate. There are three main types of topology:

- A *single connection* topology represents a network architecture in which there is a single-connection between a master and each of one or more devices. This is a topology used for connection-oriented communications.
- *Broadcast* is the type of topology with one broadcaster and multiple receivers. This topology is used for connection-less communications.
- *Mesh* is a topology made up of multiple nodes each of which can communicate with any other node either directly or via intermediate nodes. This type of topology enables large-scale multi-hop networks of Bluetooth devices.

Finally, since a lot of issues have been neutralized with the major updates of the standard, is important to understand the potential of an attack the **version** of the BLE protocol that is afflicted by it.

Table 3 presents an implementation of the taxonomy proposed using the attacks described in this survey. Such attacks have been divided following the three different communication modes that BLE can have, in accordance with the topological characteristic described above. At the end of each presented attack a possible mitigation is proposed and it is explored more in depth in Section 5.

4.1. Attacks on single connections

Jamming: in this threat, the attacking device tries to change the transmitted signal or completely jam it by sending a signal causing interference into communications between two other nodes [35]. According to [30], different types of jamming can be distinguished:

- *continuous* jamming, where the attacker fully occupies the channel without any brake;
- *deceptive* jamming, where the attacker repeats a correct signal that is being transmitted from another device, thus increasing the overhead rate of the shared medium;
- *random* jamming, where the attacker starts broadcasting at random times and transmits for a random period of time, disrupting low latency communications and hardening its detection from a watchdog;
- *reactive* jamming, where the attacker listens to the activity in the given channel and starts broadcasting when it registers a signal;

By its nature, an effective jamming attack is quite expensive from the point of view of the energy due to the multiple transmissions required.

Mitigation: there are several ways to counter this type of attack:

- *Regulated transmitted power* this is to use more power to transmit the signal to communicate, jamming will be more difficult because a stronger noise signal is needed to obstruct the original signal;
- *Frequency-Hopping Spread Spectrum* the frequency hopping spread spectrum (FHSS) is used to transmit radio signals by switching a carrier in between many frequency channels. This happens quickly, benefiting from a shared algorithm between the transmitter and the receiver;
- *Direct-Sequence Spread Spectrum* direct-Sequence Spread Spectrum (DSSS) transmissions are performed by multiplying the data (RF carrier) transmitted and a digital pseudo-noise (PN) signal at a much higher frequency than the original. This process causes the RF signal is replaced by a very wide bandwidth one with the spectral equivalent of a noise signal. However, the noise is filtered at the receiving end to recover the original data. This original signal processing by DSSS also makes it difficult for the attacker to decode the transmitted RF carrier to recover the original signal;
- *Hybrid FHSS/DSSS* hybrid communication FHSS/DSSS represents the desired anti-jamming measure. Generally speaking, direct sequence systems achieve interference mitigation by using a wider bandwidth to transmit the signal, while FHSS through interference prevention. So hybrid FHSS/DSSS develops the robustness to counter the near/far problem in DSSS communication schemes.

Btlejack attack: the Btlejacking technique,[3] intercepts and takes advantage of the BLE supervision timeout in CONNECT_REQ PDU to detect a connection and works with BLE versions 4.0 and later. This timeout defines the time after which a connection is considered lost if there are no valid packets and is used by both the central and the peripheral. After intercepting the connection, it jams to interrupt the communication between the control unit and the peripheral. Finally, it hijacks the aborted connection to connect to the peripheral and consequently access the device. This attack compromises the integrity and confidentiality of the device data.

Mitigation:As suggested by the author in [36], the only possible countermeasure is to use BLE Secure Connections with the packet injection protection option.

BLE Co-located Application Attacks: this class of attacks is detailed by [27]. The authors, present BLECryptracer, which consists of identifying the presence of an application-layer security that stands between the BLE device and the android application. In this scenario, there are basically two BLE applications, a benign application that performs BLE pairing, and a malicious application that should not be able to access BLE data, but this attacks demonstrates that the BLE credentials are implicitly available to all applications on the device. In this way, unauthorized application can access protected association attributes for the BLE device even if the pairing process is started by another application since this data is saved on the device and accessible to all applications, instead of being only for those that require specific permissions to use BLE.

This guarantees to the attacker the same access level to paired-protected data on the device such as the official app, with the difference of not having to start the association and furthermore, the user is not aware of this unauthorized access when it takes place.

For this reason, BLECryptracer uses taint analysis techniques to analyze all calls to the BLE setValue and getValue methods to check if the data read from the peripheral and written to the controller are cryptographically processed.

Mitigation: this vulnerability arises due to design problems and a solution could be to modify the Bluetooth specification and insert specific protections for data at higher levels. This, however, would require changes to all devices within the ecosystem, which can potentially lead to fragmentation and reduced interoperability, as making specific changes to each device could make them no longer compatible to each other. The work in [27] describes several solutions for applications developers and for devices developers too. Developers are advised to implement end-to-end security between applications and BLE firmware, and they exhort manufacturers to implement permission levels on read and write requests to the protected characteristics.

Unexpected Public Key Crash (CVE-2019-17520)[4]: this weakness is present in the legacy pairing procedure, explained in Section 3, which is managed by the SMP. This vulnerability leads to a peripheral crash if SMP public key is sent before the starting of the SMP pairing procedure and the key is not ignored. The device attempts to copy the package in a null target address causing the device crash which must be necessarily restarted manually.

Sequential ATT Deadlock (CVE-2019-19192): this attack consists in sending two consecutive packets of ATT request so as not to give the recipient time to respond to first request and then send it into deadlock.

Invalid Channel Map Crash/Deadlock (CVE-2020-10069 and CVE-2020-13594): in this vulnerability, the device accepts packets that have the channel map field set to null during the connection. In this case, the connection fails but the raised error does not allow the peripheral to resume the advertisement process, resulting in a device crash.

Link Layer LLID deadlock (CVE-2019-17061 and CVE-2019-17060): The availability of BLE devices is compromised, requiring the user to manually restart the product to re-establish BLE communication. The device enters a faulty state if it receives a packet with the Link Layer Identifier (LLID) field cleared. This issue is only exposed when the user tries to connect to the peripheral thanks to a never ending connection or pairing process in the link layer due to receive LLID field cleared.

HCI Desync Deadlock (CVE-2020-13595): due to the return of a wrong packet number in a corner-case scenario during HCI event, an attacker can disable Bluetooth advertisement feature for peripheral devices. To exploit this bug, the attacker must send a specific invalid packet, causing a Message Integrity Check (MIC) error on the connection event. In this way the device tries to recover packets resetting the stack due to de-synchronization and in turn, compromising the availability.

[3] https://github.com/virtualabs/btlejack.

[4] The Common Vulnerabilities and Exposures (CVE) is a kind of dictionary for publicly known security vulnerabilities and exposures. All vulnerabilities have a unique identifier and are listed in the MITER system and in the US National Vulnerability Database.

Invalid Connection Request (CVE-2019-19193): an invalid connection request from central device to peripheral could lead the peripheral device to a deadlock state, therefore user have to manually restart its. This happens when some devices do not correctly handle some request connection parameters that are contained in Adv Ch. Payload of the Data Channel Packet, such as the connection interval or timeout fields (called respectively interval or timeout) equal to zero, during a pairing attempt at the link layer level.

Invalid Sequence Memory Corruption (CVE-2020-10061): the attacker first sends to the device an Anchor Point packet with bits $NESN$ (Next Expected Sequence) and SN (Sequence) set to 1, which causes an invalid operation on the victim's internal packet buffer. Subsequently if the attacker continues to send further packets it fills the victim's retry-buffer and then the victim device suffers memory corruption and possible crash, while connecting in the link layer.

Invalid L2CAP fragment (CVE-2019-19195): the attacker sends a packet with a improper PDU size causing the cutting of the L2CAP header which can lead to a deadlock of the device during the communication. This vulnerability is extremely dangerous because forces the victim to perform a reboot from remote, compromising its availability.

Silent Length Overflow (CVE-2019-17518): some peripherals respond to packets with unexpectedly large LL (Link Layer) Length field. When such packets are sent, the target stops responding and crashes. This behavior indicates that a buffer overflow has occurred for some types of packets such as pairing request in the link layer. A remote execution scenario is also possible due to this vulnerability.

Truncated L2CAP (CVE-2019-17517): this vulnerability consists in performing a buffer overflow by sending a legitimate L2CAP payload but declaring a different size in bytes of it, paving the way for the attacker to choose arbitrarily how many bytes can be overwritten, to perform an overflow of the L2CAP reception buffer. This results in a loss of availability due to the denial of service and in the worst case the attack could also execute remote code potentially compromising permanently the device and the entire BLE network.

Key Size Overflow (CVE-2019-19196): This weakness is caused by two issues encountered during pairing procedure of devices which employ a particular SMP implementation. The attack is a buffer overflow that crashes products with pairing support enabled. This attack consists in sending a key to the peripheral with a length of up to 255 bytes during the link layer encryption procedure (first issue, because the maximum encryption key size is within 7 to 16 bytes) and despite this problem, the peripheral rejects the pairing without abnormal behavior. As a consequence, the victim accepts that the LL encryption procedure is performed before the pairing procedure, even if this fails in a later stage (second issue). These two problems lead to a buffer overflow since in this way it is possible to force the victim into allocating the over sized key buffer length and when the pairing request is accepted the overflow occurs with the firmware that tries to allocate the key. Although this vulnerability does not expose easy control on the buffer overflow, it could be possible to bypass the encryption and then the attacker may steal user information.

Link Layer Length Overflow (CVE-2019-16336, CVE-2019-17519): in this weakness the attacker can perform a buffer overflow by manipulating the Link Layer Length Field which is padded with much more bytes than expected and so more bytes then expected are allocated in memory causing as usual a denial of service in the first stance. Thanks this weakness the attacker could also reverse engineer firmware in order to then proceed with a remote code execution compromising integrity.

Zero LTK Installation (CVE-2019-19194): The link layer Zero LTK Installation attack occurs when the peripheral negotiates a zero pairing key (the key length equals to zero instead of 16 bytes), in this way even session key is compromised since it is derived from a valid LTK (which in this scenario is equal to zero). In this scenario, the attacker manages to circumvent the security of the BLE device and could access protected data because the keys distribution procedure is completely bypassed.

DHCheck Skip (CVE-2020-13593): it consists in partially bypassing security during Phase 2 of the pairing in the link layer of some SoC (system-on-a-chip) vendors. The DHCheck can be skipped by starting the LL Encryption Procedure earlier and lead to installation of the LTK generated in the unfinished pairing process. With this weakness, like to Zero LTK installation, the keys distribution procedure is bypassed. DHCheck has similar requirements to Zero LTK installation, without using a zeroed LTK, but with the same result. Due to the weakness of the key, these last two attacks compromise confidentiality of the data.

Mitigation:As mentioned in [20], some vulnerabilities mentioned above, are mitigated through the release of newer updates by the manufacturer. However, some of these vulnerabilities are still present pending future patches.

Denial of Service (DoS) Attack: a denial-of-service is an attack that makes a network or generally an IT resource unavailable to users by disrupting services. The attacker sends excessive messages to a server without return address for authentication approval. In this way the server waits for the connection to start, keeping the network busy. The very same concept in BLE is easy to deploy and difficult to mitigate.

In [2], the aforementioned concept is extended by describing a specific DoS attack known as denial-of-sleep (DoSL) attack or Battery-Draining-Denial-of-Service Attack which combines two major issues for an IoT device: **Energy drain and Denial of Service**. In particular, the attack paralyzes the device by consuming the battery by making repeated attempts of connections and disconnections to the target. Dos and battery exhaustion attacks, are the main methods to deny or degrade the availability of BLE services. Using this type of attack on a smartphone, for example, can invalidate the device from receiving phone calls and reducing its battery life by as much as 97% [37].

Mitigation: The first countermeasure available for DoS attacks is the introduction of the multi-master model in BLE Version 4.1. This model enables the possibility for a device to have multiple masters at the same time. In this way, even if an attacker increase the advertisement rate to connect to an available slave, the slave would always continue to send advertisements messages to look for more other possible masters, preventing it to be invisible to the rest of the network.

Battery exhaustion: one of the most popular attack related to availability for BLE devices, is the Battery exhaustion [38], in which the attacker intentionally consumes device battery lifespan to disable the device and the network as a whole. This attack affects one of the fundamental characteristics of the BLE protocol, i.e. the battery life of the devices and it can lead to major disruption since there is not always the possibility of recharging or replacing the battery especially in sensor networks deployed in critical areas (e.g. in Bridge Health monitoring applications or in Medical BLE applications). The authors in [29] have conducted a lot of experiments that shows how smart wristbands under this kind of attacks are not only deprived from their battery power, but also from the availability of their main services due to frequent phases of disconnection and reconnection from their legitimate users.

Mitigation: As stated by Zang et al. in [29], all of the wristbands analyzed exchange unencrypted data in the communication with the smartphone. The implementation of message encryption at the application level would help the wristbands to understand which messages are legit and which ones to discard, helping to reduce the notifications sent to the user, their relative access to the vibration sensors and to the display and thus their battery consumption.

The authors in [39] describe how any forms of encryption can be evaded by a technique called packet injection which allows an unknown third party to disrupt or intercept packets from the consenting parties that are communicating. In this case packet injection is used to demonstrate how Long Term Key, exchanged during the pairing phase, is derivable by a simple brute force. However, the packet injection method has limitations.

The work in [40] introduces a novel attack called BLE injection-free attack that represents the evolution of packet injection, removing some limitations. The latter exploits a vulnerability caused by the

size of a device's bonding list which allows to successfully force LTK renegotiation for a connection despite unsuccessful attempts via neither packet injection nor connection jamming. Moreover, the parameters used for key generation during pairing can be easily differentiated with a simple communication sniffing. One of these parameters is the PIN (Personal Identification Number).

The process by which the PIN is derived and through that, originate the link key which encrypts the channel is described by authors in [41]. To perform a successful sniffing attack, an attacker needs specialized tools, like the aforementioned Ubertooth, Crackle[5] and similar, and Ad-Hoc platforms.

The work in [42] for instance, interception of run-times communicated packets is done via Ubertooth which is a cheap, open-source Bluetooth network sniffer. When a device transmits, thanks to this sniffer we can intercept the communication between the devices. After capturing the communication packets, they must be decrypted before they can be accessed. Decryption becomes trivial if a complete association event is present in sniffed packages. However Crackle allows cracking even if the flow of data is incomplete. This can also be achieved with BLE Long-Lived Real-time connections [43]. This tool can then return a .pcap file of the data acquired with the clear text and the respective Temporary Key and Long Term Key.

In general, BLE devices are still vulnerable to the packet injection attack, however the limitations imposed by the BLE protocol such as the very short transmission distance, make the attack very difficult to perform. Theoretically, this type of attack allows to be able to subsequently execute a Man-in-the-Middle (MitM) attack within a network, compromising its security.

Mitigation: the work in [40] describes some possible mitigation to counter the injection-free attack based on a better implementation of how the bonding list and bonding requests are handled by the developers.

Man in the Middle (MitM) Attacks: this type of attack indicates usually a passive or active intrusion of a malicious attacker in a communication between two devices. If the devices in question are part of a BLE network, the entire network can be considered compromised. In its classic operation on other kind of wireless networks, when the client sends a message to the server, the attacker impersonates the latter and, conversely, when the server sends a message, the attacker impersonates the client.

This precise behavior cannot happen in a BLE connection since one single device cannot be connected simultaneously with two different groups of devices.

The work in [44], widely demonstrates how a MitM attack is performed through the use of two different devices by the attacker communicating each other over a different mean, like, for instance, Websockets. One of the two attacking devices will connect to the target server acting as client, while the other will connect to the victim client acting as server. The data transferred between the two malicious devices are those that would have been transferred legitimately between the two victim devices. Such data, can also be modified by the attacker before being exchanged. This whole process exploits the packets of GATT to access to data of the victim affecting data confidentiality.

Mitigation: The most of the time the MiTM attacks are effectively mitigated by the presence within the network of nodes capable of analyzing traffic data such as Intrusion Detection Systems. We will discuss in depth about them in Section 5.

KNOB attack on BLE: through KNOB attack, also known as Key Negotiation Downgrade attack, is possible to force two BLE devices to renegotiate in the link layer long-term key and session key with the lowest entropy specified by the standard. Since the BLE specification does not require the integrity protection of the feature exchange phase which includes the key negotiation, the entropy of the BLE encryption key can be reduced to 7 bytes, the minimum acquired by the standard, regardless of the use of authenticated secure connections. The reduction of entropy and the consequent choice of weak keys, will allow a hypothetical attacker to perform brute force on the same keys.

This results in decrypting the communication between two endpoints, jamming the channel, and crafting valid decrypted Bluetooth packets. The threats generated by the KNOB attack can be disruptive in an environment where BLE is now an integral part of the functions that make up many IoT applications.

In [45], is pointed out that in Android devices, thanks to downgrading, it is possible to successfully conduct an MitM attack. In addition, the same downgrading attacks are also successfully tested on the other major commercial Operating systems which appear to not properly support the Secure Connection mode.

Mitigation: various solutions have been proposed by [28] to mitigate the KNOB attack. Some of these can be applied by individual vendors, without changing the standard in any way (Legacy compliant).

Other solutions instead concern the possible solutions that Bluetooth can directly apply to its standard (Non-legacy Compliant). A Legacy Compliant countermeasure can be to set a minimum value that the entropy of the LTK must assume in the feature exchange phase implementation in the BLE host, for instance 16 bytes minimum.

A Non-legacy compliant countermeasure instead, can be of permanently remove entropy negotiation from BLE pairing. In fact, the LTK should always use a minimum entropy of 16 bytes and the KeySize parameter should be eliminated as well.

An alternative may be to establish and authenticate a new key that will serve to protect the integrity of the feature exchange phase, including KeySize. Both solutions require changing the standard Bluetooth and BLE host.

4.2. Attacks on broadcast networks

Selective jamming: according to the authors in [31], a jammer can be selective if programmed to attack only specific frames. Since BLE advertising beacons are sent redundantly over different channels, we need to further distinguish between narrowband and broadband jamming.

- A constant wideband jammer emits noise indefinitely over a wide range of frequencies;
- A constant narrowband jammer emits the jamming signal permanently but only on a single BLE advertising channel;
- A reactive wideband jammer relies on the observation that BLE beacons are only sent at certain times. Therefore this emits the jamming signal only during the transmission of the victim frame;
- A reactive narrowband jammer outputs the jamming signal over a single BLE ad channel only when a frame transmission to attack has been detected.

Mitigation: a wideband jammer can be easily detected by an IDS with a spectrum analyzer. Instead against a narrowband jammer, since only a single BLE channel can be disturbed, it is possible to apply frequency hopping in order not to lose the frame beacons transmitted on other channels.

Spoofing attacks: in a conventional spoofing attack, the opponent first collects information about the advertisement packets since they are in clear text, then, after the attacker has cloned the benign server Bluetooth Device Address (BDA) to impersonate it, he starts to forward those packets in broadcast. In this way, the opponent makes sure that when other devices try to reconnect to the benign server, they will receive the counterfeit packets and connect to the opponent server instead. The attacker is able to perform above process, thanks to the exploitation of the pairing process in link layer and consequently damaging confidentiality.

As already mentioned before, [32] explains the modalities and conditions that allow an attacker to perform a successful spoofing

[5] https://github.com/mikeryan/crackle.

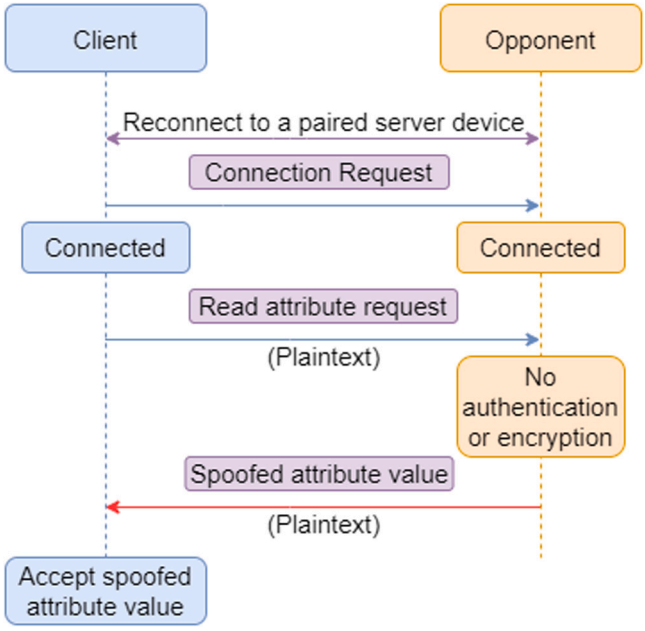

Fig. 7. BLE Spoofing Attack for reactive Authentication.

attack. Spoofing takes place during the connection phase and in this regard, the **Reactive** and **Proactive** authentication methods described by the BLE specification are used to perform this task. In the first case, the weakness lies in the fact that the client relies on a server error message to possibly increase the level of security. First the client sends an attribute read request to the server at the lowest security level allowed, i.e. clear text communication. The server, realizing that the security level does not correspond to what the client should assume for the attribute, i.e. level 3, encryption and authentication, sends back an error message forcing the client to increase this level.

An attacker who impersonates the benign server, instead of sending back an error message, sends a spoofed attribute value expected by the client to enable the minimum required security level. In this way the client, not encountering an error message, assume that the attribute can be accessed at the lowest security level by accepting the spoofed attribute. Fig. 7 shows the completion of the attack from the sending of the connection request.

Regarding proactive authentication, it is the client that proactively enables encryption and consequently also authentication. According to the BLE specification, if encryption enabling process fails, then the client should either re-pair or abort the connection. However, some Android devices and iOS-based, even if the encryption enabling process fails, they continue the connection in plain text allowing an attacker to spoof.

A spoofing attack can have disruptive effects in several scenarios. During an experiment described in [46] it was shown that two different spoof attacks can compromise a building occupancy detection system: Evil Twin attack and Beacon Swap attack.

In the first scenario, an attacker clones and impersonate the victim's beacon. In the second, the attacker has no physical access and remotely hijacks the beacon and modifies its configuration (for example, swaps the identities of two beacons). To demonstrate this, the experiment was conducted in a laboratory dividing it into 7 sectors. Each sector had one or more BLE beacons that continuously sent out broadcast data packets, according to the IBeacon protocol, that are received via a locator APP from a tag device which in turn will send these packets to a remote control server. The latter, through collected beacon data and an internal trained classifier, will establish the user's location in the building. After carrying out the two attacks mentioned above, there is a significant deterioration in the classification results of their Logistic Regression classifier which means for instance, the disruption of an indoor localization system or in a terrorist attack scenario, the opponent can implement a beacon exchange to mislead rescuers and stop the rescue operation in the building.

In Figs. 8 and 9, a conceptual scheme is shown that represents the scenario in the case of the normal functioning of the localization system and in the case in which an Evil Twin attack is carried out thus compromising the integrity of the entire network. As shown in the figures, in the former case, the server correctly predicts that the device is in Sector 2 thanks to the RSSI generated by the tag, while if the Evil twin is executed, the server could be confused by the RSSI generated from the cloned packets of beacon B3, assuming that the device is in Sector 3.

Mitigation: As described in [32], to reduce the chances of this attack being successful, the BLE stack implementations need to be improved to further secure the re-connection mechanism between previously paired devices. This could be done by mechanisms that have been introduced in the Core Specification 5.2, namely Security Mode 3 (Section 3), currently used for securing broadcast communications.

4.3. Attacks on mesh networks

While some of the attacks previously discussed can be used as injection point for mesh networks, especially the ones that allow full control of a node, in this part we focus our attention on the attacks that only target the Mesh profile. Despite the security mechanism discussed in Section 3, Bluetooth Low Energy Mesh networks are still vulnerable to attacks. Even if there are a wide number of different keys, most of them are based on the device key and on the application key, which can both be recovered by hardware exploitation. These exploits usually happens by physically stealing an active device or by recovering a device after it has reached end-of-life if its memory is not properly erased and disposed of. A stolen Device Key is very dangerous since it allows the full control of the victim node, making it a *malicious* node capable to target different nodes in the network that are not physically reachable by the attacker.

The hardware attacks identified by the authors in [9] are:

- **Simple Power Analysis—SPA**: this technique involves observing the acquisitions of energy consumption over time. In particular, the ECDH key agreement algorithm is known to be vulnerable to SPA attacks if not implemented correctly. By doing so, the pairing process could be compromised as it could lead to the restoration of the DevKey, giving access to all the security credentials sent during the device configuration.
- **Differential Power Analysis**: this type of attack is based on divide and conquer strategy, in which the aim is to derive small parts of the AppKey or NetKey independently. In the case of a Mesh network, the attacker could carry out the attack by sniffing traffic or injecting network PDUs.
  Mitigation: the countermeasure against power analysis attacks is called "*masking*". It consists in making energy consumption independent of the processed data.
- **Fault Attack**: this type of attack is usually applied to cryptographic algorithms to induce an error during one of the calculations and thus have a pair of output values, one correct and one incorrect. By analyzing this pair of values, the attacker can retrieve information on the key used. However, to carry out this attack, the attacker still need to check the input. To make this attack applicable, there exists a technique called statistical fault attack, which only requires knowledge of the wrong values encrypted with the same key. At this point, the attacker could infer the key used for the specific encryption algorithm.

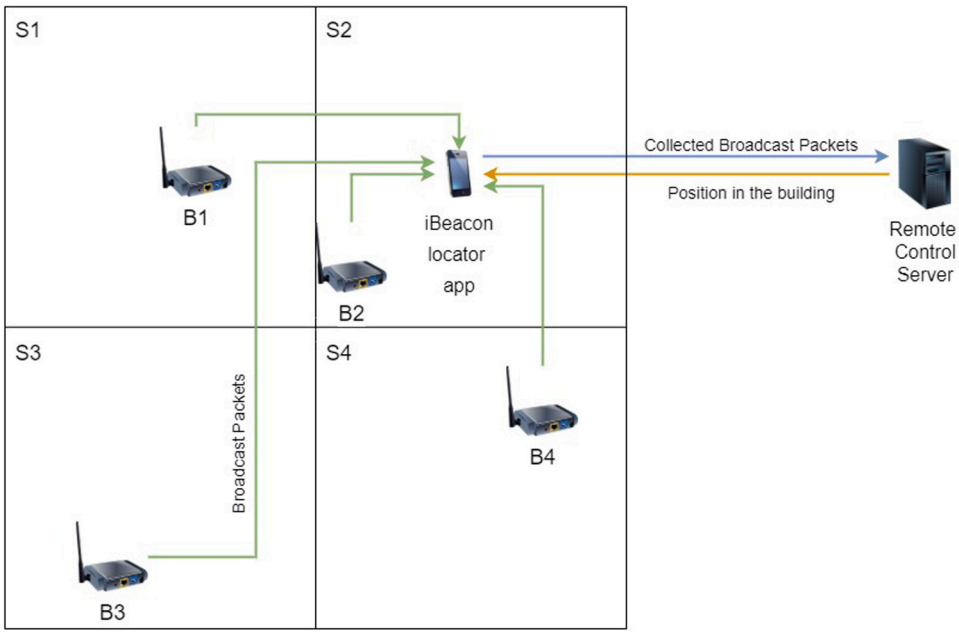

Fig. 8. Normal functioning of the localization system.

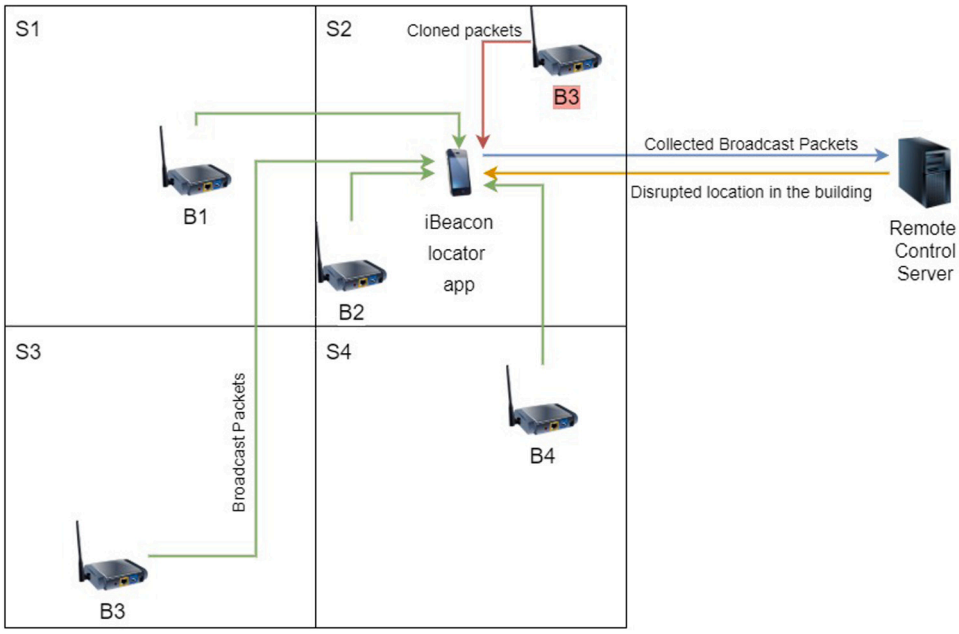

Fig. 9. Disruption of the localization system by carrying out an Evil Twin attack.

The attacks can be divided on the basis of their purpose: denial of service attacks (DoS), information acquisition attacks and attacks for reducing the Quality of Service (QoS).

The denial of service attacks foresee the alteration of the legitimate network operations. This kind of attacks can be carried on at different levels of the BLE Mesh stack from the physical destruction of the device, to the electromagnetic occupation of the whole shared medium as already discussed in the previous sections.

Going up the BLE Mesh stack, we have different kinds of attacks that can be either conducted to a denial of service or to a partial decrease of the quality.

One of the most severe attack affecting the BLE Mesh network is the *selective forwarding attack* [47], where a malicious relying node does not forward selected messages, so the message is not propagated further. If the node drops only part of messages, the attack is called a *Gray hole attack*. In the extreme case, when the malicious node does not rely any message, the attack is called a *Black hole attack*. Due to flooding approach for implementing many-to-many communication in BLE mesh networks, the effectiveness of the attack depends on the number of infected nodes, the density of the network and on the roles the node have with the network, especially if the number of Low Power and Friends nodes is significant.

Another way to remove packets from the network before they arrive to the destination node is by malforming the TTL value of packets [48]. By exploiting the algorithm for calculating the initial time to live TTL value, instead of using the default value defined by the standard, which is 7. This brings in some cases to values that are not be adequate to reach all the nodes in large networks. In the algorithm, the initial

TTL value for sending the given message to the destination node with the given unicast address is based on Heartbeat message sent by that node. Thus, if payload of the Heartbeat message created by destination node is malformed, then TTL value required to deliver message to the destination node may be underestimated and rejected by relying nodes before arriving to the target.

Another major threat for the Mesh networks is represented by the possible exploitation of the design vulnerabilities of the "friendship" concept [49] described in Section 3. These attacks are extremely dangerous because their principal target are the Low Power nodes and the disruption of the communication between them and their Friends. The attacks related to the "friendship" feature identified so far are:

- **Collection of the friendship security materials**: because of the fact that the credentials are always derived from the NetKey, which is owned by all the nodes within the same network, an intruder can reconstruct the friendship security material of two nodes by using its NetKey to decipher messages exchanged in the network. Once an LPN has been identified, the attacker can take the *LPNAddress* and *LpnCounter* fields from the Friend Request messages that the node sends. Furthermore, by filtering the Friend Offer messages addressed to that node, the attacker can also obtain the *FriendAddress* and *FriendCounter* fields, enabling the possibility of directly calculate the Friendship Security Material.
- **Exploit *Poll* message policies**: despite being the most used message in the friendship, its size is fixed to just one byte at the transport level. An adversary could easily monitor the outgoing traffic from the LP node he intends to attack, and decide to eavesdrop on the communication and selectively jam those messages purely based on observing the length field of the packet header. In fact, the complete size of a Poll message PDU at the Network Layer of only 19 bytes. If an LPN node cannot successfully send its Poll messages, it is impossible for it to participate in the network without prohibitive energy consumption.
- **Improper use of the *Clear* message**: this is considered as one of the major drawbacks of the friendship design. According to the standard [11] each node in the network can send a *Clear* message with the master credentials owned by all members of the network. The rationale of this design lies in the fact that it is not always the task of an LP node to send a *Clear* message, but in case of the establishment of a new friendship with a node, the new Friend must inform the old one that the pairing has ended. However, in this way it can be easy for an intruder to terminate the friendship since the only element required to successfully send such a message are the address of the LPN you want to attack and its counter. Moreover, there is no actual strict requirement the counter to be valid except that it needs to be greater than the previous ones, making sufficient to assign a very high value to it. To perform this technique, the attacker has a lot of opened possibilities, such as the decryption of the packets coming from the LP node, the discovery the address of a node even by brute force since there are not so many possible MAC addresses of an LPN node. Obviously, if this type of attack is successful it would involve both a considerable energy consumption, since the attacked LP node would have to restart its pairing routine every few minutes, and also a considerable data loss at the application layer. If a friend node receives a valid *Clear*, it will instantly free the entire message queue for that node.

At the time of this writing there are is no mitigation available for this class of attacks in the literature. This is due mostly to the fact that these vulnerabilities are intrinsic to the standard design and they cannot be handled without changing it.

5. Attack prevention methodologies

As described in Section 2, Bluetooth mesh networking is based on a client–server architecture and allows the generation of very large-scale networks made up of thousands of devices. These devices, named also nodes, implement their own functionalities (models) that change the condition of the device itselef (states). The server makes its state available to allow clients to interact with it via a state request. The interaction changes the state of the server as for example in the simplest case of a binary switch where the state is either on or off. To cope with threats in mesh networks as presented in Section 4, Intrusion Detection Systems (IDS) can be deployed to mitigate and prevent major damages to the traffic flow. For BLE Mesh networks there are very few implementations of IDSs.

In [50] a detection and prevention system against Battery Exhaustion Attack is presented. The approach used is to implement a defense for three different levels of attack. In this case, the level 1 attack is when the attacker sends connection/disconnection requests repeatedly before establishing the connection. Level 2 attack, on the other hand, is when the attacker sends requests to use services that are not available in the network. Finally, the level 3 attack occurs when the malicious device is already connected to the mesh network, making repeated requests to services. This attack is more difficult to detect since valid requests must be distinguished from malicious ones. The IDS, uses as a defense for Level 1 attacks a timer and counter to calculate the frequency of connections/disconnections made by a device. If this frequency exceeds a certain threshold of connections/disconnections in a given time period, then the device is considered malicious. The same approach is used for a defense against Level 2 attacks, but instead of connection/disconnection requests, requests for services are considered. Defending against Level 3 attacks requires a more elaborate approach. In fact, a priority level and a counter are associated to each node. With each request made by the node, its priority level decreases while at the same time, the counter increases. When the counter reaches a certain threshold, the device is forced to change its connection to the mesh network through a new provisioner.

In [48] the authors discuss the optimization of the placement of a set of watchdog nodes responsible for traffic analysis inside a BLE Mesh network. Following the anomaly detection approach, they collect data periodically for a given time period in order to model the default behavior of the network in the most accurate way possible. Every interval is called *time window*. For each time window a set of parameters is computed, like for instance the average and the standard deviation of the number of packets per source address, the average and the standard deviation of the number packets per destination address, the average and the standard deviation of the TTL and RSSI values and some others. These time windows are classified and clusterized with a finite Gaussian mixture modeling (GMM), allowing the self-computation of the number of clusters (mixture components) without the need of choosing it a priori.

During the detection phase, the watchdog records newer time windows with a fixed periodicity that must have the same length of the ones used in the training phase and extracts their description parameters.

In this way, the watchdog can compute the euclidean distance between the GMM model and each fresh time window that represents how much the single set of packet are differing from the standard behavior defined in the training phase. If such distance exceeds a given threshold it means that some anomaly is currently going on in the network, that can be either identified as a generic malfunctioning or as a Denial of Service attack. In this case, the watchdogs will raise the alarm giving the control to an human operator. To detect the greatest number of anomalies, the watchdogs have to be physically put in optimum placements in order to maximize their capability.

While this work is carried out mostly on simulation, the authors provide a real implementation and prove their results over Nordic

Semiconductor boards. Their experimental setup, however, places all the nodes in very tight spaces, altering the generality of values such as the RSSI, which is based on the distance between two nodes exchanging messages. and the time to live (TTL) attribute of each packet.

In [51], a machine learning IDS is presented based on pattern classification and recognition of DoS attacks such as Gray Hole attack and Black hole attack against a server in the network. This IDS bases its misuse analysis on machine learning aiming at understanding if within the network there are any attackers, by creating a model able to perform a multi-class pattern classification of different traffic flows happening in a legitimate situation and when attacks are ongoing. The research focused mostly on the Black Hole and the Grey Hole attacks since they are the most simpler to implement yet the most disruptive.

One of the main contributions of the author's work is the application of a data collection system based on ESP32, which partially overcome the issue of the absence in the literature of a training data set for BLE mesh networks.

By analyzing the results, this IDS proved that there are actually strong correlations between the time windows. This allows the detection of the ongoing anomalies with a strong accuracy, but its performance strongly degrades when it comes the time to understand what exactly the aforementioned anomalies are. Therefore, while the IDS has proved to have excellent potentialities and while this approach can be successful even in large implementations, the experiments done so far have been insufficient to give a more precise accuracy range.

A novel work presented by [26] shows a machine learning based method to detect any MitM attacks for BLE devices. The approach used by the authors was to detect suspicious activity using reconstruction and classification techniques, applying a Temporal Convolutional Network (TCN) and a combination of Convolutional Neural Network (CNN) and Random Forest. With their classification model, the authors obtained results with high performance of separability between "normal" and "attack" packets with 99% accuracy and 0.3% false positives. This work can be a good example of countering the MiTM attacks exposed by [44] in Section 4.

In the paper [52] it is also presented a solution for detecting, analyzing and mitigating MITM attack especially against No-input–No-output devices. Such devices have really strict requirements in terms of battery consumption because they usually serve critical applications, like for instance a smart pacemaker in the e-health sector, where Bluetooth is one of the most used protocols thanks to its easy accessibility and compatibility with the most of the hardware. After the simulation of session hijacking and data manipulation on real hardware as we presented in Section 4, the authors detect these Man in the Middle attacks in BLE NiNo devices and propose a mitigation framework named MARC that performs an anomaly detection aimed to evaluate:

- **M**alicious scan requests;
- **A**dvertisement intervals;
- Anomalies of **R**SSI levels;
- Presence of **C**loned node addresses;

Finally, they propose as main points to be considered in the prevention of threats the address white listing, the analysis of advertisement frequency and the threshold on RSSI level.

6. Conclusions

This work surveys security features and issues of Bluetooth Low Energy (BLE) and of its Mesh Profile, covering attacks against the different layers of their protocol stacks and indicating possible mitigation approaches. We provide an introductory description of BLE and of its mesh networking profile and we review their security features. We then describe attacks that can affect the pairing process and BLE connections. We highlight that BLE has been subjected to several attacks over the years and that many of them still threaten most versions. Our work also provides a taxonomy of the attacks on BLE networking and their countermeasures. Emphasis is given to Bluetooth Mesh networking and its security features, as it is anticipated that this new communication mode will be key to enable most application scenarios (e.g., mesh networks of devices, location services, contact tracing, etc.). We show that in this case the use of Intrusion Detection Systems is highly recommended. Our review concludes with a description of attack prevention methodologies to counter the listed attacks.

CRediT authorship contribution statement

Andrea Lacava: Writing – original draft, Investigation, Resources. **Valerio Zottola:** Writing – original draft, Investigation, Resources. **Alessio Bonaldo:** Investigation, Resources. **Francesca Cuomo:** Writing – review & editing, Supervision. **Stefano Basagni:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Bluetooth SIG Alliance, Bluetooth Technology Overview, https://www.bluetooth.com/learn-about-bluetooth/tech-overview/.

[2] Y. Zhang, J. Weng, R. Dey, X. Fu, Bluetooth low energy (BLE) security and privacy, in: X.S. Shen, X. Lin, K. Zhang (Eds.), Encyclopedia of Wireless Networks, Springer International Publishing, 2019, pp. 1–12, http://dx.doi.org/10.1007/978-3-319-32903-1_298-1.

[3] Bluetooth SIG Alliance, Bluetooth core specification v.5.1, 2019, [Online; accessed May 11, 2022]. https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=457080.

[4] Bluetooth SIG Alliance, Bluetooth core specification v.5.2, 2019, [Online; accessed May 11, 2022], https://www.bluetooth.com/specifications/specs/core-specification-5-2/.

[5] A.A. Pammi, Threats, Countermeasures, and Research Trends for BLE-Based IoT Devices (Ph.D. thesis), Arizona State University, 2017.

[6] B.R. Moyers, J.P. Dunning, R.C. Marchany, J.G. Tront, Effects of Wi-Fi and bluetooth battery exhaustion attacks on mobile devices, in: 2010 43rd Hawaii International Conference on System Sciences, 2010, pp. 1–9, http://dx.doi.org/10.1109/HICSS.2010.170.

[7] M. Zubair, D. Unal, A. Al-Ali, A. Shikfa, Exploiting bluetooth vulnerabilities in E-health IoT devices, in: Proceedings of the 3rd International Conference on Future Networks and Distributed Systems, in: ICFNDS '19, Association for Computing Machinery, New York, NY, USA, 2019, http://dx.doi.org/10.1145/3341325.3342000.

[8] H.J. Tay, J. Tan, P. Narasimhan, A Survey of Security Vulnerabilities in Bluetooth Low Energy Beacons, Parallel Data Lab Technical Report CMU-PDL-16-109, Carnegie Mellon University, 2016.

[9] A. Adomnicai, J.J.A. Fournier, L. Masson, Hardware security threats against bluetooth mesh networks, in: 2018 IEEE Conference on Communications and Network Security, CNS, 2018, pp. 1–9, http://dx.doi.org/10.1109/CNS.2018.8433184.

[10] M.R. Ghori, T.-C. Wan, M. Anbar, G.C. Sodhy, A. Rizwan, Review on security in bluetooth low energy mesh network in correlation with wireless mesh network security, in: 2019 IEEE Student Conference on Research and Development, SCOReD, IEEE, 2019, pp. 219–224.

[11] Bluetooth SIG Alliance, Mesh profile specification v.1.0.1, 2019, [Online; accessed May 11, 2022], https://www.bluetooth.com/specifications/specs/mesh-profile-1-0-1/.

[12] Nordic Semiconductor, Bluetooth mesh concepts, 2021, [Online; accessed May 11, 2022], https://developer.nordicsemi.com/nRF_Connect_SDK/doc/latest/nrf/ug_bt_mesh_concepts.html.

[13] Bluetooth core specification version 5.2 feature overview, 2020, URL https://www.bluetooth.com/bluetooth-resources/bluetooth-core-specification-version-5-2-feature-overview/.

[14] D. Hankerson, A. Menezes, NSA suite B, in: Encyclopedia of Cryptography and Security, Springer US, Boston, MA, 2011, p. 857, http://dx.doi.org/10.1007/978-1-4419-5906-5_648.

[15] S.M. Darroudi, R. Caldera-Sànchez, C. Gomez, Bluetooth mesh energy consumption: A model, Sensors 19 (5) (2019) http://dx.doi.org/10.3390/s19051238, URL https://www.mdpi.com/1424-8220/19/5/1238.

[16] R. Rondón, A. Mahmood, S. Grimaldi, M. Gidlund, Understanding the performance of bluetooth mesh: reliability, delay, and scalability analysis, IEEE Internet Things J. 7 (3) (2019) 2089–2101.

[17] J. Yin, Z. Yang, H. Cao, T. Liu, Z. Zhou, C. Wu, A survey on bluetooth 5.0 and mesh: New milestones of IoT, ACM Trans. Sen. Netw. 15 (3) (2019) http://dx.doi.org/10.1145/3317687.

[18] M. Tan, K.A. Masagca, An investigation of bluetooth security threats, in: 2011 International Conference on Information Science and Applications, 2011, pp. 1–7, http://dx.doi.org/10.1109/ICISA.2011.5772388.

[19] P. Cope, J. Campbell, T. Hayajneh, An investigation of bluetooth security vulnerabilities, in: 2017 IEEE 7th Annual Computing and Communication Workshop and Conference, CCWC, 2017, pp. 1–7, http://dx.doi.org/10.1109/CCWC.2017.7868416.

[20] M.E. Garbelini, C. Wang, S. Chattopadhyay, S. Sumei, E. Kurniawan, Sweyntooth: Unleashing mayhem over bluetooth low energy, in: 2020 USENIX Annual Technical Conference, USENIX ATC 20, USENIX Association, 2020, pp. 911–925, URL https://www.usenix.org/conference/atc20/presentation/garbelini.

[21] M.R. Ghori, T.-C. Wan, G.C. Sodhy, Bluetooth low energy mesh networks: Survey of communication and security protocols, Sensors 20 (12) (2020) 3590.

[22] J. Padgette, K. Scarfone, L. Chen, Guide to bluetooth security, NIST Special Publ. 800 (121) (2012) 25.

[23] A. Ray, V. Raj, M. Oriol, A. Monot, S. Obermeier, Bluetooth low energy devices security testing framework, in: 2018 IEEE 11th International Conference on Software Testing, Verification and Validation, ICST, 2018, pp. 384–393, http://dx.doi.org/10.1109/ICST.2018.00045.

[24] R. Krejčí, O. Hujňák, M. Švepeš, Security survey of the IoT wireless protocols, in: 2017 25th Telecommunication Forum, TELFOR, 2017, pp. 1–4, http://dx.doi.org/10.1109/TELFOR.2017.8249286.

[25] M. Ammar, G. Russello, B. Crispo, Internet of things: A survey on the security of IoT frameworks, J. Inf. Secur. Appl. 38 (2018) 8–27, http://dx.doi.org/10.1016/j.jisa.2017.11.002, URL http://www.sciencedirect.com/science/article/pii/S2214212617302934.

[26] A. Lahmadi, A. Duque, N. Heraief, J. Francq, Mitm attack detection in BLE networks using reconstruction and classification machine learning techniques, in: MLCS 2020-2nd Workshop on Machine Learning for Cybersecurity, 2020.

[27] J.B. Pallavi Sivakumaran, 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: Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy, 2018.

[28] K.R. Daniele Antonioli, Low entropy key negotiation attacks on bluetooth and bluetooth low energy, in: ACM Transactions on Privacy and Security, 2020.

[29] Q. Zhang, Z. Liang, Security analysis of bluetooth low energy based smart wristbands, in: 2017 2nd International Conference on Frontiers of Sensors Technologies, ICFST, 2017, pp. 421–425, http://dx.doi.org/10.1109/ICFST.2017.8210548.

[30] W. Xu, W. Trappe, Y. Zhang, T. Wood, The feasibility of launching and detecting jamming attacks in wireless networks, in: Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing, in: MobiHoc '05, Association for Computing Machinery, New York, NY, USA, 2005, pp. 46–57, http://dx.doi.org/10.1145/1062689.1062697.

[31] S. Bräuer, A. Zubow, S. Zehl, M. Roshandel, S. Mashhadi-Sohi, On practical selective jamming of bluetooth low energy advertising, in: 2016 IEEE Conference on Standards for Communications and Networking, CSCN, 2016, pp. 1–6, http://dx.doi.org/10.1109/CSCN.2016.7785169.

[32] J. Wu, Y. Nan, V. Kumar, D.J. Tian, A. Bianchi, M. Payer, D. Xu, BLESA: Spoofing attacks against reconnections in bluetooth low energy, in: Proceedings of WOOT 2020 @ Usenix Security Symposium, virtual workshop, 2020, pp. 1–12.

[33] A.K. Goel, A. Rose, J. Gaur, B. Bhushan, Attacks, countermeasures and security paradigms in IoT, in: 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies, Vol. 1, ICICICT, 2019, pp. 875–880, http://dx.doi.org/10.1109/ICICICT46008.2019.8993338.

[34] M. Warkentin, C. Orgeron, Using the security triad to assess blockchain technology in public sector applications, Int. J. Inf. Manage. 52 (2020) 102090.

[35] O.E. Mouaatamid, M. Lahmer, M. Belkasmi, Internet of things security: Layered classification of attacks and possible countermeasures, Electr. J. Inf. Technol. (9) (2016) URL http://www.webmail.revue-eti.net/index.php/eti/article/view/98.

[36] D. Cauquil, You'd BETTER secure YOUR BLE DEVICES OR we'll KICK your BUTTS !, 2018, [Online; accessed May 11, 2022], https://media.defcon.org/DEFCON26/DEFCON26presentations/DEFCON-26-Damien-Cauquil-Secure-Your-BLE-Devices-Updated.pdf.

[37] H. O'Sullivan, Security Vulnerabilities of Bluetooth Low Energy Technology (Ble), Tufts University, 2015.

[38] K.V. Ritesh, A. Manolova, M. Nenova, Abridgment of bluetooth low energy (BLE) standard and its numerous susceptibilities for internet of things and its applications, in: 2017 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems, COMCAS, 2017, pp. 1–5, http://dx.doi.org/10.1109/COMCAS.2017.8244814.

[39] M. Ryan, Bluetooth: With low energy comes low security, in: 7th USENIX Workshop on Offensive Technologies, WOOT 13, USENIX Association, Washington, D.C., 2013, URL https://www.usenix.org/conference/woot13/workshop-program/presentation/ryan.

[40] A.C.T. Santos, J.L.S. Filho, Á.Í.S. Silva, V. Nigam, I.E. Fonseca, BLE injection-free attack: a novel attack on bluetooth low energy devices, J. Ambient Intell. Humaniz. Comput. (2019) http://dx.doi.org/10.1007/s12652-019-01502-z.

[41] Y. Shaked, A. Wool, Cracking the bluetooth PIN, in: Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services, in: MobiSys '05, Association for Computing Machinery, New York, NY, USA, 2005, pp. 39–50, http://dx.doi.org/10.1145/1067170.1067176.

[42] S. Sevier, A. Tekeoglu, Analyzing the security of bluetooth low energy, in: Proceedings of ICEIC 2019, Auckland, New Zealand, 2019, pp. 1–5.

[43] S. Sarkar, J. Liu, E. Jovanov, A robust algorithm for sniffing BLE long-lived connections in real-time, in: Proceedings of GLOBECOM 2019, Waikoloa, HI, 2019, pp. 1–6.

[44] T. Melamed, An active man-in-the-middle attack on bluetooth smart devices, Saf. Secur. Stud. 15 (2018) 2018.

[45] Y. Zhang, J. Weng, R. Dey, Y. Jin, Z. Lin, X. Fu, Breaking secure pairing of bluetooth low energy using downgrade attacks, in: 29th USENIX Security Symposium, USENIX Security 20, USENIX Association, 2020, pp. 37–54, URL https://www.usenix.org/conference/usenixsecurity20/presentation/zhang-yue.

[46] W. Oliff, A. Filippoupolitis, G. Loukas, Evaluating the impact of malicious spoofing attacks on bluetooth low energy based occupancy detection systems, in: 2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications, SERA, 2017, pp. 379–385, http://dx.doi.org/10.1109/SERA.2017.7965755.

[47] L.K. Bysani, A.K. Turuk, A survey on selective forwarding attack in wireless sensor networks, in: 2011 International Conference on Devices and Communications, ICDeCom, 2011, pp. 1–5, http://dx.doi.org/10.1109/ICDECOM.2011.5738547.

[48] M. Krzysztoń, M. Marks, Simulation of watchdog placement for cooperative anomaly detection in bluetooth mesh intrusion detection system, Simul. Model. Pract. Theory 101 (2020) 102041, http://dx.doi.org/10.1016/j.simpat.2019.102041, Modeling and Simulation of Fog Computing URL http://www.sciencedirect.com/science/article/pii/S1569190X19301728.

[49] F. Álvarez, L. Almon, A.-S. Hahn, M. Hollick, Toxic friends in your network: Breaking the bluetooth mesh friendship concept, in: Proceedings of the 5th ACM Workshop on Security Standardisation Research Workshop, in: SSR'19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 1–12, http://dx.doi.org/10.1145/3338500.3360334.

[50] Z. Guo, I.G. Harris, Y. Jiang, L.-f. Tsaur, An efficient approach to prevent battery exhaustion attack on BLE-based mesh networks, in: 2017 International Conference on Computing, Networking and Communications, ICNC, 2017, pp. 1–5, http://dx.doi.org/10.1109/ICCNC.2017.7876092.

[51] A. Lacava, E. Giacomini, F. D'Alterio, F. Cuomo, Intrusion detection system for bluetooth mesh networks: data gathering and experimental evaluations, in: SPT-IoT 2021: The Fifth Workshop on Security, Privacy and Trust in the Internet of Things, SPT-IoT 2021, Kassel, Germany, 2021.

[52] M. Yaseen, W. Iqbal, I. Rashid, H. Abbas, M. Mohsin, K. Saleem, Y.A. Bangash, MARC: A novel framework for detecting MITM attacks in ehealthcare BLE systems, J. Med. Syst. 43 (11) (2019) 1–18.

Andrea Lacava received his B.S. in Computer Engineering and his M.S. in Cybersecurity from Sapienza, University of Rome, Italy in 2018 and 2021, respectively.

He is currently pursuing a double Ph.D. degree in Computer Engineering at the Institute for the Wireless Internet of Things at Northeastern University, MA, USA and in Information and Communication Technology (ICT) at Sapienza, University of Rome, Italy.

His research interests focus on the O-RAN architecture, 5G and beyond cellular networks and Bluetooth Low Energy.

Valerio Zottola received his B.S. in Computer Science and his M.S. in Cybersecurity from Sapienza University of Rome, Italy in 2019 and 2022, respectively.

He is currently working as security analyst at Leonardo S.p.A.

His fields of interest are malware analysis, machine learning and incident response.

Alessio Bonaldo received his B.S. in Computer and System Engineering at La Sapienza University of Rome in 2019 and is currently attending the M.S. in Cybersecurity, with his graduation scheduled for the end of 2023.

At the moment he holds the position as a consultant in a computer company.

He is occupied in the developments for cloud-based solutions and services.

Francesca Cuomo received the Ph.D. in Information and Communications Engineering in 1998 from Sapienza University of Rome. From 2005 to October 2020 she was Associate Professor and from November 2020 she joined "Sapienza" as Full Professor teaching courses in Telecommunication and Networks. Prof. Cuomo has advised numerous master students in computer engineering, and has been the advisor of 13 PhD students in Networking.

Her current research interests focus on: Vehicular networks and Sensor networks, Low Power Wide Area Networks and IoT, 5G Networks, Multimedia Networking, Energy saving in the Internet and in the wireless system.

Francesca Cuomo has authored over 156 peer-reviewed papers published in prominent international journals and conferences. Her Google Scholar h-index is 30 with over 3850 citations.

Relevant scientific international recognitions: Two Best Paper Awards. She has been in the editorial board of Computer Networks (Elsevier) and now is member of the editorial board of the Ad-Hoc Networks (Elsevier), IEEE Transactions on Mobile Computing, Sensors (MDPI), Frontiers in Communications and Networks Journal. She has been the TPC co-chair of several editions of the ACM PE-WASUN workshop, TPC Co-Chair of ICCCN 2016, TPC Symposium Chair of IEEE WiMob 2017, General Co-Chair of the First Workshop on Sustainable Networking through Machine Learning and Internet of Things (SMILING), in conjunction with IEEE INFOCOM 2019; Workshop Co-Chair of AmI 2019: European Conference on Ambient Intelligence 2019. She is IEEE senior member.

Stefano Basagni is with the Institute for the Wireless Internet of Things and a professor at the ECE Department at Northeastern University, in Boston, MA. He holds a Ph.D. in electrical engineering from the University of Texas at Dallas (December 2001) and a Ph.D. in computer science from the University of Milano, Italy (May 1998). Dr. Basagni's current interests concern research and implementation aspects of mobile networks and wireless communications systems, wireless sensor networking for IoT (underwater, aerial and terrestrial), definition and performance evaluation of network protocols and theoretical and practical aspects of distributed algorithms. Dr. Basagni has published over twelve dozen of highly cited, refereed technical papers and book chapters. His h-index is currently 46 (May 2022). He is also co-editor of three books. Dr. Basagni served as a guest editor of multiple international ACM/IEEE, Wiley and Elsevier journals. He has been the TPC co-chair of international conferences. He is a distinguished scientist of the ACM, a senior member of the IEEE, and a member of CUR (Council for Undergraduate Education).