Received 20 December 2021; revised 14 January 2022; accepted 4 February 2022. Date of publication 7 February 2022; date of current version 16 February 2022.

Digital Object Identifier 10.1109/OJCOMS.2022.3149732

Security and Privacy Threats for Bluetooth Low Energy in IoT and Wearable Devices: A Comprehensive Survey

ARUP BARUA[1], MD ABDULLAH AL ALAMIN[1] (Graduate Student Member, IEEE), MD. SHOHRAB HOSSAIN[1] (Member, IEEE), AND EKRAM HOSSAIN[2] (Fellow, IEEE)

[1]Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka 89120, Bangladesh

[2]Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB R3T 2N2, Canada

CORRESPONDING AUTHOR: E. HOSSAIN (e-mail: ekram.hossain@umanitoba.ca)

ABSTRACT Bluetooth Low Energy (BLE) has become the de facto communication protocol for the Internet of Things (IoT) and smart wearable devices for its ultra-low energy consumption, ease of development, good enough network coverage, and data transfer speed. Due to the simplified design of this protocol, there have been lots of security and privacy vulnerabilities. As billions of health care, personal fitness wearable, smart lock, industrial automation devices adopt this technology for communication, its vulnerabilities should be dealt with high priority. Some segregated works on BLE were performed focusing on various vulnerabilities, such as the insecure implementation of encryption, device authentication, user privacy, etc. However, there has been no comprehensive survey on the security vulnerabilities of this protocol. In this survey paper, we present a comprehensive taxonomy for the security and privacy issues of BLE. We present possible attack scenarios for different types of vulnerabilities, classify them according to their severity, and list possible mitigation techniques. We also provide case studies regarding how different vulnerabilities can be exploited in real BLE devices.

INDEX TERMS Bluetooth Low Energy (BLE), BLE vulnerabilities, passive eavesdropping, device fingerprinting, privacy attack, IoT, wearable device, security tools.

I. INTRODUCTION

BLUETOOTH Low Energy [1], [2], also known as Bluetooth Smart, is the most widely used communication protocol for IoT devices. It is a Wireless Personal Area Network (WPAN) technology developed and maintained by Bluetooth Special Interest Group (SIG). Nokia initially developed BLE as Wibree [3] and now adopted by Bluetooth SIG [4], which currently has over 20000 members. BLE was first introduced in Bluetooth Core Specification 4.0 [1] in June 2010, and it has some distinct features compared to the classic Bluetooth [2]. BLE protocol was originally designed for short-range communication with smart IoT sensors and devices with very limited power consumption, e.g., a BLE device running in a small battery can last for 1-4 years [5].

The Internet of Things (IoT) has been increasingly implemented in industrial systems, health-care systems, military applications, beacons, smart home products, and other applications over the previous decade. As of 2019, there are 14.2 billion linked IoT devices, with that figure predicted to rise to 25 billion by 2021. Most of these IoT devices use BLE for data communication and Internet connectivity. The widespread use of Classic Bluetooth [1] in billions of mobile phones, laptops aided the adaptation of BLE as they share a similar implementation. Both Classic Bluetooth and BLE are supported by almost all modern operating systems, including Windows 10, Linux, Android, and Mac OS.

BLE has been most popular among other low power wireless solutions, such as ZigBee [6], [7], Z-wave, and Wavenis [8]. BLE is more energy conservative than its competitors, such as ZigBee and ANT protocols [9]. BLE is distinguished from other wireless technologies (such as Bluetooth and WiFi) by its extremely low power consumption. It has a simplified protocol stack,

which is to blame for many of its security and privacy flaws.

Due to the fact that BLE is now used by billions of devices, it is critical to investigate its security vulnerabilities. All of these personal and industrial devices simplify our lives and increase our productivity, but they also expand the attack surface of these systems. As BLE is widely used in health care applications, its security and privacy risks could be fatal [10]. A BLE connection is nearly indestructible once established. However, insecure pairing, inappropriate authentication, and poor protocol implementation (e.g., lack of suitable cryptography) expose BLE devices to eavesdropping, pin cracking, Man-In-The-Middle (MITM), and other attacks. Security vulnerabilities cause a smuggle of personal data, unlocking smart locks, misinterpretation of the message, battery drain for IoT devices, etc.

Several works on security and privacy threats [11], [12], [13] for BLE have been published in a separate manner, with the authors analyzing the security architecture and performing some attacks exploiting the protocol's insecure implementation. Many security researchers from academia and industry presented various attacks [14], [15], [16], [17] on Bluetooth Low Energy devices at various security conferences and conventions such as Defcon. However, to the best of our knowledge, there has not been any study that provides a comprehensive survey of all possible exploits in BLE technology that could be exploited to attack BLE devices. The *objective* of this paper is to identify important security and privacy risks for BLE, to classify those threats, and to make recommendations on how to mitigate those attacks. This paper performs a *comprehensive survey* of existing works and provides a taxonomy of security and privacy issues in BLE protocol that is used in billions of mobile phones, IoT and wearable devices. This will tremendously benefit researchers, software engineers, and manufacturers of IoT devices that are developing these protocols.

The *contributions* of this paper are can be summarized as follows:

- Identify the flaws in BLE security architecture that makes it vulnerable to many security and privacy threats,
- Classify BLE threats based on different types of vulnerabilities found in several popular (latest) versions of the BLE protocol,
- Provide recommendations to mitigate attacks against BLE,
- Present several case studies of real attacks on IoT devices (with a open source tools to execute those attacks) and discuss lessons learnt when security recommendations are not taken seriously by the device manufactures,
- Introduce some new attractive domains of BLE that might assist to enhance user satisfaction, and
- Identify open research directions and opportunities in this field.

Note that, there have been several complete surveys on the beacon or iBeacon [18], [19] protocol that uses Bluetooth Low Energy protocol for proximity or location-based services. As a software extension of the BLE protocol, the beacon has several different vulnerabilities and security measures. The security and privacy issues related to beacons are not discussed in this paper.

The rest of the paper is organized as follows. In Section II, we discuss related literature on IoT, classic Bluetooth, and BLE. In Sections III and IV, we briefly describe the protocol stacks, security architecture, and pairing methods of Bluetooth and BLE. In Section V, we classify and discuss the security and privacy threats for BLE and provide possible countermeasures. Several case studies of recent attacks against BLE are discussed in Section VI to understand how security recommendations are ignored by device manufacturers. Section VII gives an overview of some well-known tools to perform BLE attacks. In Section VIII, we discuss the latest and upcoming features of BLE that can lead to new research directions. Finally, Section IX concludes this paper.

II. RELATED WORKS

In this section, we provide an extensive overview of the prior works on IoT security, classic Bluetooth (that include BLE security), and segregated works on BLE security and privacy. First, we review few works on IoT and its security threats in Section II-A. Then, we offer some notable surveys on classic Bluetooth in Section II-B. Then, we present existing works on the architecture of BLE, its security threats in smart wearable, and IoT devices in Section II-C. Finally, we identify the fact that there is a lack of comprehensive survey work on BLE security and how our work can *bridge* the knowledge gap between security researchers and BLE device manufactures.

A. EXISTING WORKS ON IoT THREATS

There have been a number of comprehensive surveys [20], [21], [22] on the applications [23] of IoT devices for home and industrial purposes [24], their interconnectivity [25], network topology [26] and the network protocols [27], [28], [29] that are widely used for connectivity. Hwang [30] discussed the privacy issues of personal information in the era of IoT services and Pacheco and Hariri [31] suggested a security framework for IoT infrastructure and protection mechanisms from cyber attacks. Apart from these works, there are also some excellent surveys and works of literature [32], [33], [34], [35], [36], [37], [38], [39], [40] on security and privacy issues of IoT devices used in home appliances [41], industries, health applications [42], [43], sensor based IoT devices [44] and provide countermeasures [45] to mitigate [46] these attacks as well as future research opportunity [47], [48], [49], [50].

All the above works focused on the overall security and privacy threats of IoT and wearable devices. However, none of these works provide any comprehensive study on BLE

and all the possible security attacks for current and previous versions of this technology.

Several recent survey studies examine the security of front-end sensors, back-end system [51]; others concentrate on protecting specific IoT services [52]. Zhou *et al.* [53] discuss the security implications of various unique features found in IoT devices. Each of these pieces of literature focuses on a particular aspect of IoT devices and aims to unpack security concerns and prevention. Nonetheless, security concerns raised by the BLE protocol go unaddressed.

Numerous survey papers [51], [54], [55] refer to the IoT devices' communication medium as a wireless network and discuss the attack vector and various countermeasures. None of these papers discuss the security concerns raised due to the vulnerabilities in the BLE protocol.

B. SURVEYS ON SECURITY THREATS AGAINST CLASSIC BLUETOOTH

There have been few surveys [56], [57], [58], [59], [60] on Classic Bluetooth technology and its vulnerabilities. Be-Nazir Ibn Minar and Tarique [61] provided an extensive survey on Bluetooth threats, countermeasures, and future security issues. Hassan *et al.* [62] presented a complete attack taxonomy of Bluetooth and mitigation techniques. These works do not focus on BLE, and so they do not provide necessary information to understand key security issues in BLE. The BLE vulnerabilities are quite different from classic Bluetooth and it has its own attack vectors that will be discussed later in this paper.

C. EXISTING STUDY ON BLE AND OUR CONTRIBUTIONS

There have been a number of works on the functionalities [63], [64], performance [7], and applications [65] of BLE [66], [67], [68] in IoT devices. Many researchers worked on different applications of BLE especially medical health monitoring devices [69], [70], heart rate sensors [71], [72], smart homes [73], [74], smart vehicles [75], smart locks [76], [77], [78], finding location [79], wristbands [80]. All these papers confined their focus on few vulnerabilities of BLE, and lack complete attack scenarios and proper mitigation techniques.

Gomez *et al.* [66] discussed the main features and applications of BLE. Siekkinen *et al.* [7] discussed the BLE architecture and its lower power consumption. Ryan [11] and Kwon *et al.* [12] discussed on the architectural flaw of BLE and its poor implementation of encryption and different key exchange [81], [82]. Sevier and Tekeoglu [83] demonstrated some common vulnerabilities on BLE devices using open-source hardware and tools. These papers, however, cover BLE architectural flow, but do not incorporate complete attack procedures, privacy issues, and mitigation techniques.

Cha *et al.* [84] illustrated security issues of random MAC addresses used in BLE smart vehicles. Uher *et al.* [13] pointed out how various types of Denial of sleep (DoSL) attacks can sabotage a network of IoT sensors. Das *et al.* [85] discussed how user privacy such as user activity (sitting, walking, sleeping), could be detected only by passively observing the data communication packets between a BLE wearable device and a smartphone. These works focus on specific privacy issues of BLE in a *segregated manners*, and do not provide a comprehensive mitigation techniques.

O'Sullivan [86] surveyed and described different attacks in BLE in 2015. However, many variations of recent privacy attacks on smart wearable devices, security features of the latest version of BLE (v5) and countermeasures techniques are missing in this work.

The only work that comes closest to our objective is the work Zhang *et al.* [87] which focused on the well known security and privacy vulnerabilities such as (MiTM attack, DoS, eavesdropping) of the BLE protocol. The authors discussed BLE architectural design and provided different attack vectors and countermeasures. But it only covers general BLE attacks and does not provide a *comprehensive taxonomy* of attack vectors, multiple attack techniques, and countermeasures proposed by a wide range of researchers. Our work covers case studies of real exploitation of these vulnerabilities in wearable and IoT devices, details description of the tools, software to carry out these attacks and new features as well as security measures of latest BLE v5.

Apart from the above works, there is also some other literature on the security and privacy vulnerabilities of BLE in IoT, wearable devices. But to the best of our findings, our paper is the only work that provides an *comprehensive survey* of the BLE threats and provides threat-wise detailed recommendations to make secured BLE applications. It classifies and analyzes real exploits of BLE vulnerabilities in older as well as latest versions, and provides recommendations to mitigate threats. Our work mainly focuses on BLE technology and its security and privacy vulnerabilities found in existing wearable and IoT devices.

III. CLASSIC BLUETOOTH TECHNOLOGY

Bluetooth [2], [88] technology was invented as a replacement of the cable, to exchange data continuously and in a wireless network with other peripheral devices, such as the cell phone, laptop, head-phone within a short-range. Bluetooth was initially standardized as IEEE 802.15.1 [89], but now the Bluetooth SIG maintains the development of the specification and its trademark. Bluetooth operates between 2.402 and 2.480 GHz in unlicensed but not unregulated ISM band. Bluetooth transmits data as a packet in a master-slave arrangement, where a master can connect up to seven slaves. This network composition is known as a piconet.

Bluetooth is used in pretty much every communication device. There are various reasons for using this technology, such as quickly transfer photos, songs, contacts, files or even share Internet connection. People can also make a hands-free call through the smartphone, smartwatch, or even car kits using Bluetooth. Devices that work in pairs, such as computer-mouse, computer-keyboard are now also using Bluetooth technology as a replacement of the cable.

Nevertheless, it consumes relatively low power, so that devices such as a cell phone, wristwatch having limited battery backup do not face any issue.

A. BLUETOOTH VERSIONS

Bluetooth SIG released five major Bluetooth versions. All versions are backward compatible. Some significant features in each version are discussed as follows:

- *Bluetooth 1.x:* The first version of Bluetooth was released in May 1998. Nowadays, it is rarely used. It has many security issues in pairing and has a limited speed limit (up to 1 Mbit/s).
- *Bluetooth 2.x:* This version was released in 2005. It was very popular among the featured mobile phones because of it's simplified pairing process. It supports up to 3 Mbit/s of transfer speed.
- *Bluetooth 3.x:* Specification of Bluetooth version 3 was adopted by Bluetooth SIG in April 2009. It has a higher data transfer speed (up to 24 Mbit/s) than its previous versions. But at the same time, higher speed costs higher power consumption.
- *Bluetooth 4.x:* Released in June 2010, this version comes with the most fascinating Low Energy (LE) feature. This specialty allows BLE to incorporate low battery IoT devices. It has higher speed capabilities and supports a 50-100m connectivity range. Bluetooth 4.1 comes with the feature of indirect IoT device connection. Previously BLE IoT devices needed to connect to a smartphone to use the Internet. However, Bluetooth 4.2 added an IPv6 layer to the BLE protocol stack. As a result, IoT devices can use the BLE protocol as a platform to support IPv6 communication.
- *Bluetooth 5.x:* This version was released in 2016 and has significant performance improvement compared to all of its predecessors. We will provide a detailed description of this version in Section VIII-A.

B. CLASSIC BLUETOOTH VS. BLE

Bluetooth classic and Bluetooth smart is designed considering different use cases in mind. There are a few fundamental differences between these two protocols on their technical specification, design, implementation, and use cases. Table 1 records the difference between classic Bluetooth and BLE [90], [91]. Classic Bluetooth is useful for managing large amounts of data, and BLE is useful for exchanging a small amount of data periodically with very little energy consumption. Between these two variations, the usability depends on the use cases and restrictions of the device. Both classic Bluetooth and BLE use the same spectrum range, but with a different number of channels. BLE is better than classic Bluetooth for near field communication with higher battery durability. Due to the tradeoff between high data rate and energy consumption, BLE has to give up a high data transfer rate to preserve low energy consumption. On the contrary, classic Bluetooth has a higher bit rate with higher energy consumption.

TABLE 1. Differences between classic Bluetooth and BLE.

Characteristics	Bluetooth	BLE
Distance	100 m	$<$100 m
Physical channels	79 channel with 1 MHz spacing	40 channel with 2 MHz spacing
Topology	Scatternet	Star bus
Active slaves	7	Unlimited
App throughput	0.7-2.1 Mbit/s	0.27-1.37 Mbit/s
Power consumption	Low (less than 30 mA)	Very low (less than 15 mA)
Max data rate	1-3 Mbps	1 Mbps via Gaussian Frequency Shift Keying (GFSK) modulation
Modulation	GFSK (Modulation index is 0.35), 8DPSK , $\pi/4$ DQPSK	GFSK (Modulation index is 0.5)
Latency in data transfer	Approx. 100 ms [92] from a non-connected state	Approx. 6 ms [92] from a non-connected state
Voice capabilities	Yes	No
Peak current consumption	$<$30 mA	$<$15 mA
Message size (bytes)	358 (max)	8 to 47
Error correction	8 bit Cyclic Redundancy Check (CRC)	24 bit CRC
Security	64b/128b, application layer	128 bits Advanced Encryption Standard (AES), user defined application layer
Spreading	FHSS (1MHz channel)	FHSS (2MHz channel)
Device Address Privacy	None	Private device addressing available
Discovery/Connect	Inquiry/Paging	Advertising
Use cases	Mobile phones, gaming, headsets, wearables, PCs, healthcare, automotive, sports & fitness etc.	Mobile phones, gaming, automotive, wearables, PCs, fitness & sports, security, healthcare, industrial, etc.

IV. BLE TERMINOLOGIES AND PROTOCOL STACK

BLE is specified in Bluetooth version 4.0 [2] by the Bluetooth SIG. It is not an upgrade of classic Bluetooth, rather a new technology that is designed especially for low-powered IoT devices. These two protocols have different design specifications and applications [66]. Some of the key concepts of BLE architecture are described in this section.

- *Classic Bluetooth (BR/EDR):* BR stands for basic rate, and EDR stands for advanced data rate. This wireless communication is defined from Bluetooth specification 1.0.
- *Bluetooth Low Energy:* A new low power wireless communication protocol from Bluetooth core specification 4.0. It uses the Bluetooth brand name and has different applications than classic Bluetooth.
- *Single Mode:* If a device implements a single-mode, then it means it can communicate with only a BLE device but not with devices that only support BR/EDR.

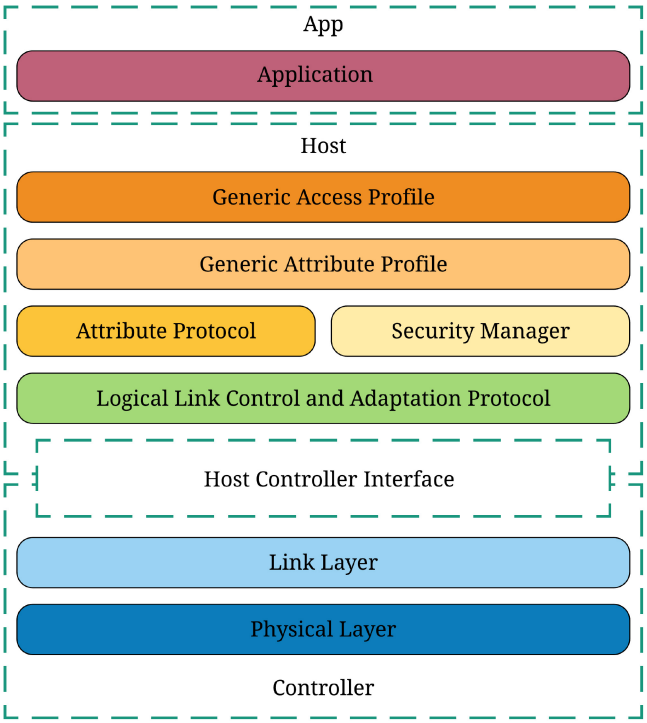

FIGURE 1. BLE protocol stack.

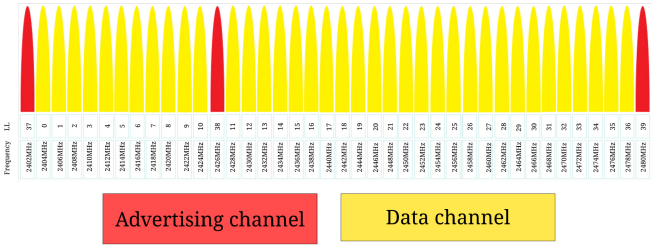

FIGURE 2. BLE radio frequency.

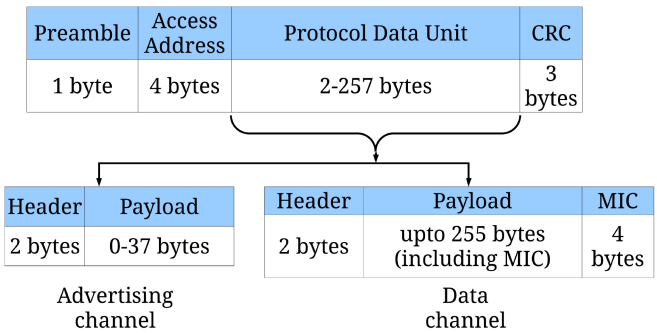

FIGURE 3. BLE packet format.

- *Dual Mode:* If a device implements dual-mode, then it can communicate with both classic Bluetooth and BLE devices.

Some of the core terminologies, the protocol stack, pairing mechanisms used in BLE different BLE versions (v4.0, v4.1, v4.2, and v5.0), strength, and weakness of BLE protocol are discussed as follows.

A. OVERVIEW OF BLE PROTOCOL STACK

The main building blocks of the BLE protocol stack are illustrated in Fig. 1 and they are: the controller, the host and the application layer. These layers are discussed in details as follows.

1) CONTROLLER LAYER

This is the lowest part of the BLE protocol stack (Fig. 1). It includes the hardware to transmit and receive data. This module contains the physical layer and the link layer.

Physical Layer: Bluetooth and BLE are incompatible with each other due to the differences in their physical layer and link-layer implementations. As shown in Fig. 2, BLE uses 40 radio frequency (RF) channels with 2 MHz spacing [66] whereas classic Bluetooth uses 79 different frequency channels, advancing every 1 MHz. There are mainly two types of channels: advertising channels and data channels. There are three advertising channels, which are used for device discovery, broadcast device information, and connection establishment, respectively. The remaining 37 data channels are used for data transmission between connected devices. During data transmission, BLE uses Adaptive Frequency Hopping (AFH) to avoid interference and rapidly hops between those 37 channels. At the physical layer, BLE uses Gaussian Frequency Shift Keying (GFSK) modulation with a 250 kHz offset.

The physical layer consists of the radio hardware that performs the modulation and demodulation of the analog signal. As BLE is designed for low-end sensors, various security mechanisms are implemented with a simpler design, such as the rate of channel changing, whitening seed, etc. This compromise makes it easier for the attacker to sniff communication and run other malicious activity passively.

BLE Packets: BLE packets (shown in Fig. 3) are of two types: data packet and advertising packet. Both packets start with a one-byte preamble. It is followed by the 4-byte access address, which is used to identify radio communication in the physical layer. After that, comes 2-257 byte of Protocol Data Unit (PDU). Data and advertising channels are different in PDU. An advertising channel PDU consists of a 2-byte advertising packet type header and 0-37 byte payload. A data channel PDU consists of a 2-byte data channel header, followed by 0-255 byte payload respectively. The payload in the data channel starts with 4-byte L2CAP header and ends with a 4-byte Message integrity check (MIC). Finally, every packet ends with a 3-byte CRC.

Bluetooth Device Address: Similar to classic Bluetooth, BLE devices use a 6-byte address known as Bluetooth Device Address (BD_ADDR), which can be used to identify the device uniquely. However, a BLE device can also have a random address that is programmed or generated at runtime in a device. This random address can either be static or private. The static random address does not change, but the

private random address changes periodically. A private random address can either be resolvable or non-resolvable. The former is resolved at the link layer (as discussed below), but the later cannot be resolved by any device and its sole purpose is to prevent tracking.

Resolvable Private Addressing (RPA): From Bluetooth version 4.2, there is a mechanism for using Resolvable Private Addressing (RPA) resolution for BLE devices. The primary concern of RPA is to maintain the secrecy of a BLE device. If a device uses the same BD_ADDR for advertisement, it can be easily identified by the attacker. But at the same time, using a vague random number as the address can not solve the issue, because previously paired devices cannot detect the device either. Each paired device maintains a local Identity Resolving Key (IRK). This IRK is used to resolve the RPA.

Link Layer: This is the very first software layer of BLE protocol which handles the different states of connection and all the security checking (CRC, MIC) of data packet. The link layer has five different states:

- *Standby:* It is the default idle state of a device.
- *Advertising:* In this state a device advertises itself to connect with other device through advertising packets.
- *Scanning:* It is state when a device listens for advertising packets to initiate connection.
- *Initiating:* After selecting a specific advertiser, a connection is initiated with the intended device.
- *Connection:* It is the data transmission phase between two connected devices.

Different manufacturer use different hardware, and they implement the link layer for those custom hardware. It is the responsibility of the link layer to create a communication link between two BLE devices using physical data channels.

BLE defines two roles for the link layer: master and slave, respectively. The master initiates connection and slave advertises. A master can be connected with multiple slaves. Such a master-slave network of BLE devices is also called a piconet as in classic Bluetooth. The slaves are usually less powerful devices, and they sleep by default and wake up periodically to receive packets from the master.

2) HOST CONTROLLER INTERFACE (HCI)

Bluetooth specification defines the HCI protocol, which specifies a format to facilitate two-way communications between the host and the controller. It receives events from the controller and can even send a command to the controller. If the host and controller part are developed by separate manufacturers, it is the responsibility of HCI to assure standard communication between them.

3) HOST LAYER

This is the middle layer of the BLE protocol stack that enables applications to scan, discover, connect, and exchange information with peer devices in a standard and interoperable way. It consists of a Logical Link Control and Adaptation Protocol (L2CAP), Attribute Control Protocol (ATT), Security Manager (SM), Generic Attribute Profile (GATT), and Generic Access Profile (GAP). Various modules of the host layer are described as follows.

- *L2CAP:* L2CAP is one of the essential layers of the BLE protocol stack. It is the simplified and optimized version of classic Bluetooth L2CAP [66]. The L2CAP is primarily responsible for establishing two devices as a protocol multiplexer. L2CAP receives multiple communication protocols from the upper layers (ATT, SMP) and encapsulates them in BLE packet formats. This layer is also responsible for segmentation and reassembly of variable length data packets for transmission in a best-effort approach [66].
- *Security Manager:* SM is a service that defines the overall security of the BLE protocol. This layer has two major roles. First is Security Manager Protocol (SMP) which controls the pairing mechanism (Section IV-C), key distribution and key management of a device. Second role is Security toolbox whose primary task of it is to encrypt and decrypt data. It uses AES encryption to improve message integrity. Furthermore, it is responsible for hiding the MAC address to improve privacy and for identifying and trusting remotely connected devices.
- *Attribute Control Protocol:* ATT and GATT are introduced in Bluetooth's core specification, and every BLE profile uses them. ATT is a client-server-based stateless low-level protocol that defines data exchange between a client and a server. The protocol also defines a server and client's responsibility: a server should expose some attributes that a client can discover, read, and write. For a given connection, a device can act as a client or server or both, independent of whether it is a master or a slave device.

  An attribute has four elements. A 16-bit handle to uniquely identify an attribute, a 16-bit unique identifier (UUID) defines the attribute type, a value to read and write, and some associated permissions available for a client, respectively.
- *Generic Attribute Profile:* GATT is built on top of ATT. It organizes the attributes into services and defines a framework to operate these services. It also defines some BLE libraries. It systematically manages how data is formatted and then exchanged between devices. During the data transfer, it defines the responsibility of a client and a server. The client sends requests to the GATT server and it can read/write data on the server. The main responsibility of the server is to store the data and when the client makes a request, it makes those attributes (i.e., data) available to them.

A peripheral contains one or many GATT profiles, which consist of many services. Each service is composed of characteristics and distinguished by Universally Unique ID (UUID). Each characteristic contains zero or several descriptors. All these services, characteristics, and descriptors are in the form of attributes. UUID is a very commonly used

terminology in BLE. This is a unique number to uniquely identify BLE attributes, i.e., services and characteristics. The UUIDs are public, and the peripheral devices broadcast these IDs. This is how the central devices know which services the peripherals are providing and how to access them. There are two types of UUIDs. One is public UUID (16 bit) predefined by Bluetooth specification. Another type is vendor-specific UUID (128 bit) that is used for vendor's custom services.

Generic Access Profile (GAP) is the highest level protocol, which makes the device visible to the outside world. It takes care of device discovery and connection establishment. It also focuses on the security aspects of the protocol such as different security modes and their connection procedures. This defines how devices interact with each other at a lower level.

In the basic sense, GAP defines distinct roles for devices, such as Central, Peripheral, Broadcaster, Observer. A BLE device can be a central or a peripheral depending on the situation. But at a given time, it can only play one role, not both.

- *Central (Master):* These devices are more computationally powerful, such as mobile phones, computers, tablets. During the data exchange, these devices do heavy computation. It is the responsibility of central devices to initiate and manage connections.
- *Peripheral (Slave):* These devices are low-end small devices with less computational power. Most of the peripherals are sensors that advertise to be connected with the central devices such as heart rate monitor, proximity sensors, etc.
- *Broadcaster:* It broadcasts data via different advertising channels. They are the peripheral devices and they receive the data sent by the central devices.
- *Observer:* An observer is a central device, that listens for broadcast packets and initiates the connection.

4) APPLICATION LAYER

This is the highest level interface in the BLE stack, and it contains the interface, the application logic, and the structure of the application in the BLE devices. This layer directly communicates with the host layer of the BLE protocol. The BLE app developers should have a functional understanding of this layer.

B. CONNECTION PROCESS

To securely exchange data between two BLE devices, a secured link layer connection needs to be established. The connection process comprises of two separate acts.

- The peripheral (slave) device sends advertising packets periodically and waits for the connection request from the master device.
- The central (master) device scans for the advertisement packets and initiates the connection establishment process.

A connection event occurs when two devices exchange information periodically after being connected. This helps BLE devices to preserve energy because devices remain in sleep mode, and wake up only when there is a connection event, exchange packets and then go back to sleep mode again.

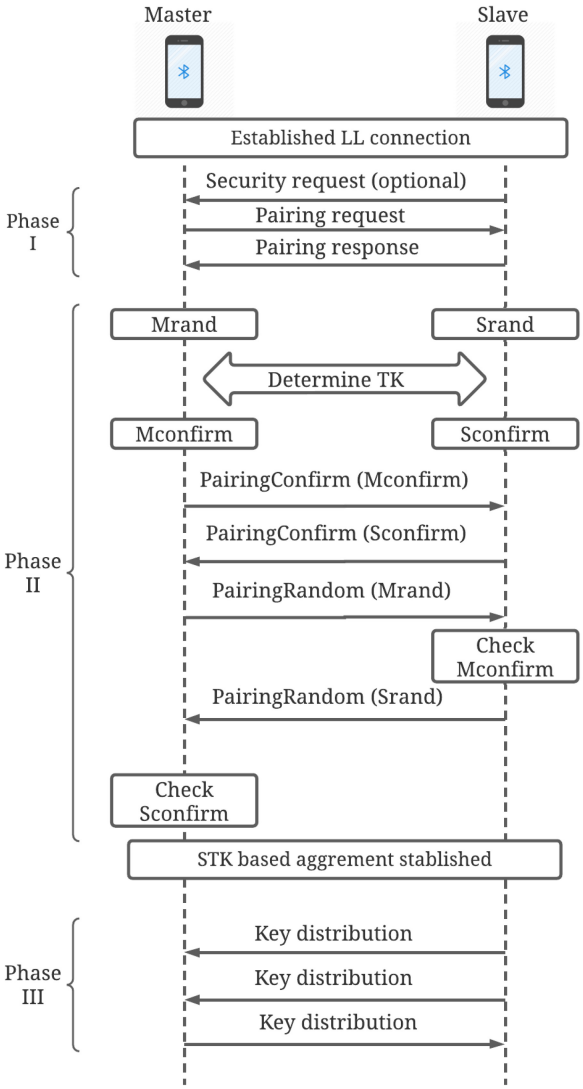

FIGURE 4. BLE connection phases.

BLE specification uses AES-CCM cryptography to encrypt the data packets. AES encryption is considered very secure but the key exchange mechanisms that the BLE protocol uses introduce some vulnerabilities that attackers might exploit to break the encryption. In Bluetooth smart, there are mainly two types of connection options: Legacy connections and Secured connections [93], [94].

1) LEGACY CONNECTIONS

The connection process used in BLE protocol (v4.0 and v4.1) is known as BLE Legacy connections. In legacy connections, BLE implements a custom key exchange protocol, which is the root cause of various BLE threats, such as passive eavesdropping, MITM, offline pin cracking, etc.

The Security Manager (explained in Section IV-A3) completes this connection process in three phases which is shown in Fig. 4.

- *Phase I:* This phase begins when a BLE device sends a pairing request to establish a connection. Then the two devices exchange various important information such as their I/O capabilities (display, keyboard), link key size, authentication, etc. Based on this information, the BLE device chooses appropriate pairing methods (Section IV-C). One important thing about this phase is all the data exchanged in this phase are in plain text. So if an attacker is active in this phase, then it can easily decrypt the communication, furthermore can perform active MITM attack.
- *Phase II:* Phase 2 begins after the successful completion of phase 1. In this phase, the devices generate a 128 bit random value (Mrand/Srand) afterwords choose a pairing method based on their I/O capabilities (Section IV-C). This pairing method is used for the agreement on the Temporary Key (TK). Next both devices compute Mconfirm and Sconfirm from the generated random values, TK and other device parameters. After that both of them exchange the confirm values to ensure they are using the same TK. After this, the devices generate Short Term Key (STK) using TK and random values. This STK is used to encrypt the communication between these two devices. An important thing to note that both Mrand and Srand are exchanged as plain text. If any attacker can brute force the TK it can easily calculate Mconfirm, Sconfirm and finally the STK. Secure transaction of TK is necessary to ensure secrecy of STK and other keys.
- *Phase III:* This phase is optional and depends on the information shared in phase 1. This phase is called bonding. Many transport-specific keys are exchanged encrypted by STK in this phase. Most of the keys are generated by slave but master also deliver some keys in case the roles are reversed in future. The keys are described as follows.
  - *IRK (BLE v4.2 & upward devices):* It is used to resolve and generate a random address.
  - *Connection Signature Resolving Key (CSRK):* It is used to digitally sign the data PDU. The receiver also verifies the data PDU with this CSRK.
  - *Long term key (LTK):* It is generated by slave and used to encrypt the current and future sessions.
  - *Encrypted Diversifier(EDIV) and RAND:* It is generated by slave and used to create and resolve LTK.

2) SECURE CONNECTIONS (BLE V4.2 & UPWARD DEVICES)

BLE Secure connections were introduced in BLE v4.2. The latest BLE devices are fully backward compatible with older BLE (v4.0 and v4.1) devices. BLE v4.2 and above devices can perform legacy connections as well as BLE secure connections.

In the secure connections, the above-described phase 1 and phase 3 are the same. The only difference is in phase 2. The secure connections do not use a custom key exchange protocol like the legacy connections. They do not use TK and STK, rather use a single LTK to encrypt the communication. It provides more security because this LTK generation/exchange is based on the Elliptic Curve Diffie Hellman (ECDH) public-key cryptography. After this, the devices use one of the pairing methods (Section IV-C) to authenticate the connection. An LTK is generated on both devices and the communication between these two devices is encrypted using this LTK.

C. PAIRING METHODS

BLE has mainly four pairing methods [81], [94] and they are discussed below.

1) JUST WORKS

This is the simplest and most delicate form of pairing method as it does not require anything to authenticate other devices. The devices that do not have any display or input interfaces, such as headphones, sensors, etc. have to use this method for pairing. Table 2 provides a comparison among the different pairing methods in terms of the I/O capabilities needed in the devices.

- *Just Works in Legacy Connections:* The TK is set to 0, and hence, it is very easy for the attackers to brute-force the STK and decrypt the communication. Additionally, this provides no mechanism to verify the devices participated in the authentication process, and so there is no MITM protection.
- *Just Works in Secure Connections:* BLE introduced ECDH key exchange in low energy (LE) secure connections from Bluetooth version 4.2, where it provides a good amount of security to prevent passive eavesdropping. In this pairing method, the connection initiating device and the non-initiating device use a random seed in the key generation process. However, it is still unprotected from MITM attacks as it does not provide any mechanism to verify the authenticity of the connection.

2) OUT OF BAND

In this pairing mechanism, the Temporary Key (TK) is shared between two connecting devices via different wireless Out Of Band (OOB) technology like NFC or tethering. The TK should be a unique random number with length up to 128 bits. If TK is random and a big number, then it provides sufficient protection from passive eavesdropping. The passive eavesdropping and MITM protection in this method is dependent on the OOB technology. If the attacker can sniff OOB communication, then it can perform eavesdropping and MITM attacks on BLE devices. Otherwise, this pairing provides sufficient protection from these attacks. For

TABLE 2. Decision table for pairing method selection.

		Initiator						
		Display	Display (Yes,No)		Keyboard	No I/O	Keyboard with Display	
			LE Legacy connections	LE Secure connections			LE legacy connections	LE secure connections
Responder	**Display**	Just Works	Just Works		Passkey	Just Works	Passkey	
	Display (Yes,No)	Just Works	Just Works	Numeric Comparison	Passkey	Just Works	Passkey	Numeric Comparison
	Keyboard	Passkey	Passkey		Passkey	Just Works	Passkey	
	No I/O	Just Works	Just Works		Just Works	Just Works	Just Works	
	Keyboard with Display	Passkey	Passkey	Numeric Comparison	Passkey	Just Works	Passkey	Numeric Comparison

legacy connections, the OOB offers the best security if the OOB channel is secured.

For BLE secure connections, this pairing method is fundamentally the same. The TK, public keys, nonces are shared between two connecting devices using OOB technology.

3) PASSKEY

If the device does not support OOB technology, then the passkey pairing method can be applied. Both the connecting devices should have keyboard input and display output capabilities for this pairing method. It provides a mechanism to verify connecting devices and provides considerable protection from MITM attacks. However, this method is vulnerable to other attacks, and for the highest level of protection, the BLE applications should use OOB or numeric comparison (Section IV-C4) pairing method.

- *Passkey in Legacy Connections:* One of the ways to implement this pairing method is to generate a random six-digit passkey and display it on one device. The user has to enter the same passkey on the other device that wants to connect. If the passkeys on both devices match, then the connection proceeds and a TK is be generated utilizing this number. If the attacker is not present during the pairing process, then this process provides relatively good protection from passive eavesdropping. However, if the attacker is present and sniffs all the pairing packets, then it becomes relatively easy for the attackers to brute-force the six-digit number and guess the STK.
- *Passkey in Secure Connections:* The passkey method for LE secure connections has better MITM protection than LE legacy connections. The pairing mechanism is fundamentally the same as the LE legacy connections. In addition to this, a 128-bit nonce is used to authenticate the connection.

4) NUMERIC COMPARISON METHOD

This pairing method is only available for BLE secure connections. This pairing method is very similar to the *Just works* pairing method with a key difference. Just works pairing method has no protection against MITM as it does not have any way to verify connecting devices. The way this pairing method protects against the MITM attack is by adding a layer on top of just works pairing methods. After exchanging keys, both devices generate a six-digit number and show it on their display. The user has to choose the correct number on the device to confirm the connection. This extra layer protects the device from the MITM attack.

5) SUMMARY OF THE PAIRING METHODS

The devices having no I/O capabilities can only use Just works pairing methods which makes these devices most vulnerable. Table 2 highlights a decision table, which specifies what pairing method can be used based on the initiator and responder device's I/O capabilities. It summarizes the final pairing methods based on the five I/O capabilities (display, display (y/n), keyboard, no I/O, keyboard with display) of the initiator and responder device. It shows that if any of the initiator or the responder device does not have any input devices then Just works pairing mechanism is used. Similarly, when both devices have keyboards or displays with keyboard, then in legacy connections Passkey paring method, and in secured connection, Numeric Comparison method is used, which is considered to be most secure.

V. THREATS AGAINST BLE

BLE security architecture is different from classic Bluetooth (discussed earlier in Section III-B). This is because BLE is designed to support communication with very low energy consumption, computationally-constrained sensors, and IoT devices. In BLE, there is a tradeoff between performance, security, and privacy concerns over low energy consumption [11]. BLE specification has a different mechanism for secured connection, such as link-layer encryption, device bonding, device white-listing. Unfortunately, a vast number of IoT devices available in the market have not implemented these security mechanisms properly. These have led to a wide range of security threats [11], [95]. As BLE protocol has low end-to-end security, an attacker can take over or disrupt a whole network of BLE devices deployed in an industrial automation system.

Fig. 5 illustrates a comprehensive taxonomy of security and privacy threats of BLE protocol. We group all the threats in eight categories based on the exploitation approach and severity. Attacks having similar strategy and attitude towards victim are coalesced into one category. Many of these threats share common underlying architectural or implementation

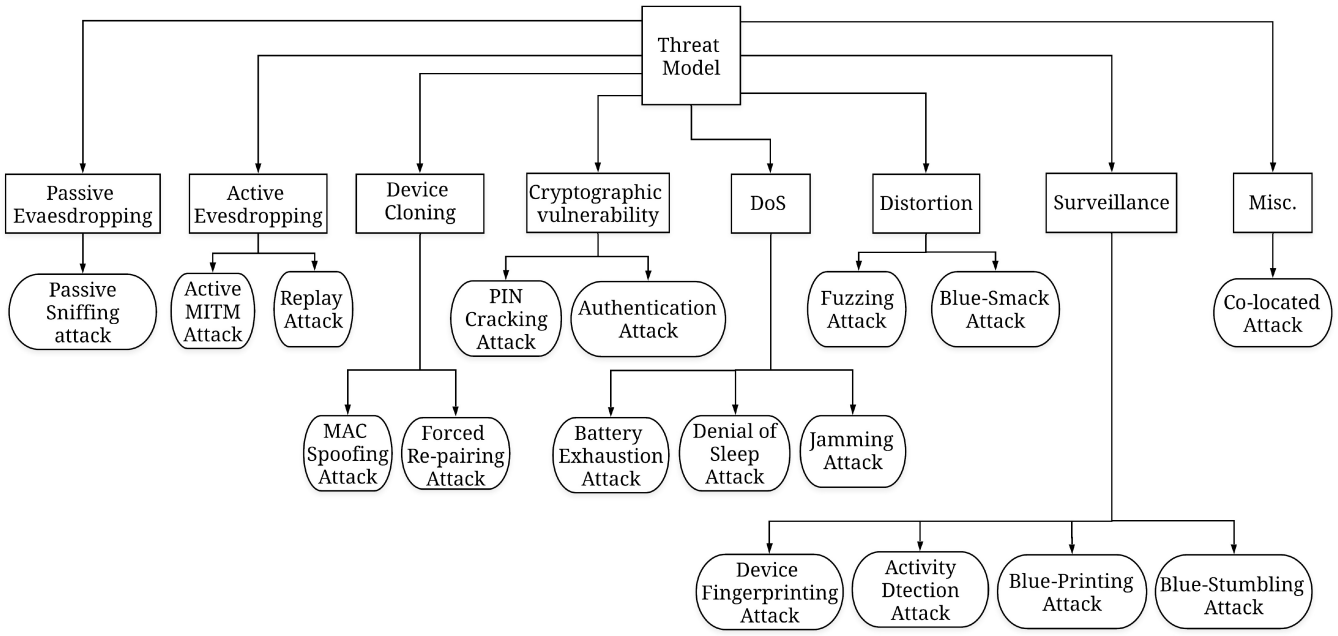

FIGURE 5. BLE threat model on the basis of the attack domain.

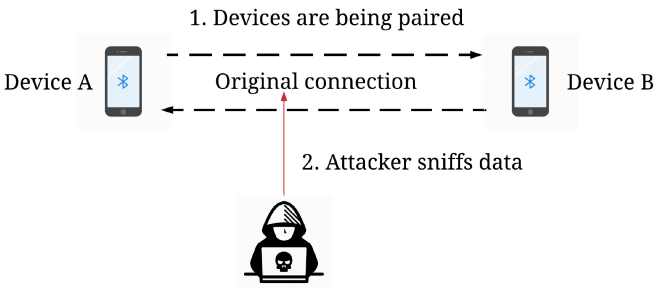

FIGURE 6. Passive sniffing attack.

flaws of the protocol, and hence, the same mitigation process applies to all of them. Some attacks are complementary to one another. Throughout this section, we organize our attacks following this threat model sequence.

A. PASSIVE EAVESDROPPING

Passive eavesdropping is the entry point of all types of BLE attack. Here the attacker secretly monitors all the communication between the connected devices, which eventually leads to various vicious attacks. Passive sniffing (shown in Fig. 6) is this type of attack where the attacker's main intention is to capture packets for evil aspire.

1) PASSIVE SNIFFING

The attacker somehow places himself in the path of data transmission which allows him to eavesdrop, and capture every data that is being transmitted. Wireless communication is especially vulnerable to a passive sniffing attack because the data is broadcast in the air and the attacker only needs a sniffing device to capture the radio communication. BLE is especially susceptible to this attack because of its simplified and predictable design of channel hopping.

Attacking Procedure and Severity: Most BLE devices are vulnerable to this type of attack because of the low security in the BLE protocol and its poor implementation of encryption and custom key exchanging method (discussed in Section IV-B) that enables the attacker to decrypt the communication quite easily [11], [80], [93], [96].

Just works pairing method in both BLE legacy connections (Section IV-B1) and secure connections (Section IV-B2) are vulnerable to passive eavesdropping. However, BLE secure connections provide better protection than legacy connections because, for a new connections, they use ECDH algorithm to generate the LTK, which is used to encrypt data during communication. On the other hand, legacy connections use a custom key exchange protocol, which is vulnerable to passive sniffing.

Passkey pairing method in legacy connections is also vulnerable because of the short-length of TK. If the attacker can sniff the packets that are exchanged during bonding, he can brute force the TK followed by STK, and eventually generate the LTK and decrypt all subsequent communication [12].

The attacker must know the access address, hop interval, hop increment, and CRC [11] to follow a connection. Hop interval and hop increment remain the same for an established connection, but access address varies over time. The attacker observes a single channel at a time to determine these three pieces of information and can be aware of an access address change by keeping track of missed packets [97]. With all this information an attacker can even sniff without the initial connection setup information in BLE legacy connections. But most of the eavesdropping attacks happen by deceiving the BLE devices to unpair and force them to renegotiate keys through different techniques, such as device cloning, jamming, injection-free

technique [98] so that the attacker can monitor the key exchange.

As BLE devices communicate in the wireless channel, anyone just with a sniffing device, such as Ubertooth (Section VII-A), SmartRF Sniffer (Section VII-B) can sniff communication as illustrated in Fig. 6. Once wireless communication between BLE devices is captured, the attacker can use different packet analyzing tools such as Wireshark (Section VII-D) to analyze the captured packets.

In BLE, link-layer encryption is used to protect the confidentiality of the data. But after successfully capturing the radio communication, the attacker can use different techniques such as pin cracking attack (Section V-D1) to crack the encryption. Successful sniffing can lead to various higher level dangerous attacks such as such as MITM attacks (Section V-B1), offline pin cracking attacks (Section V-D1), fuzzing attacks (Section V-F1), privacy leakage [85] which are described in details later. Even successful radio capturing can lead to some harmful attacks such as replay attack (Section V-B2) where sometimes the attacker does not even need to decrypt the packets.

Mitigation of Passive Sniffing: BLE specification provides countermeasures against passive eavesdropping by offering secure paring options described in Section IV-C. But many of the developers are not aware of such vulnerabilities and secured pairing methods. Some of the recommendations to mitigate this attack are outlined as follows.

- BLE Secure connections are always preferable to the legacy connections as they provide much better encryption.
- Just works pairing method should be avoided as it does not provide any protection from eavesdropping.
- Secure pairing options such as Numeric comparison, Out of bound, or at least Passkey entry should be used if possible.
- The users are recommended to use a secured, private environment to connect with their wearable devices for the first time.
- All data should be encrypted by AES-128 algorithm (nearly unbreakable) [99].

These security measures ensure that even if the attacker is successful in capturing transmitted radio packets, it is nearly impossible for him to decrypt the message without the shared key.

B. ACTIVE EAVESDROPPING

In these attacks, the attacker places himself in the communication path and steals information. MITM and Replay are two variations of active eavesdropping attack. In MITM, the attacker actively participates in the communication process and corrupts the integrity of data. Unlike MITM, replay attack does not intrude connection between sender and receiver, rather the attacker captures packet and furthermore retransmits packets in replay attacks. These variations of the active eavesdropping attacks are discussed as follows.

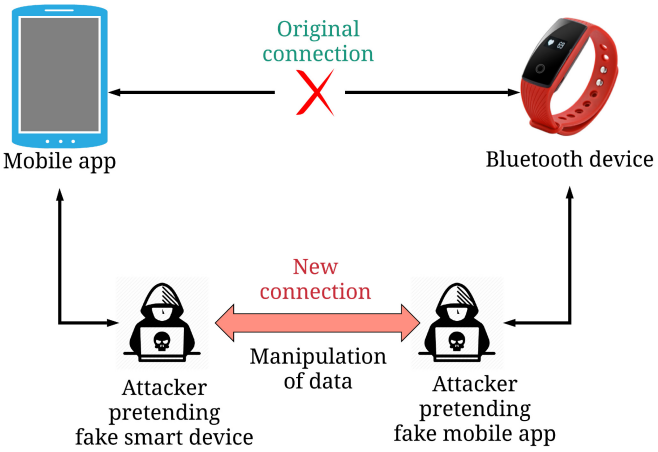

FIGURE 7. Man-In-The-Middle attack.

1) MAN-IN-THE-MIDDLE ATTACK

It is a very common form of threat for wireless communication, and there have been lots of works on the techniques of MITM attacks in IoT devices [100] for both classic Bluetooth [101], [102] and BLE [96]. By positioning oneself in the center of the BLE central and peripheral devices, an attacker can simply conduct an MITM attack on them. The attacker intercepts a packet sent by one of the devices, modifies it, and then sends it to the other device (Fig. 7). Both devices are unaware that their data is being intercepted and perhaps altered by a rogue device.

Attacking Procedure and Severity: In a BLE communication, the master (usually the powerful device, such as a phone) and the peripheral devices communicate over GATT protocol (explained in Section IV-A3). In this scenario, the peripheral device broadcasts signal, and the master connects with it. But the problem is that the GATT protocol's internal mechanism, i.e., *Characteristics, and Services* can easily be cloned and spoofed. With a little trick, the attacker can impersonate the BLE peripheral device, and then the master connects with this malicious device. Now the malicious device can propagate connection with the original device and perform an active MITM attack as illustrated in Fig. 7.

If Just works (Section IV-C1) pairing mechanism is used during bonding, then all versions of BLE using legacy connections (Section IV-B1) or secure connections (Section IV-B2) are susceptible to this attack. Most devices such as medical sensors, keyboards, headphones that do not have I/O capabilities, use the Just works pairing mechanism and are vulnerable to MITM attack [103]. Even if the peripheral devices have I/O capabilities an attacker can force Just works mode, by placing him between these two connecting devices and falsely advertising its input/output capabilities [104]. So, then the master trusts this malicious device and agrees to use Just works pairing method [96].

Passkey entry and Numeric comparison (Section IV-C4) are considered safe against MITM attacks but if the attacker is nearby, and can sniff data transmitted through Near Field

Communication (NFC), then he can still perform MITM attacks successfully. Many IoT devices available in the market are vulnerable to MITM attacks, and so data integrity and confidentiality are compromised. Recent studies [105] found that most of the industrial IoT automation tools use BLE devices with Just works and no further security measures. This makes it easy for the attacker to clone the device, and attackers can have unauthorized access to the peripheral devices. Various smart wearable devices communication over classic Bluetooth are also susceptible to MITM attack [106]. By performing the MITM attack, the attacker can potentially hack smart locks [14], smart home devices [17], smart wearables [96], [103] etc.

Mitigation of MITM: First of all, BLE device manufacturers should properly implement the bonding and encryption standards of the BLE protocol. It is also highly recommended not to use the Just works (Section IV-C1) pairing method. The developers are also encouraged to use secure connections as they provide much better cryptographic protection compared to legacy connections.

Another recommendation is if the central device (i.e., master) knows that the corresponding peripheral (e.g., smart band or the lock) has I/O capabilities, then during the pairing process, the MITM flag must be specified, thereby ensuring secure pairing methods such that *Numeric comparison, Out of bound or Passkey entry* is used. These pairing protocols require different I/O capabilities as discussed in Section IV-C.

There are many open-source tools, such as GATTacker (Section VII-E), BTLEJuice (Section VII-F) that can help IoT and wearable device commercialization companies to perform basic testing, and help to analyze and find BLE vulnerabilities on their products.

2) REPLAY ATTACKS

Replay attack is a common form of attack for wireless communications where the attacker captures legit communication packets and then re-transmits those packets later [107] with malicious intend (shown in Fig. 8).

Attacking Procedure & Severity: The replay attack is particularly dangerous because the attacker can easily sniff the communication of BLE central and peripheral devices. After intercepting the packets, the attacker does not always need to necessarily decrypt the packets to carry out this attack. An encrypted message contains data and encrypted keys. The attacker can simply retransmit the whole intercepted packet, and if necessary protections are not taken, then he can do lots of damage to the systems [17] by exploiting this vulnerability. Unlocking smart locks [76], [78], sending fake notifications [80] can be performed exploiting this vulnerability.

Some open-source software and tools, such as BtleJuice (Section VII-F) and Gattacker (Section VII-E) can be used to perform the replay attack for hacking health monitoring devices [16] or to unlock mobile phones, play music or take pictures, etc. A Ubertooth and a few lines of python

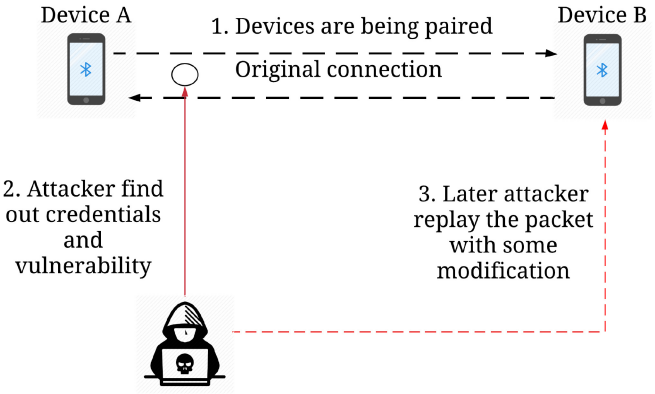

FIGURE 8. Replay attack.

code is enough to perform replay attack on BLE devices that uses Just works pairing method [83].

Mitigation: There are certain recommendations to stop or mitigate this attack, for example, encryption should be implemented properly. Better security framework between GATT and application layer of BLE protocol can prevent the severity of replay attack [99]. Additionally, authentication test mechanism [108], such as nonce or random session key, timestamp, a password for each transaction should be used with every packet transmitted between master and peripheral devices. It ensures that even if the message is captured and retransmitted by the attacker, the session key has already been expired, and resending it no longer works. If this is implemented, then the attacker will need to decrypt the captured packets to change the encrypted key or nonce part of the packet correctly.

C. DEVICE CLONING

In device cloning attacks, the perpetrator makes a fool of the victim by pretending itself as a device that is trusted by the victim. If the victim mistakenly connects, attackers can actively steal victim's data as well as cause massive damage to the victim's devices. Two types of device cloning attacks (e.g., MAC spoofing and Forced repairing) are discussed here. The attacker steals the MAC and GATT characteristics of a device and makes a clone in the MAC spoofing attack, whereas uses this idea to forcefully repair to this cloned device is a forced repairing attack.

1) MAC SPOOFING

MAC Spoofing [109] is a common form of attack where the attacker changes its MAC address (Section IV-A1) and pretend to be someone else (Fig. 9), a legit user or device. Just spoofing the MAC address is not a severe threat but this can be exploited to carry out other destructive attacks, such as Authentication attack (Section V-D2), Denial of service (DoS) attack, etc. that can hamper the integrity and availability of the system [110].

Attacking Procedure of MAC Spoofing: Usually, the BLE companion mobile application looks for the advertisement

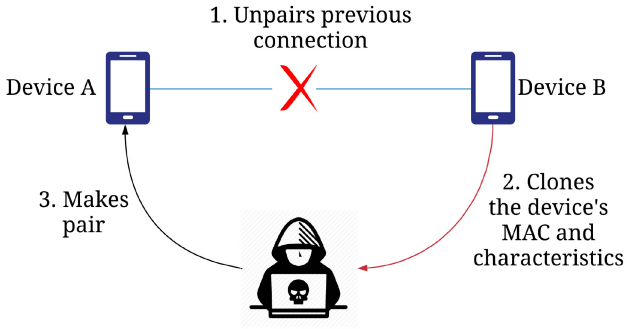

FIGURE 9. MAC spoofing attack.

packets of the peripheral devices and potentially be vulnerable to evil twin attack [111]. The attacker spoofs the MAC address as well as GATT services and pretends to be the peripheral device by cloning the real devices GATT services using software, such as Gattacker (Section VII-E). Then the mobile application tries to communicate with the fraudulent device, and this could lead to lots of malicious attacks [112]. The attacker can also perform this attack during the link key generation, which eventually leads to a MITM attack. Even a false data injection attack can be performed exploiting this attack [112].

Mitigation of MAC Spoofing: The use of RPA described in (Section IV-A1) can lessen the MAC spoofing attack. The idea of using the whitelist and blacklist (details in DoS mitigation (Section V-E4)) proposed by Uher *et al.* [13] can be compelling because the same RPA (Section IV-A1) is not used twice and once an RPA is used, it is blacklisted. Additionally a monitoring system can be designed to detect anomalies in cyber and physical features of the advertising packets generated by an attacker [113].

2) FORCED RE-PAIRING

BLE devices go through the pairing, bonding, i.e., LTK generation when they connect for the first time. BLE specification [2] provides mechanisms for the connected devices to forget the link key. In this attack, the attacker tricks the paired devices to unpair and go through the connection establishment, i.e., pairing phase again [114].

Attacking Procedure: If two devices can successfully bond and generate LTK then they use that key in authentication and data communication [68]. The attacker can't crack the encryption until they get disconnected. In this case, the attacker sends LL_Reject_Ind signal to the victim device [11]. A slave device sends LL_Reject_Ind signal to tell the master device that the previously saved LTK is lost or no longer valid. Thus attacker forcefully compels the victim device to go through the vulnerable bonding process with the original device or with the attacker [105].

Unpairing two connected devices itself is not a very malicious act. But after successfully carrying out this attack, the perpetrator can perform other serious attacks such as eavesdrop passively or even actively perform MITM attack.

Mitigation: The prevention of MAC spoofing attack (Section V-C1) can reduce its severity radically. It is highly recommended to use LE secure connections (Section IV-B2) and avoid Just works pairing. Users should keep an eye on the paired list to detect any dubious activity [62].

D. CRYPTOGRAPHIC VULNERABILITY

Here, we discuss the cryptographic weakness and flawed key exchange mechanism in the BLE protocol. In these attacks, the attackers try to break the encryption and thus the confidentiality and authenticity of the communication. There are mainly two variations of cryptography related attacks, Offline PIN cracking and Authentication attack. They differ in the periods of breaking encryption. In an authentication attack, the perpetrator monitors the connection process and steals shared keys, wherein offline pin cracking attack perpetrator captures data during connection, later breaks cryptography with the help of some tools (Section VII).

1) OFFLINE PIN CRACKING ATTACK

All BLE communications use link-layer encryption, i.e., data is encrypted, and then it is sent through the air. As we discussed in the eavesdropping attack, this radio transmission can be captured by the attacker, but the attacker needs the key or PIN that was used to encrypt the data. Cracking this PIN offline is a very common form of attack for BLE.

Attacking Procedure: PIN Cracking attack can be done in many ways, such as using brute force to crack the PIN [12], which can be performed online or offline. An offline brute-force PIN cracking attack is much more dangerous, due to unrestricted attempts by the adversary to decipher the PIN to decrypt sensitive information. One variant of this attack is to use a dictionary, also known as ciphertext, and this can help to achieve that result considerably faster.

There are some well-known tools to crack TK and LTK from the captured packets (described in Section VII). These tools sniff all the transmitted packets, and the attacker can apply a filter to identify the crucial packets. Though data is encrypted, a tool named Crackle (Section VII-G) can very quickly brute force the TK most of the time. With this TK, Crackle can identify which STK and LTK were used to encrypt the session. If a short length TK is used then it can be cracked very easily. Most of the BLE devices used in health care and medical telemetry applications are vulnerable to PIN cracking attacks [81].

Mitigation: The real problem with this attack is the improper implementation of encryption. The root of the problem is the TK is very short and predictable. In Just works pairing method, TK is set to 0 and in Passkey method, the length of TK is only 6 digits long. Variable length TK based on a 2-bit security flag can solve this problem. TK size should be increased in ascending order of security flag value. For a 24 digit TK it will take nearly 600,000,000,000 years to crack the PIN [12].

2) DEVICE AUTHENTICATION (BONDING) ATTACK

In this attack, the attacker exploits the cryptographic weakness [115] of BLE pairing, i.e., bonding process. During the bonding of two new BLE devices, they choose a pairing method according to their I/O capabilities, exchange their public keys, authenticate the connection, and then finally generate the LTK. The attacker passively observes the key exchanging and connection authentication process and tries to recalculate the shared key for himself.

Attacking Procedure: Older BLE devices that use legacy connections are especially vulnerable to this attack, as they use cryptographically flawed [12] key exchange mechanism. Especially Just works and Passkey key pairing methods use very predictable TK, which can be exploited by the passive observer to crack LTK. Cryptographic properties of the Bit Commitment protocols in the passkey authentication lacks binding property which grant an attacker to change the committed message freely [115]. This allows the attacker to bypass the passkey authentication.

There are some forms of authentication attacks that crack the shared keys exchanged in the pairing process [105] and they are as follows:

- *Guessing Pairing Key:* The attacker brute forces the six-digit pin key used for authentication. When a master and slave tries to connect, the attacker tries to guess the pin or uses different combinations to identify the exact pin.
- *Eavesdropping Encryption Key:* The Attacker uses Ubertooth (Section VII-A) to read all the key exchange messages and decrypt it. There are two ways an attacker can decrypt the communication [83]. Sometimes Ubertooth data contains pairing information. It then becomes trivial to brute force the TK. Another way is to use Crackle (Section VII-G) to crack the communication.
- *Stealing Link Key From the Device:* There are many BLE devices in the market whose hardware is not secured enough to protect the stored encryption key. From these devices, the attacker can extract the encryption or other keys stored in the device.
- *Low Entropy Key Negotiation:* Lowering entropy key or downgrade attack [112] means degrading the size of the link key. According to BLE specification, this size will be 7-16 bytes. In Phase I (Section IV-B1) of the BLE pairing process, both devices exchange their expected link key size, and the generated link key size will be the minimum of them. In this attack perpetrator modifies the pairing request and response in the first pairing phase and changes the link key size to 7 byte [104] so that attacker can easily brute-force the 7-byte link key.

Mitigation: BLE has mechanisms to prevent brute-forcing attacks by increasing the waiting time for the attacker device exponentially when multiple authentication attempts fail. It is also recommended to use a random BD_ADDR address to mitigate this attack.

The solution for the PIN cracking attack (Section V-D1) can also mitigate the authentication attack [12]. But if anyhow the attacker knows the PIN and other shared keys this solution will not work. Zhang *et al.* [99] proposed forward and backward security, described in their security framework, can mitigate this attack's severity, so that even if STK is leaked, the attacker cannot crack the transmission, because encryption keys are freshly generated. Increasing the minimum link key size is also recommended to circumvent low entropy key negotiation [104].

E. DENIAL OF SERVICE

DoS is also known as the flooding attack. Both Bluetooth and BLE are vulnerable to DoS attacks. In this attack, the primary intention of the attacker is to make the resources of the system unavailable to the intended users. In BLE, the attacker primarily targets the master, so that the slave cannot get proper services in the BLE mesh network. This attack happens in the *Physical and Network layer*. Several varieties of DoS attacks are studied in various literature, such as Battery exhaustion, Denial of sleep (DoSL), and Jamming attacks. There are some methodical differences among these attacks. Jamming attack targets the radio spectrum, battery exhaustion attack drains device battery by forcing heavy computation, DoSL attack makes frequent service requests to cause sleep deprivation respectively. Detailed descriptions are outlined as follows.

1) BATTERY EXHAUSTION ATTACK

The main characteristic feature of BLE is that it requires very little power to operate. BLE is designed in such a way that it remains wake only for a short period and transfer data and then goes back to sleep mode again [66]. Battery exhaustion [116] attack targets this unique feature of BLE by keeping the device awake all the time. Bluetooth piconet is subject to this form of attack. The attacker prevents the devices from entering into low power idle mode and thus drains the battery. This disrupts the users from having continuous access to resources. Martin *et al.* described three variations of battery exhaustion attacks [117] and they are as follows:

- Service request power attacks: The attacker continuously sends authenticating requests and thus keeps the device always awake.
- Benign power attacks: Here, the attacker forces the BLE devices to do very heavy computation. In this attack, the attacker sends valid data which inherently causes battery drain.
- Malignant power attacks: Here, the attacker breaks the security of the operating system and modifies the permission of some executables. In this way, the attack not only causes battery drain but also injects the Trojan horse virus.

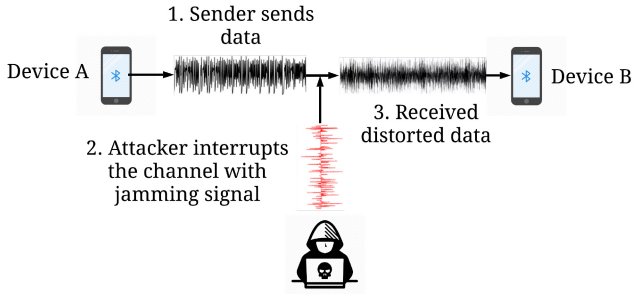

FIGURE 10. Jamming attack.

2) DENIAL OF SLEEP

BLE uses universal wireless sensor network (WSN) [118], [119] which is prune to the DoSL attack [13]. Sensor nodes work in Zero Interaction Authentication (ZIA) mode, which allows the BLE devices to connect to a malicious device without proper authentication. This lack of proper authentication can lead an attacker to make multiple fast connections, causing severe power drain. According to the nature of the attack, we can define three types of threat model [120] as follows:

- *Level 1:* This attack happens when the attacker is not connected to the mesh network but continuously makes connection and disconnection requests.
- *Level 2:* When the attacker gets access to the mesh network, they can conduct this type of attack. Here, the attacker frequently makes an invalid service request, making the network busy serving invalid requests.
- *Level 3:* The attacker frequently makes a valid request and makes the network busy. These types of attacks are challenging to identify, as the requests are legitimate.

3) DOS USING JAMMING

This type of attack occurs in the physical layer. The perpetrator sends an unnecessary signal through the communication channel and creates radio noise between the connected devices [121] as illustrated in Fig. 10. Jamming can be either wideband or pulse band [122]. Different variations of Jamming attacks are discussed as follows:

- *Constant Jamming:* The attacker sends a jamming frame constantly, thus disrupting the communication channel. This type of attack is very easy to detect, as the jamming signal is constant.
- *Deceptive Jamming:* The attacker sends a jamming signal periodically and achieves his desired goal. As the sending of the frame is periodic, it is difficult to detect the attack.
- *Random Jamming:* It is the combination of the constant and deceptive jamming attack. Here, the attacker sometimes sends a jamming frame continuously and sometimes periodically. This type of attack is also tough to detect, as it is entirely random and depends on the attacker.
- *Reactive Jamming:* The attacker marks a channel and any communication through that channel triggers the attacker to send the jamming frame. This attack is very energy efficient as it marks a specific channel and only attacks when communication is made through it. This attack is very hard to detect, as the attacker can randomly switch among channels.

4) MITIGATION

There are different types of DoS attack and there are also different recommendations to mitigate them.

- *Mitigation Using RPA:* As discussed in Section (Section IV-A1), RPA can be useful for BLE devices to prevent DoS attacks from random devices. If the RPA cannot be resolved or the address is random, then the received packet can be discarded. The whitelist and blacklist policy can cut a huge deal in mitigating DoS attacks [13]. The whitelist contains all types of valid addresses for advertisers, scanners, and initiators, and if the attacker uses any random address, then those addresses would be discarded. On the other hand, blacklist policy proposes that an RPA should be used once, previously used RPAs should be blacklisted. If the same RPA is used again, then the device will remain asleep, and the connection will be neglected.
- *Mitigation of Battery Exhaustion Attack:* Martin *et al.* proposed a power-secure architecture [117] for battery-powered devices and this can be used in BLE too. This architecture has two primary features: the multi-layer authentication, which prevents energy waste from service request attacks, and the energy signature monitor to catch the energy-craving intruder.
- *Mitigation of DoSL Attack:* As described earlier, DoSL attacks can be quite harmful to the BLE mesh network. A few defense mechanisms to mitigate this threat are discussed as follows.
  - *Limit Number of Requests:* BLE mesh network should use a counter and timer to calculate the connection or disconnection frequency [120]. If the attacker keeps on sending a lot of authentication requests and if this frequency value crosses a predefined threshold, then that connection will be marked as malicious.
  - *Use Multiple Node:* Mitigation of level 3 attack is hard because requests are valid. Using a counter with every node [120] in the BLE mesh network will be appropriate. When a request is made, the counter increases. If it reaches a predefined threshold, then, it is recommended to force the connection to switch to a new node, so that battery draining is prevented.

F. DISTORTION

Here, the attacker exploits the vulnerability of BLE protocol services (GATT, L2CAP) or BLE data packets and tries to

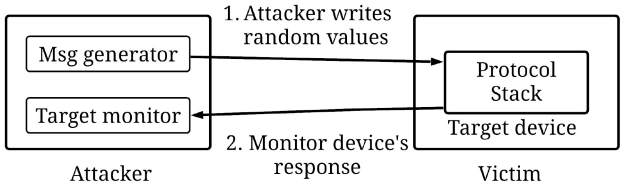

FIGURE 11. Fuzzing attack.

disrupt the services of the BLE devices. Fuzzing and Blue-smack attacks are of this category. Fuzzing attack targets GATT characteristics or data packets, where the Blue-smack attack intends to modify L2CAP. Detailed procedures are discussed here.

1) FUZZING ATTACK

In the fuzzing attack, the attacker uses a program that sends improper random data or carefully crafted malformed data to the device, and this might cause the BLE device to crash or misbehave [105]. The Fuzzer tests application's input handling capability [59]. The attacker writes an invalid value in the BLE GATT server's characteristics (shown in Fig. 11) and observes how the device reacts. This type of attack can even crush a program by buffer overflowing and thus causing a DoS.

Attacking Procedure and Severity: For the Fuzzing attack, the attacked device needs to be a GATT server. The attacker first learns the GATT services and characteristics of a device by an active scan. Now, the attacker knows all the public characteristics of the device and whether they are writable. The attacker then writes some non-standard values in this atomic filed. For some type of fuzzing attacks, the attacker needs to know how much data BLE commands need. To prevent this, BLE uses variable argument size. But the attacker can even randomly generate a variable-length argument to overcome this.

Anthony Rose and Ben Ramsey demonstrated [78] these exploits by unlocking real smart locks. They used Ubertooth to sniff the packets of the lock and its server. Then they overwrote the packet, inserted some random values, and sent that packet back to the lock, which unlocked the smart lock. Apala Ray and his team [105] studied this threat in real devices and they were successful 44 out of 50 test sets.

Mitigation: There is no perfect solution to fix this exploitation. But the severity of the Fuzzing attacks can be mitigated significantly if proper measures taken by the device manufacturers. The developers should not trust user input, i.e., the packets received and should run proper validation checks on them. The BLE module should contempt all the packets, which are not formatted according to the BLE specification [59]. But it does not fully alleviate this attack, because the random packet can sometimes contain malicious data in the correct format.

2) BLUE SMACK

Both Bluetooth classic and BLE use L2CAP for data transmission services. In this attack, the attacker targets L2CAP protocol and disrupts services. This is also known as the ping of death attack.

Attacking Procedure: There are many ways this attack can be executed. L2CAP ping packet size is limited. If the attacker can send specifically crafted malformed or oversized packets to the L2CAP layer, this might cause the crash of the device [62]. The attacker can use the Bluez package that comes with official Linux to carry out this attack [123].

Mitigation: Device Manufacturers should follow the specification carefully while implementing the L2CAP layer in the BLE devices.

G. SURVEILLANCE

A BLE device must protect the privacy of its users. But due to some architectural design issues of the protocol and lack of proper security enforcement, attackers can steal a person's identity as well as private data. Here we have discussed four types of surveillance threats of BLE: Device fingerprinting, Activity detection, Blue-printing, and Blue-stumbling. Activity detection attack intends to identify a person's day to day activity, whereas the other three attacks intend to identify the LE device. But these three attacks use different technique to distinguish a device: Fingerprinting attack analyzes its companion mobile app, Blue-printing attack uses broadcasted MAC address, Blue-stumbling attack utilize advertisement packets respectively.

1) DEVICE FINGERPRINTING

Device fingerprinting is a technique of identifying a device uniquely using different device-specific features, such as MAC Address, UUID, advertisement packets, GATT services, etc. It leads to the violation of the privacy of the users.

Attacking Procedure: Many of BLE peripheral devices have static MAC address which can be used to track the devices. It is found that many BLE IoT devices are vulnerable to fingerprinting by analyzing IoT mobile apps and sniffed static UUID from the advertisement [124]. The detailed attacking procedure illustrated in Fig. 12 and is described as follows.

- UUID has a hierarchical structure and a value set analysis [125] is necessary to identify the UUID hierarchies for fingerprinting, and at the same time, it identifies the app level vulnerabilities, such as the improper usage of cryptography.
- The UUIDs hierarchy found in the first step and sniffed advertisement UUIDs lead to fingerprinting IoT devices. But multiple apps may use the same UUID. So, it is necessary to connect with the device.
- Value-set analysis identifies an app-level vulnerability, so it leads to the discovery of the devices which are vulnerable to sniffing or may be accessed without authorization.

There is another way to carry out this attack, where an attacker can even bypass BLE's anti-tracking mechanism [126], such as device MAC randomization, and still

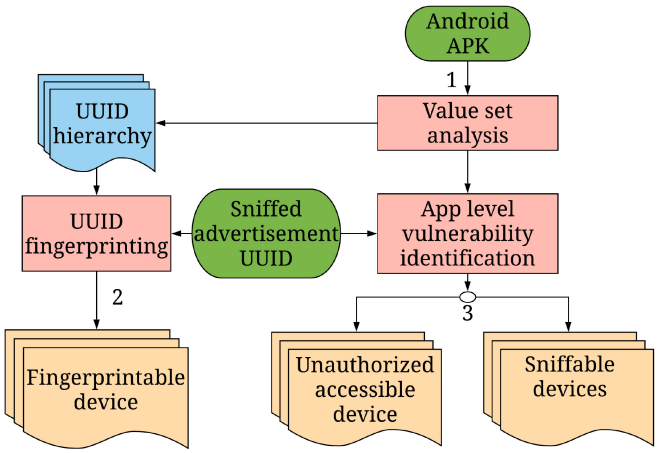

FIGURE 12. Device fingerprinting attack.

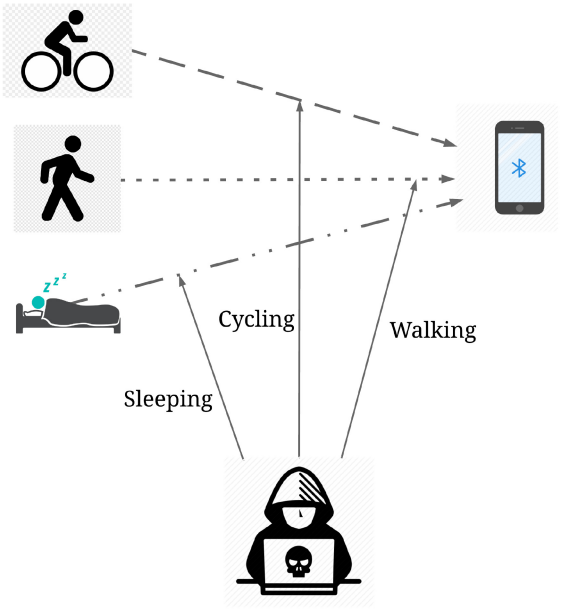

FIGURE 13. Surveillance attack: User's activity detection.

track the BLE device by exploiting BLE's GATT profile. As the information of the GATT profile is publicly available and it contains many of the device-specific information, an attacker can easily collect this information to track the device.

Mitigation: App-level vulnerability needs to be resolved to mitigate this attack. The developer should implement a secure cryptographic function and should hide authentication credentials. Its solution requires three dimensions [124]. First of all, app-level protection is needed. Developers should not use hardcoded UUID, instead, they should use encryption to hide UUID. Secondly, channel level countermeasures should be ensured by implementing disrupt signals broadcast. As a result, the attacker can only sniff disrupted signals. Another protection can be done from the protocol level. When devices connect for the first time, a default UUID is used, but after this, a new dynamic UUID should be generated in the companion app.

In order to mitigate GATT tracking it is recommended to restrict the GATT profile's default permission to the authenticated users only [126]. So that unauthenticated users will not be able to read the value of ATT characteristics. Another recommendation is to make GATT profile inaccessible to unauthenticated users.

2) ACTIVITY DETECTION AND USER TRACKING

There are a lot of concerns regarding privacy for IoTs, especially for wearable devices [127]. There are certain cases where keeping track of a user might be very helpful. For example, there are some diseases such as Alzheimer, where the patient needs continuous monitoring. Detecting motion and user activity of these patients can be helpful [128]. But tracking a user without his consent is a serious privacy violation.

Attacking Procedure: An attacker can get private information by passively observing the BLE smart wearable device and smartphone communication [85]. Some of the issues regarding privacy are discussed as follows.

- *User Tracking:* Detecting the presence of a person without his permission is a privacy violation [85]. The Attacker can detect a person mainly in two ways, one is, static fixed UUID or MAC address, and another is public GATT services. An attacker can look for UUIDs or spoof the GATT profile and pretend to be the peripheral device, and if the companion app tries to connect with it then the attacker knows that the user is in close proximity. Thus the attacker can detect a user in a crowded place or find out if the user is inside the home. Korolova and Sharma [129] studied the feasibility of cross-app tracking of the user using nearby BLE devices and raised concerns and awareness to prevent this.
- *User Activity Detection:* Attackers can detect user's current activity, such as walking, running, sitting by observing traffic between fitness devices and smartphones as there is a strong correlation between user activity and traffic data [85] (Fig. 13). Almost in 89% cases, it is possible to identify a particular person in a group of five people. It is also possible to determine the heart rate sensor data in some BLE devices by observing Received Signal Strength Indicator (RSSI) variations [130].

Mitigation: It is a very broad issue and there are many studies on this threat. It is highly recommended to use a random MAC address. Recent studies [85] show that on most of the fitness devices even from top manufacturers, do not implement them. Address randomization does not provide complete mitigation as users can still be tracked by the GATT profiles.

Most of the other solutions require a change in BLE specifications. An architectural change in the protocol with

a three-way handshake and advertisement confidentiality is needed to prevent device tracking [131]. But this will not provide user control over tracking. Fawaz *et al.* [132] presented a device-agnostic system, named BLE-Guardian, which protects the privacy of users. It enables the user to control who can discover him, scan and connect to his device. As BLE communication supports a limited perimeter, it is necessary to ensure trusted connections with nearby devices. Cha *et al.* [133] introduced a preference privacy framework named PrivacyBat, which defines specifications for BLE devices to achieve privacy with nearby devices.

3) BLUE PRINTING

Blue Printing is a technique to collect detailed information such as device model, manufacturer, unique identifier (IMEI), software versions etc. [61], [134] about the device. Both classic Bluetooth and BLE is vulnerable to this attack. This attack itself might not do much harm of its own, but this can be used to plan further attacks on the victim devices.

Attacking Procedure and Severity: It is not a severe attack, but it results in privacy leakage. BLE protocol specifies that BLE peripheral devices should publicly broadcast its GATT services from where the attacker can collect this information. Attackers can collect statistics on how many devices are deployed by a particular manufacturer. This attack can be critical if there is a well-known security issue for a particular device.

There are many open-source tools available that can be used to carry out this attack. For example, an open-source tool called BluePrint. Reference [61] can gather information on Bluetooth stack. Another handy tool is nRF Connect for mobile available in Google play store which can be used to carry out this attack very easily (Fig 14).

Mitigation: Due to the limitation of the BLE protocol, there is not a good solution to this problem. One recommendation is that devices should make device-specific GATT services available to only authenticated users [126]. The BLE-Guardian [132], mentioned in the activity detection's mitigation can overcome this problem to a great extent without changing the BLE protocol stack.

4) BLUE STUMBLING

It is the process of searching for a device that has known security vulnerabilities.

Attacking Procedure: It is not an active attack, rather an initialization process of other serious attacks, that is going to be exploited later. The attacker sniffs vulnerable devices from a crowded place, so that attacker remains unidentified. Devices with security flaws become the attackers prey.

Mitigation: Devices should be kept in non-discoverable mode when the BLE connection is not needed so that attackers won't be able to discover it. Devices should share very minimum information with unauthenticated devices [126].

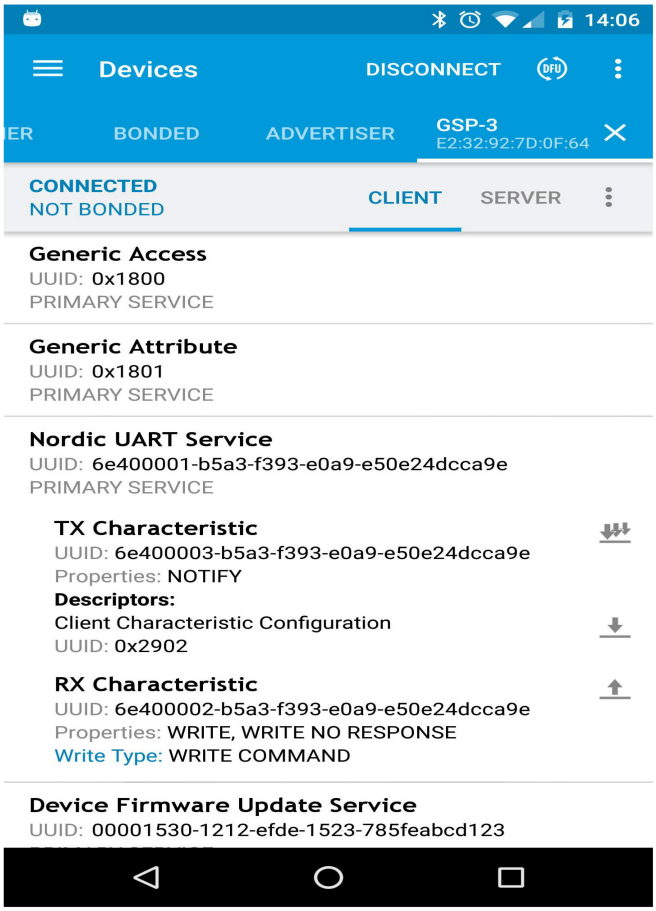

FIGURE 14. nRF Connect [135] Android app.

H. MISCELLANEOUS

There are some threats to BLE devices that are not provoked by BLE protocol but for some other modules, such as a companion app or operating system of the BLE devices. The attackers can exploit these vulnerabilities and steal data from BLE devices.

1) CO-LOCATED MOBILE APPLICATION ATTACK

This threat is not caused by the vulnerability of the BLE protocol itself, rather, the vulnerability is in the companion Android app. The main problem is that BLE bonding credentials which are initiated and authenticated by the authorized mobile application [136], and stored in the Android phone is potentially available to all other applications of the same device [137]. An installed malicious app can potentially misuse the common BLE channel [137], access unauthorized pairing-protected data of the BLE devices by exploiting the pair-bonding keys (LTK) stored in the common BLE module of the mobile phone.

About 70% of medical devices having BLE enabled [136], is prune to co-located attacks. The malicious application can monitor heart rate, blood pressure, or glucose of the user. The attacker can even overwrite any data in the BLE device and cause unexpected behavior.

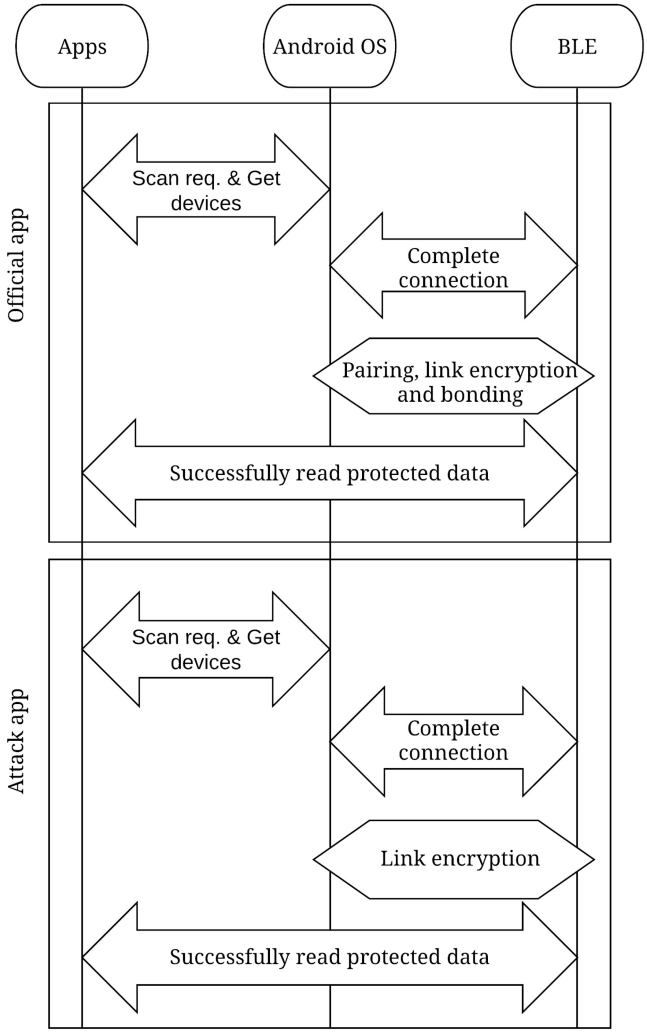

FIGURE 15. Accessing pair protected data of BLE devices by Co-located apps [136].

Attacking Procedure: Sivakumaran and Blasco [136] studied this threat at length and demonstrated that 45% out of 18,900 Android apps were vulnerable and could access protected pairing data with surprisingly fewer permissions.

- *Attack 1 (Accessing Global Pairing Credentials):* The trusted, official companion app makes a pairing request and completes authentication successfully and the Android device stores the link key in the common BLE module. But when another app from that same device tries to connect with the same peripheral device, the Android OS allows this unofficial app to use the LTK generated by the official app (Illustrated in Fig. 15). As a result, this co-located app can communicate with the peripheral devices bypassing the authentication process.
- *Attack 2 (Reuse of Permission):* If an app has permission to use Bluetooth then it can connect to any BLE peripheral devices. If the trusted application is in communication with the peripheral device, the malicious app can directly connect to the peripheral and read-write to the characteristics of it (Illustrated in Fig. 16). The main concern of this attack is multiple applications can concurrently use the same peripheral.

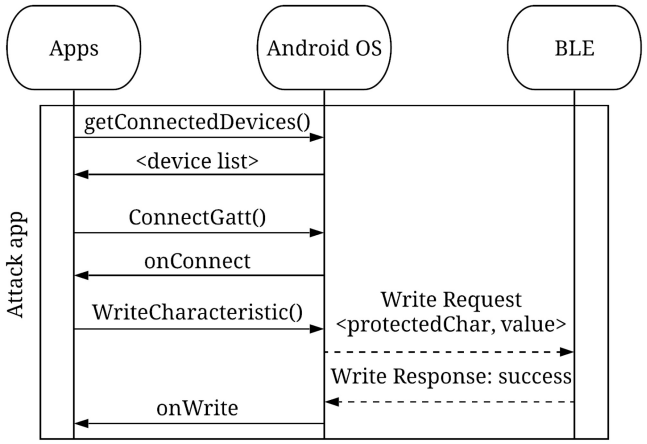

FIGURE 16. Co-location attack 2: Reusing existing connection to read sensitive data [136].

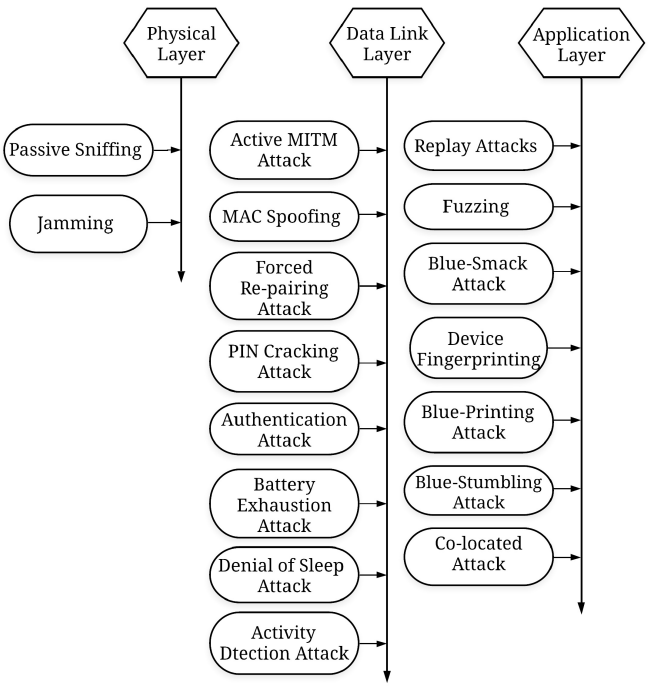

FIGURE 17. BLE threats classification on the basis of OSI layers.

Mitigation: BLE specification and Android should have provided more data protection from such threats. When an application requests pairing with a Bluetooth device, the Android platform should create a bonding policy for that connection [137]. Again proper implementation of end to end security and app-level authentication from the developer side can make it difficult for the imposter to read or write BLE characteristics and thus mitigate this attack.

I. SUMMARY OF BLE ATTACKS

In Table 3, we summarize all the attacks, their severity, their impact on the system's confidentiality, integrity, and

TABLE 3. Summary of all the attacks discussed above and their severity.

Attack	Threat To			Severity	Applicable to C. Bluetooth	Applicable BLE Versions	
	Confidentiality	Integrity	Availability			Legacy connection	Secure connection
Passive sniffing	Yes	No	No	High	Yes	Just works. Passkey: If attacker is active during pairing process.	Just works: It provides more protection compared to legacy Just works methods.
Man-In-The-Middle	Yes	Yes	No	High	Yes	Just works in all versions.	
Replay	No	No	Yes	Medium	Yes	All but depends on successful eavesdropping.	
MAC Spoofing	No	Yes	No	Low	Yes	Versions before 4.2.	
Forced Re-pairing	No	No	No	Low	Yes	Versions before 4.2.	
Offline PIN Cracking	Yes	Yes	No	Medium	Yes	Just works. Passkey: If not implemented correctly.	Just works: Relatively better encryption.
Authentication attack	Yes	Yes	No	Medium	No	All: Depends on implementations.	
Denial of Service	No	No	Yes	High	Yes	All	
Fuzzing	No	No	Yes	Medium	Yes	All	
Blue Smack	No	No	No	Low	Yes	All	
Device Fingerprinting	No	No	No	Medium	No	All: Depends on implementations.	
Activity Detection	No	No	No	Medium	No	All: Application Vulnerability.	
Blue Printing	No	No	No	Low	Yes	All	
Blue Stumbling	No	No	Yes	Low	Yes	All	
Co-Located attack	Yes	Yes	No	High	No	All: Device vulnerability.	

availability. We categorize the severity of these attacks into high, medium, and low based on their impact and recent occurrence. At the same time, we also point out the BLE versions and pairing mechanism that are exposed to these exploits. This provides a comprehensive overview of BLE security threats. Some of these attacks are also applicable to classic Bluetooth. But the attack technique is different from BLE as classic Bluetooth has secured pairing mechanism and better protection to the shared keys.

As we can see, there are various types of vulnerabilities that can be found in IoT and wearable devices that use BLE. But a big number of issues are due to the poor design of the protocol. Table 4 focuses on the *flawed design* of BLE security architecture, cryptographically vulnerable pairing methods, privacy issues of BLE, specially earlier released versions (BLE v4.0 and v4.1) by National Institute of Standard and Technology (NIST) [93] and other security researchers. The basic difference between Table 3 and Table 4 is, the first one summarizes all the threats applicable to BLE, and the second one identifies which architectural flaws are the reasons for these threats.

To have a better understanding of the vulnerabilities of BLE, we summarize all the threats of BLE on three OSI layers (e.g., Physical, Data-link, and Application) as illustrated in Fig. 17. The attacks that take place in the radio spectrum are assorted as a physical layer attack. From Fig. 17, it is clear that most of the attacks happen in Bluetooth data link-layer. BLE data transmission occurs between the data link-layer of two devices. In most attacks, the attacker captures the link-layer packet, that's why link-layer security is the primary concern of BLE developers. Vulnerabilities that are caused by improper implementation of the protocol by the device manufacturers or developers are classified as an application layer vulnerability.

VI. CASE STUDY OF REAL BLE EXPLOITS

Smart wearable devices have enriched our life as well as personal health in many different ways. It can provide us with insights about our health condition, such as blood pressure, heart rate, daily calorie burn, etc. Most of these devices use BLE for communicating with the companion mobile app [65] for sending sensor data such as blood pressure monitoring [65], [138], [139], walking speed, etc. In this section, we present some common vulnerability patterns uncovered by different research teams, that could have been easily avoided if device manufacturers were conscious about security. At the end of this section, we provide a summary of *recommendations* for device manufactures and users to protect themselves from many known vulnerabilities. Some of the recent studies and threats of vulnerabilities in real devices are described as follows.

A. SWEYNTOOTH VULNERABILITIES

Recently a research team from Singapore outlined a series of vulnerabilities in BLE devices, named SweynTooth vulnerabilities [140]. It may have a significant bump on

TABLE 4. Security design issues of different versions of BLE protocol.

No	Vulnerability	Security Standards and Remarks	BLE Versions
1	BLE Just works pairing method provides no MITM protection.	BLE devices should be paired in a private and secure environment to minimize eavesdropping and MITM risks [93]. An attacker can successfully capture BLE communication packets and modify data exchanged between trusted devices. It is highly recommended not to use Just works pairing methods.	4.0 4.1 4.2
2	Device privacy with static fixed public address.	If an attacker can capture BLE address for a particular user, then the user's privacy can be compromised. For BLE, the developer should implement randomized address for privacy [93]. RPA should be used to protect privacy.	4.0 4.1
3	There are no passive eavesdropping protection in BLE legacy connection.	The attacker i.e., eavesdroppers can capture secret keys (IRK, LTK, CSRK) successfully sniff BLE communication. Secure pairing options such as Numeric comparison, Out of bound, Passkey entry should be used [93].	4.0 4.1
4	BLE legacy connection Mode 1 Level 1 does not require any security mechanisms, such as encryption or authentication.	Security Mode 1 is insecure in both classic Bluetooth (BR/EDR) and BLE. BLE secure connections such as security Mode 1 Level 4 (ECDH encryption and authenticated pairing) is recommended. [93]	4.0 4.1 4.2
5	There are no user authentication.	The BLE smart specification only provides device authentication. There is no application-level security, such as user authentication. Application developers can implement this via application logic [93].	All
6	Link keys can be stored improperly.	The attacker can modify and read data unless keys are stored securely and protected via access controls [93].	All
7	Lack of security services	The protocol should have good auditing, non-repudiation services. Developers have to implement these in the application layer [93].	All
8	Always discoverable devices are vulnerable to privacy and security attacks	BLE devices will have to discoverable to be connected. But then attackers can fingerprint and perform other attacks. After bonding, devices should be put to unrecoverable [93].	All
9	Vulnerable key exchange protocol	In legacy connections, the BLE protocol has used insecure custom key exchange protocol. Encryption used in Just works and Passkey pairing methods in legacy connections are vulnerable. Newer version devices are encouraged to use secure connections which ensure the Elliptic Curve Diffie Hellman public-key cryptography [93].	4.0 4.1

TABLE 5. SweynTooth vulnerability types and affected manufacturers [140].

Type	Vulnerability Name	Affected manufacturers
Crash	Link Layer Length Overflow	Cypress,NXP
	Truncated L2CAP	Dialog Semiconductors
	Silent Length Overflow	Dialog Semiconductors
	Public Key Crash	Texas Instruments
	Invalid L2CAP	Fragment Microchip
	Key Size Overflow	Telink Semicondu
Deadlock	LLID Deadlock	Cypress,NXP
	Sequential ATT Deadlock	STMicroelectronics
	Invalid Connection Request	Texas Instruments
Security Bypass	Zero LTK Installation	Telink Semiconductor

the BLE SDK (Software Development Kit). Many fitness trackers, smart locks, smart plugs, medical devices, smart alarm, smart home systems, and various other wearable devices have these vulnerabilities. BLE SDKs are provided by vendors of system-on-a-chip (SoC) chipsets. IoT device makers buy these SoCs to support BLE communications.

Summary of SweynTooth vulnerabilities and affected manufacturers are listed in Table 5. Some attacks can crash devices, may cause devices to go into panic mode, forcing them to be frozen or allow the hacker to take control of the device, ignoring security.

B. VULNERABILITIES IN WEARABLE DEVICES

BLE fitness bands [141] and health devices [142] are collecting a lot of personal data, but there is not any standard for BLE applications on collecting and sharing our data. An Attacker can very easily access a lot of personal data, read the various health sensor data [143] or even guess what the user is typing by analyzing the motion sensors data from smart wearable wrist devices [144].

Arney [145] provided a overview of different types of BLE threats in medical devices. Their survey showed that medical devices were most vulnerable to MITM attack and also pointed out the link layer vulnerability in implementing BLE in medical devices. Hilts *et al.* [146] provided a comprehensive analysis of the security and privacy of various popular fitness trackers where manufacturers did not take proper security measures.

Arias *et al.* [143] discussed the security and privacy concerns of Nike+ Fuelband which is a BLE 4.0 enabled user tracking device that measures different fitness activities such as calories burned, the number of steps walked, etc. The STM32 documentation ensures that the microprocessor can lock external reads-writes against the internal flash. But Nike+ Fuelband does not have this protection and attackers can easily modify the content of this flash [143].

Hale *et al.* [147] introduced SecuWear, a multi-component research platform, for analyzing, testing, and mitigating vulnerabilities in wearable devices. As these wearable devices

have access to a lot of personal data, there have been lots of vulnerabilities found in recent studies [148], [149].

C. BLUEBORNE ATTACK

In 2017 Armis Labs revealed a new attack, named it BlueBorne, as this new threat spread through the air (airborne) and attacked devices through Bluetooth [150]. Blueborne tries to find a hole in network defense and takes complete control of the device. As almost all operating systems provide a high privilege to Bluetooth, it becomes a desirable way to the attacker. Armis disclosed four major threats to the operating system [150] and they are:

- Information leak.
- Remote code execution in Bluetooth Network Encapsulation Protocol (BNEP).
- Remote code execution in the Personal Area Networking (PAN) profile.
- MITM attack.

All users are recommended to update their operating system.

D. HACK SMART BAND

Smart band, e.g., fitness tracker is becoming very popular among people of all ages to track their daily exercise, get a mobile notification while driving, etc. One of these low price smart band is Xioami Mi Band. But security mechanisms of BLE have not been properly implemented in most of these low priced devices. Many security researchers have shown detailed ways to hack such smart bands. For example, this blog [151] post provides a detailed explanation on how to hack a Mi band from a Linux laptop and python script. The script is open-sourced in Github [152]. The attacker only needs to run this python script and then he will be able to send fake calls, message notifications the band. Whenever the mobile phone gets a text message, it sends it to the band. As the BLE secure connections are not implemented properly on these devices, the attacker can also sniff these confidential text messages. Many other smart bands from even top manufactures are also vulnerable to various attacks [146].

E. FINGERPRINTING ATTACK

Zhiqiang Lin discovered a design flaw in low-powered Bluetooth [153] devices. UUID is broadcast publicly by a BLE device, which may lead to privacy issues, such as device fingerprinting attack. Again some communication does not use encryption properly between the mobile app and the BLE device. Lin and his team took a BLE "sniffer" on a tour of the university's 1.28 square-mile campus. They found out that about 5,800 Bluetooth devices were operating, among them, 94% were vulnerable to fingerprinting attacks. Moreover, 7.4% were exposed to unauthorized access or eavesdropping attacks.

F. SMART LOCKS UNLOCKED

Nowadays keyless lock is becoming more and more popular in smart homes, smart bikes, etc. Most of these locks use BLE and the manufacturers usually do not implement the recommended secured bonding or encryption. This leads to a few serious threats, where the attacker can perform MITM or other attacks and can open the locks. Jasek [17] presented these security vulnerabilities on various BLE smart locks in Black Hat USA. They shared their finding that around 80% of BLE locks were susceptible to MITM attacks. They recommended the device manufacturers to implement link-layer encryption, device whitelisting, using secure bonding recommended by the BLE specification.

G. LESSONS LEARNED AND SOME RECOMMENDATIONS

IoT devices are extensively used in industries and medical sectors. Hence, any of its vulnerabilities can have severe consequences. Most of the times, device manufacturers lack the expertise or motivation to provide enough support and guideline to install security patches in to these smart devices. These IoT devices have different underlying OS, firmware, and regulatory permission and so our security software that is used in computers, is not applicable here. The lifespan of these devices widely varies from one to ten years. The security concern that was not considered during the market release time (of the product) could have a potentially serious effect now. Due to the unique nature, any organization should do a complete inventory of all the IoT devices connected with its network. Network traffic to and from the IoT devices should be analyzed to check if any device is sending unusual packets, or if the packets are encrypted or not. If manufactures and developers can make the devices more secured, reliable, and less intrusive, BLE devices will have more market acceptance.

In summary, if device manufacturers follow the following basic recommendations, many of these severe issues could have been avoided.

- Always try to use BLE secure connections over legacy connections as they provide much stronger encryption.
- Manufacturers should avoid the Just works pairing method [93].
- Developers should use secure pairing options, such as *Numeric comparison, Out of bound*.
- All transmitted data should be encrypted by the AES-128 algorithm [93].
- Manufacturers should take necessary steps to provide mandatory security updates to already released devices.
- Developers should use the application level authentication test mechanism, such as nonce or random session key etc.
- All encryption keys should be its maximum allowable size [93].
- Devices should only disclose device specific GATT services to the authenticated users.
- Random BD_ADDR address should be used to ensure privacy.
- Unnecessary services and profiles should be disabled [93].

Some of the security recommendations for the users to follow are listed follow.

- The users are recommended to use a secured, private environment to connect with their IoT devices [93].
- Users should turn Bluetooth off when it is not necessary.
- Users must not accept connection from untrusted [93].
- Users should not keep their devices in always discoverable mode [93].
- Do not install suspicious software, patch in mobile or computer that communicate with IoT devices.
- Users should sometimes check the list of paired, i.e., trusted devices and remove old or lost devices.
- Users should be careful if the battery consumption or data transfer speed is unusual.
- Users should try to apply security updates from trusted manufactures as soon as possible.
- Change or hide default configuration settings [93], such as default device name or model number to avoid device fingerprinting.

VII. USEFUL TOOLS

In this section, we describe some of the well known open-source hardware and software tools developed by security researchers to investigate various BLE security vulnerabilities such as passive eavesdropping [154], breaking BLE encryption [155], MITM [156], etc. Security researchers have used these tools to demonstrate security exploitation in popular IoT devices [14]. This list of free and open-source tools will help BLE developers and security researchers to set up a low-cost testing environment, analyze and assess vulnerability, and protect BLE-enabled IoT devices.

A. UBERTOOTH

In wireless technology, data is transferred through air medium and makes it possible to easily eavesdrop or even manipulate these data packets. Michael Ossmann and Dominic Spill developed Ubertooth [154], which is an open-source Bluetooth testing tool. Ubertooth provides the hardware to perform passive monitoring of communication between classic Bluetooth and BLE devices. Ubertooth one was released in January 2011 [157] and it can capture and demodulate signals in the 2.4GHz ISM band with a bandwidth of 1 MHz.

B. ADAFRUIT BLE SNIFFER

It is similar to Ubertooth, and specially designed for capturing BLE packets. It was developed by the Adafruit industry [158]. One can capture the data packets transferred between two BLE devices using this tool and then analyze data using Wireshark.

C. SMARTRF PACKET SNIFFER

It is a PC software to analyze BLE packets. It has two components. One is the SmartRF Sniffer agent that communicates with the captured device [159]. The second one is the firmware that enables CC13xx or CC26xx launchpads to be used as a capture device. It can be configured to listen to a specific communication between two master or slave devices. It can decrypt packets on two modes. The first mode requires a manual supply of LTK. The second mode is automatic, which decodes the encrypted packet and shows payload in GUI. Decryption may fail if it fails to capture the key exchanging packets from the transmission.

D. WIRESHARK

Wireshark is a widely used open-source network protocol analyzer. Many governmental, non-governmental, educational and no profit sectors use this tool to monitor their network activities and finding issues by using effective packet analysis. It can capture data from Ethernet, Bluetooth, USB band, and other communication medium and helps to analyze data. Wireshark comes with GUI which enables the user to visualize the captured packets by applying a filter.

E. GATTACKER

GATTacker is a Node.js package developed by Jasek [17] for performing BLE MITM attacks. Jasek presented it at the Black Hat USA conference in 2016. The software runs on any recent Linux, even in Raspberry Pi, where each module needs a BLE adapter. It enables the attacker to manipulate transmitted data. GATTacker can make a clone of the victim device and connects to mobile application presenting itself as the original device [156]. It also keeps an active connection with the original device. So, it can forward all the data from the device to the mobile app and perform an active MITM attack.

F. BTLEJUICE

Cauquil [14] developed BtleJuice to perform MITM attacks on BLE devices and demonstrated it at the DefCon 24 conference in 2016. BtleJuice composed of 4 components: an interception core, an interception proxy, a dedicated Web interface, and Python, NodeJS bindings [160]. The interception core and proxy are the main components.These two components run on independent machines and internally communicate through the Web Socket protocol. BtleJuice works as a proxy between the BLE peripheral and mobile apps. It captures any commands sent to the peripheral from the mobile app.

G. CRACKLE

Crackle [155] cracks the BLE encryption. It finds the flaw in the pairing mechanism, as a result a passive eavesdropper can decrypt any vulnerable communications [155]. Crackle can guess or very quickly brute force the TK used in the pairing modes (Just works and 6-digit PIN), supported by most devices. With this TK, crackle can derive all further keys used during the encrypted session that follows immediately after pairing. Crackle operates in two modes.

(a) Crack TK: It is the default mode and is used to brute force the TK in the Just works or the Passkey pairing method.

(b) Decrypt with LTK: This mode is user-dependent and lets the user supply an LTK. Crackle uses this LTK to decrypt all communication between master and slave.

H. OTHER TOOLS

Apart from these tools, there are some other open-source tools too, such as SecuWear [147] developed by Hale *et al.* It is a multi-component research platform. It is used for analyzing, testing, and mitigating vulnerabilities in wearable devices. SecuWear consists of five core components: MetaWear works as peripheral, Apache Cordova to prototype mobile apps, Ubertooth, Web service API and Django for showing log of mobile application.

VIII. NEW DOMAINS FOR BLE AND RESEARCH OPPORTUNITIES

Low energy consumption and relatively low latency in data transmission make BLE as one of the remarkable technologies to be used in the IoT sector. Due to the new BLE features [161] (such as low energy audio, IPv6 packets over BLE), BLE has the potential to be used in different domains, such as localization, smart payments, etc. In this Section, we present exciting new features of BLE v5.x, followed by the new application domains for BLE, where these new features and security aspects can be useful. We also provide a comprehensive overview of open issues and future research directions which will help BLE developers and researchers to have a better understanding of the evolution of BLE and provide research guidelines to better secure BLE devices.

A. NEW FEATURES OF BLE

Bluetooth SIG announced releasing a major Version 5.0 on June 16, 2016, and released it at the end of 2016. SIG released Bluetooth Version 5.1 and version 5.2 in the following years, in January 2019, and January 2020 respectively.

1) LATEST FEATURES

It comes with some significant improvements in some of the core fields of wireless technology. Bluetooth 5.0 is backward-compatible of older versions and Some of the features and security improvements are listed follows.

- *Cover Larger Range:* BLE v5.0 has around 4 times greater range than its earlier versions. BLE v4 supports 50-100m coverage outside and 10-20m inside. Where BLE v5 covers 200m range in outdoor and 40-50m indoors. This means it can be a replacement for WiFi.
- *Higher Speed:* BLE v4 can reach up to 1 Mbps where BLE v5 has a maximum 2 Mbps speed limit. This means that BLE v5 can be used for live streaming and can have a higher data transfer rate between the wearable device and mobile app. Devices can be even reconfigured to choose higher speed or long distance cover-up.
- *Long-Lasting Battery:* The design of better signal modulation and advancement in the use of the frequency spectrum, enable BLE v5 to consume less energy which is nearly half than its previous versions.
- *LE Advertising Extensions:* Improvement in the field of sending an advertising packet allows devices to exchange packets without synchronization. This feature enables extensive use of beacons in everywhere.
- *Increase Payload Size:* BLE v4 has little message capabilities. Message size is about 31 bytes having 17-20 bytes of payload. BLE v5.x supports up to 255 bytes of messages.
- *Slot Availability Mask:* This feature enables BLE to coexist with nearby Long term evolution (LTE) transmission. That means that while transmitting it stays away from nearby LTE transmitting channels, thus avoid packet loss between two transmissions.
- *LE Audio:* BLE v5.2 has multiple audio connectivity features that means a mobile device can connect to two audio devices at the same time by enabling the dual audio mode.
- *LE Channel Selection Algorithm:* This feature is added since BLE v5.0. This algorithm provides high throughput in data transmission by avoiding interference and multi-path fading.
- *GATT Caching:* This feature is added in BLE since BLE v5.1. This lessens the time required to discover the GATT database when connected to the previously connected devices by using GATT caching.
- *Security Manager:* The latest version of BLE provides a better SM (Section IV-A3) which makes it harder for the attacker to crack.
- *Support for IoT Devices:* Many IoT devices require less power usage but high speed and throughput for better performance. Since version 5.0, BLE provides different solutions to solve all of these problems, making it compatible with all types of IoT devices.

2) TRANSMITTING IPV6 PACKETS OVER BLE

In the era of IoT, BLE-based health sensors, devices need to communicate with the Internet. Most of the BLE devices communicate with the Internet via connecting with the companion mobile application. Internet Engineering Task Force (IETF) and SIG are working on a complete IP-based communication stack over BLE so that the BLE devices will not need to communicate with the mobile apps to connect to the Internet and will be able to send IPv6 packets over IEEE 802.15.4 (6LoWPAN) [162], [163], [164].

B. NEW APPLICATIONS FOR BLE

BLE is used in a wide range of devices from sensors to mobile phones to medical devices for a wide range of applications from personal health to home automation to industrial automation. IoT devices are going to have an exponential growth With the *advancement of 5G* and new features of BLE. In this Section, we discuss some of the new domains of application for BLE.

1) BEACON TECHNOLOGY

Beacon is a wireless transmitter, which utilizes BLE technology to transmit messages to smart devices using location-based searching [165]. Beacons are mostly used in the shopping malls, where vendor-specific beacons are installed, so that customers get notified about a vendor's products, promos, and offers in that vicinity. Beacons are also used in location-based news services and smart bookshelves [166], which monitor a person's book of interest to suggest a proper book. In the future, the beacon's monitoring system will be a source of big data for commercial companies to identify the user's demand. It can be used in event management to navigate a person through the event and provide event information from time to time. Optimizing its usages [167] has immense possibility to be used in transportation [168] to provide location information [169] and route, car door automation, user tracking in the airport, railway station, restaurant, etc.

2) BLEACH

One of the major drawbacks for BLE developers are that they do not have any control over the BLE radio driver or link layer. It works like a black box doing data transfer commands. Spörk *et al.* [164] designed BLEach which offers a tuning knob for controlling energy usage and timeliness of BLE transmission. BLEach has full support for IPv6 over BLE, and it is an open-source stack. It is lightweight, and compatible with any standard device. It can reduce energy cost by 50%. BLEach provides 3 novel services [164]. First is an adaptive duty-cycling which allows a slave to rapidly adapt to specific changes in traffic load and negotiate with border router for new connection parameters that better suit its application. Second is traffic prioritization and multiplexing, which allows the master device to prioritize a slave device low by giving low credit and high prioritize a slave device by giving higher credit. Thirdly BLEach comes with a physical layer and a BLE link, and it can perform indirect link-quality monitoring.

3) MOBILE PAYMENT

BLE can also be used for mobile payment. In every merchant shop, there are BLE beacons which are used for advertising different products. These BLE beacons can be used for near field payment. David Baldie patented a BLE mobile payment system [170], [171], where the user device will have an application that will identify the UUID broadcast by the beacon at the merchant location. This will initiate a near field payment method between a user and a merchant.

C. FUTURE RESEARCH DIRECTIONS

In this Section, we discuss current challenges and future research opportunities in seven major categories. It provides future research challenges for security researchers to investigate potential security and privacy issues of these new features of BLE and contribute to design a better guideline for IoT device manufacturers.

1) BLE MESH NETWORK

BLE was originally designed as a star topology, but for implementing industrial networks a multi-hop mesh network is necessary. So, there are many research opportunities to create a secured mesh network using short-range BLE technology. As a huge number of devices are participating in this network, there are many security and privacy concerns that need to be investigated [172] at the physical, link, network, and application layers. Some open issues in the BLE mesh network [173] are real-time communication [174], efficient multicast [173], efficient authentication, efficient auto-configuration, and inter-operability. Securing and developing a high-quality mesh networks are open research field for BLE researchers.

2) POTENTIAL NEW SECURITY ISSUES

All the additional features of BLE 5.x (mentioned in Section VIII-A) make BLE attractive for IoT devices. However, this also increases the attack surface. A larger coverage range provides the attackers with a better opportunity to access devices and eavesdrop from a long distance. Due to a high transfer rate, if an attacker can somehow access a person's data, he can copy all the personal data instantly. There is a lack of comprehensive studies on new features and security mechanisms [161] to handle confidentiality, authenticity, and the integrity of BLE 5.x. As more and more devices adopt the latest versions of the BLE protocol, new attack domains might be discovered. Another open research opportunity is to investigate a range of BLE devices from the same manufacturer to test if their entire product line is compromised because device manufacturers usually reuse the same BLE stack in many of their products [83].

3) BETTER ENCRYPTION AND PAIRING

One of the most significant vulnerabilities of BLE protocol is custom key exchange mechanisms that are used in legacy connections. There is always ongoing research to improve the pairing methods and link-layer encryption to ensure privacy and strong encryption for better security. BLE has improved its encryption [161] over its subsequent releases. However, if device manufacturers do not maintain standardized guidelines properly, this encryption can always be cracked [175].

4) BLE PERFORMANCE IMPROVEMENT

Researchers and practitioners, from the academia as well as the industry, are working together on some open research problems to improve the performance of BLE. There are lots of new research opportunities to design and improve the physical layer, i.e., radio or PHY mode that is introduced in BLE v5.x to improve the collision avoidance [176] throughput, range, and speed with low energy consumption.

BLE faces some difficulties in finding devices in a crowded environment. The idea of adaptive parameter settings [177] (i.e., Adv. interval, Adv. time per channel, scan window, scan interval), and use of random back-off to retry

the channel sensing might be attractive research field for better device discovery. More research needs to be performed to incorporate the idea of role switching of central and peripheral based on the event, which will improve the role assignment in BLE. The idea of operating BLE with other wireless technology faces some challenges. Adaptive frequency hopping to avoid interference is effective in most cases which may lead to co-existence with other wireless technology. Incorporating BLE in *5G and VANET* is a very demanding research topic [177].

5) NEW SECURITY FRAMEWORK

Developing a new security framework [133] requires some goals to accomplish. One can aim to develop a framework to ensure secure communication [99] between two BLE devices. Such a framework must have a mechanism to prevent different BLE threats and ensure reliable communication. However, adding an extra layer of security may impose computational or transmission overhead and sometimes may require additional function call from the underlying hardware. Building such a framework that is compatible with all legacy devices, is a challenging task for researchers.

Currently, IoT apps do not have any restrictions on privacy data collection. Therefore, there is a research opportunity for creating a consensus framework on policies for users to agree on what data can IoT devices collect [133], [178]. There can be a monitoring system to check if any unauthorized sensitive data is transmitted between the android app and IoT devices.

6) MACHINE LEARNING IN BLE SECURITY

There are a lot of connected BLE driven IoT devices in industrial and home automation networks. The security mechanisms [161] introduced in BLE 5.x mainly focus on securing communication between two connecting devices. There is a lack of efficient techniques to secure the BLE mesh network. Currently, there is an open research opportunity to protect the BLE mesh network from zero-day vulnerability, DoS attacks, spoofing attacks [110] using intrusion detection systems [172], [179] and intrusion prevention systems, and watchdogs. Introducing new aspects of machine learning algorithms is a promising area of research for enhancing the security and privacy of IoT devices. To identify network intrusion, neural networks (NN) [180], [181] and support vector machines (SVM) [182] might be utilized. Detection of spoofing can be accomplished through the use of a deep neural network (DNN) [183] or a support vector machine (SVM) [184]. Infinite Gaussian mixture models can enhance authentication privacy [185].

In the wrong hands, the power of AI can be exploited. AI is frequently used by attackers to uncover and exploit flaws far faster than developers can repair them. Machine learning can be used to create a clever fuzzing algorithm that generates incorrect data in order to detect device vulnerabilities [186].

7) BLOCKCHAIN IN BLE SECURITY

BLE enabled smart wearable and IoT devices have been seamlessly integrated into our everyday life. So secure data management and robust access control of IoT devices are becoming very important. The idea of using a server to connect with an IoT device rather than connecting with individual device directly may enhance device management and users privacy significantly [178]. This server will store users preferences, thus prevents IoT devices to access personal information. But if the network/server is breached, then every device connected to that network will be compromised. So there is a research opportunity [178] to use decentralized block-chain technology to secure IoT devices connected in a BLE mesh network [187], [188], [189].

IX. CONCLUSION

BLE is one of the most prominent technologies in the IoT sector throughout the world due to its power efficiency and reliable data transfer. It has become an intrinsic part of our daily life and IoT communication. Although BLE consumes very less energy and ensures faster data transmission, some of its vulnerable pairing methods leave BLE communications at risk.

In this survey paper, we have analyzed the BLE security protocol critically and identified the security flaws in device pairing, that lead to many security and privacy issues. Then we have presented a comprehensive taxonomy of different attack vectors, focusing elaborately on different techniques for each threat, and provided recommendations to alleviate these threats. We have presented several recent attack synopses and introduced some popular tools to analyze, demonstrate, and mitigate security vulnerabilities. Finally, we have discussed new features and potential new application domains that will be the driving force for adopting BLE in future IoT devices.

This in-depth study of possible threats and their countermeasures is a comprehensive reference for BLE researchers, developers, and practitioners to gain background information about the BLE security framework and its loopholes, which will encourage them to research more to improve this technology. It will help BLE developers to make a checklist of these threats and guide them in shaping new security architecture.

BLE has gone through a lot of architectural changes since its first release, and it still requires a lot of effort to pinpoint critical security issues in its new features. Recent research on different pairing mechanisms, enhancement of encryption, structured mesh topology, and protecting the BLE network using IDS and block-chain will make BLE more secure and reliable. Less energy consumption, more efficiency, and elevated security will make BLE an attractive solution in the future IoT.

REFERENCES

[1] *Bluetooth Core Specification Version 4.0, Specification of the Bluetooth System*, vol. 1, Bluetooth, Kirkland, WA, USA, 2010, p. 7.

[2] "Specification of the Bluetooth System." 2014. [Online]. Available: https://www.bluetooth.com/specifications/bluetooth-core-specification/ (accessed Jul. 17, 2020).

[3] N. Hunn, "An introduction to Wibree," EZURiO Ltd., London, U.K., Larid Tech., Chesterfield, MO, USA, Rep. WHP-060006-1V0, 2006.

[4] "Bluetooth Special Interest Group." Cryptology ePrint Archive. Bluetooth.com. 1998. [Online]. Available: https://www.bluetooth.com/about-us/ (accessed Jul. 17, 2020).

[5] B. Ray, *Bluetooth vs. Bluetooth Low Energy: What's the Difference*, vol. 2, Bluetooth, Kirkland, WA, USA, Apr. 2015.

[6] P. Kinney, "ZigBee technology: Wireless control that simply works," in *Proc. Commun. Design Conf.*, vol. 2, Oct. 2003, pp. 1–7.

[7] M. Siekkinen, M. Hiienkari, J. K. Nurminen, and J. Nieminen, "How low energy is Bluetooth low energy? Comparative measurements with ZigBee/802.15. 4," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Paris, France, Apr. 2012, pp. 232–237.

[8] C. Gomez and J. Paradells, "Wireless home automation networks: A survey of architectures and technologies," *IEEE Commun. Mag.*, vol. 48, no. 6, pp. 92–101, Jun. 2010.

[9] A. Dementyev, S. Hodges, S. Taylor, and J. Smith, "Power consumption analysis of Bluetooth low energy, ZigBee and ant sensor nodes in a cyclic sleep scenario," in *Proc. IEEE Int. Wireless Symp.*, 2013, pp. 1–4.

[10] A. Peterson, *Yes, Terrorists Could Have Hacked Dick Cheney's Heart*, vol. 21, Washington Post, Washington, DC, USA, 2013.

[11] M. Ryan, "Bluetooth: With low energy comes low security," in *Proc. 7th USENIX Workshop Offensive Technol.*, Washington, DC, USA, Aug. 2013, pp. 4–11.

[12] G. Kwon, J. Kim, J. Noh, and S. Cho, "Bluetooth low energy security vulnerability and improvement method," in *Proc. Int. Conf. Consum. Electron. Asia (ICCE-Asia)*, Seoul, South Korea, Oct. 2016, pp. 1–4.

[13] J. Uher, R. G. Mennecke, and B. S. Farroha, "Denial of sleep attacks in Bluetooth low energy wireless sensor networks," in *Proc. IEEE Military Commun. Conf.*, Baltimore, MD, USA, Nov. 2016, pp. 1231–1236.

[14] D. Cauquil, "BtleJuice: The Bluetooth smart MiTM framework," in *Proc. DEF CON 24 Internet Things Village*, Aug. 2016, pp. 4–7.

[15] T. Melamed, "Hacking Bluetooth low energy based applications," in *Proc. Int. Conf. Internet Monitor. Protect.*, Venice, Italy, Jun. 2017, pp. 1–23.

[16] J. Radcliffe, "Hacking medical devices for fun and insulin: Breaking the human SCADA system," in *Proc. Black Hat Conf.*, Jul./Aug. 2011, pp. 1–13.

[17] S. Jasek, "Gattacking Bluetooth smart devices," in *Proc. Black Hat USA Conf.*, Jul./Aug. 2016, pp. 1–15.

[18] N. Newman, "Apple iBeacon technology briefing," *J. Direct Data Digit. Market. Pract.*, vol. 15, no. 3, pp. 222–225, 2014.

[19] P. Martin, B.-J. Ho, N. Grupen, S. Muñoz, and M. Srivastava, "An iBeacon primer for indoor localization: Demo abstract," in *Proc. 1st ACM Conf. Embedded Syst. Energy-Efficient Build.*, Nov. 2014, pp. 190–191.

[20] M. U. Farooq, M. Waseem, S. Mazhar, A. Khairi, and T. Kamal, "A review on Internet of Things (IoT)," *Int. J. Comput. Appl.*, vol. 113, no. 1, pp. 1–7, 2015.

[21] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of Things (IoT): A literature review," *J. Comput. Commun.*, vol. 3, no. 5, p. 164, 2015.

[22] K. K. Patel and S. M. Patel, "Internet of Things-IoT: Definition, characteristics, architecture, enabling technologies, application & future challenges," *Int. J. Eng. Sci. Comput.*, vol. 6, no. 5, pp. 6122–6131, 2016.

[23] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Bus. Horizons*, vol. 58, no. 4, pp. 431–440, 2015.

[24] R. Mahmoud, T. Yousuf, F. Aloul, and I. Zualkernan, "Internet of Things (IoT) security: Current status, challenges and prospective measures," in *Proc. 10th Int. Conf. Internet Technol. Secured Trans. (ICITST)*, London, U.K., Dec. 2015, pp. 336–341.

[25] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.

[26] J. Yin, Z. Yang, H. Cao, T. Liu, Z. Zhou, and C. Wu, "A survey on Bluetooth 5.0 and Mesh: New milestones of IoT," *ACM Trans. Sens. Netw.*, vol. 15, no. 3, pp. 1–29, 2019.

[27] V. Karagiannis, P. Chatzimisios, F. Vazquez-Gallego, and J. Alonso-Zarate, "A survey on application layer protocols for the Internet of Things," *Trans. IoT Cloud Comput.*, vol. 3, no. 1, pp. 11–17, 2015.

[28] Q. Zhu, R. Wang, Q. Chen, Y. Liu, and W. Qin, "IoT gateway: Bridging wireless sensor networks into Internet of Things," in *Proc. IEEE/IFIP Int. Conf. Embedded Ubiquitous Comput.*, Hong Kong, China, Dec. 2010, pp. 347–352.

[29] R. Cavallari, F. Martelli, R. Rosini, C. Buratti, and R. Verdone, "A survey on wireless body area networks: Technologies and design challenges," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 3, pp. 1635–1657, 3rd Quart., 2014.

[30] Y. H. Hwang, "IoT security & privacy: Threats and challenges," in *Proc. 1st ACM workshop IoT Privacy Trust Security*, Apr. 2015, p. 1.

[31] J. Pacheco and S. Hariri, "IoT security framework for smart cyber infrastructures," in *Proc. IEEE 1st Int. Workshops Found. Appl. Self Syst. (FAS* W)*, Sep. 2016, pp. 242–247.

[32] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, "Internet of Things security: A survey," *J. Netw. Comput. Appl.*, vol. 88, pp. 10–28, Jun. 2017.

[33] K. Zhao and L. Ge, "A survey on the Internet of Things security," in *Proc. 9th Int. Conf. Comput. Intell. Security*, Dec. 2013, pp. 663–667.

[34] M. Abomhara and G. M. Køien, "Security and privacy in the Internet of Things: Current status and open issues," in *Proc. Int. Conf. Privacy Security Mobile Syst. (PRISMS)*, May 2014, pp. 1–8.

[35] M. M. Hossain, M. Fotouhi, and R. Hasan, "Towards an analysis of security issues, challenges, and open problems in the Internet of Things," in *Proc. IEEE World Congr. Services*, New York, NY, USA, Jun./Jul. 2015, pp. 21–28.

[36] V. Hassija, V. Chamola, V. Saxena, D. Jain, P. Goyal, and B. Sikdar, "A survey on IoT security: Application areas, security threats, and solution architectures," *IEEE Access*, vol. 7, pp. 82721–82743, 2019.

[37] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, "A survey on security and privacy issues in Internet-of-Things," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1250–1258, Oct. 2017.

[38] D. M. Mendez, I. Papapanagiotou, and B. Yang, "Internet of Things: Survey on security and privacy," *Inf. Security J. Global Perspect.*, vol. 27, no. 3, pp. 162–182, 2018.

[39] J. Y. Kim, W. Hu, D. Sarkar, and S. Jha, "Long-term secure management of large scale Internet of Things applications," *J. Netw. Comput. Appl.*, vol. 138, pp. 15–26, Jul. 2019.

[40] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017.

[41] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for IoT security and privacy: The case study of a smart home," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom workshops)*, Kona, HI, USA, Mar. 2017, pp. 618–623.

[42] S. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K.-S. Kwak, "The Internet of Things for health care: A comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, 2015.

[43] A. I. Newaz, A. K. Sikder, M. Rahman, and A. S. Uluagac, "A survey on security and privacy issues in modern healthcare systems: Attacks and defenses," 2020, *arXiv:2005.07359*.

[44] A. K. Sikder, G. Petracca, H. Aksu, T. Jaeger, and A. Uluagac, "A survey on sensor-based threats to Internet-of-Things (IoT) devices and applications," 2018, *arXiv:1802.02041*.

[45] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT security techniques based on machine learning: How do IoT devices use ai to enhance security?" *IEEE Signal Process. Mag.*, vol. 35, no. 5, pp. 41–49, Sep. 2018.

[46] I. Butun, P. Österberg, and H. Song, "Security of the Internet of Things: Vulnerabilities, attacks and countermeasures," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 616–644, 1st Quart., 2020.

[47] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, and S. Shieh, "IoT security: Ongoing challenges and research opportunities," in *Proc. IEEE 7th Int. Conf. Service-Oriented Comput. Appl.*, Nov. 2014, pp. 230–234.

[48] M. A. Khan and K. Salah, "IoT security: Review, blockchain solutions, and open challenges," *Future Gener. Comput. Syst.*, vol. 82, pp. 395–411, May 2018.

[49] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of IoT systems: Design challenges and opportunities," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, San Jose, CA, USA, Nov. 2014, pp. 417–423.

[50] F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine learning in IoT security: Current solutions and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1686–1721, 3rd Quart., 2020.

[51] J. S. Kumar and D. R. Patel, "A survey on Internet of Things: Security and privacy issues," *Int. J. Comput. Appl.*, vol. 90, no. 11, pp. 20–26, 2014.

[52] L. Chen *et al.*, "Robustness, security and privacy in location-based services for future IoT: A survey," *IEEE Access*, vol. 5, pp. 8956–8977, 2017.

[53] W. Zhou, Y. Jia, A. Peng, Y. Zhang, and P. Liu, "The effect of IoT new features on security and privacy: New threats, existing solutions, and challenges yet to be solved," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1606–1616, Apr. 2019.

[54] P. M. Chanal and M. S. Kakkasageri, "Security and privacy in IoT: A survey," *Wireless Personal Commun.*, vol. 115, no. 2, pp. 1667–1693, 2020.

[55] L. Babun, K. Denney, Z. B. Celik, P. McDaniel, and A. S. Uluagac, "A survey on IoT platforms: Communication, security, and privacy perspectives," *Comput. Netw.*, vol. 192, Jun. 2021, Art. no. 108040.

[56] C. T. Hager and S. F. MidKiff, "An analysis of Bluetooth security vulnerabilities," in *Proc. IEEE Wireless Commun. Netw. (WCNC)*, Mar. 2003, pp. 1825–1831.

[57] Y. Zou, J. Zhu, X. Wang, and L. Hanzo, "A survey on wireless security: Technical challenges, recent advances, and future trends," *IEEE Access*, vol. 104, pp. 1727–1765, 2016.

[58] A. M. Lonzetta, P. Cope, J. Campbell, B. J. Mohd, and T. Hayajneh, "Security vulnerabilities in Bluetooth technology as used in IoT," *Sens. Actuator Netw.*, vol. 7, no. 3, p. 28, 2018.

[59] J. Dunning, "Taming the blue beast: A survey of Bluetooth based threats," *IEEE Security Privacy*, vol. 8, no. 2, pp. 20–27, Mar./Apr. 2010.

[60] H. Wen, Z. Lin, and Y. Zhang, "FirmXRay: Detecting Bluetooth link layer vulnerabilities from bare-metal firmware," in *Proc. ACM SIGSAC Conf. Comput. Commun.*, New York, NY, USA, Nov. 2020, pp. 167–180.

[61] N. Be-Nazir Ibn Minar and M. Tarique, "Bluetooth security threats and solutions: A survey," *Int. J. Distrib. Parallel Syst.*, vol. 3, no. 1, p. 127, 2012.

[62] S. S. Hassan, S. D. Bibon, M. S. Hossain, and M. Atiquzzaman, "Security threats in Bluetooth technology," *Comput. Security*, vol. 74, pp. 308–322, May 2018.

[63] B. Dunsbergen and K. Kubo, "Sleep control for network of Bluetooth low energy devices," U.S Patent 9 282 582, Mar. 2016.

[64] M. Radhakrishnan, A. Misra, R. K. Balan, and Y. Lee, "Smartphones and BLE services: Empirical insights," in *Proc. IEEE 12th Int. Conf. Mobile Ad Hoc Sens. Syst.*, Dallas, TX, USA, Oct. 2015, pp. 226–234.

[65] B. Yu, L. Xu, and Y. Li, "Bluetooth low energy (BLE) based mobile electrocardiogram monitoring system," in *Proc. IEEE Int. Conf. Inf. Autom.*, Shenyang, China, Jun. 2012, pp. 763–767.

[66] C. Gomez, J. Oller, and J. Paradells, "Overview and evaluation of Bluetooth low energy: An emerging low-power wireless technology," *Sensors*, vol. 12, no. 9, pp. 11734–11753, 2012.

[67] R. Heydon and N. Hunn. "Bluetooth Low Energy." Bluetooth SIG. 2012. [Online]. Available: https://www.bluetooth.org/DocMan/handlers/DownloadDoc.ashx

[68] N. K. Gupta, *Inside Bluetooth Low Energy*. Boston, MA, USA: Artech House, 2016.

[69] A. H. Omre and S. Keeping, "Bluetooth low energy: Wireless connectivity for medical monitoring," *Diabetes Sci. Technol.*, vol. 4, no. 2, pp. 457–463, 2010.

[70] X. Fafoutis *et al.*, "Designing wearable sensing platforms for healthcare in a residential environment," *EAI Endorsed Trans. Pervasive Health Technol.*, vol. 3, no. 12, p. e1, 2017.

[71] L. Guo-Cheng and Y. Hong-Yang, "Design and implementation of a Bluetooth 4.0-based heart rate monitor system on iOS platform," in *Proc. Int. Conf. Commun. Circuits Syst.*, vol. 2. Chengdu, China, Nov. 2013, pp. 112–115.

[72] A. M. Chan, N. Selvaraj, N. Ferdosi, and R. Narasimhan, "Wireless patch sensor for remote monitoring of heart rate, respiration, activity, and falls," in *Proc. 35th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBS)*, Jul. 2013, pp. 6115–6118.

[73] M. Collotta and G. Pau, "A novel energy management approach for smart homes using Bluetooth low energy," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 12, pp. 2988–2996, Dec. 2015.

[74] Y. Prakash, V. Biradar, S. Vincent, M. Martin, and A. Jadhav, "Smart Bluetooth low energy security system," in *Proc. Int. Conf. Wireless Commun. Signal Process. Netw.*, Chennai, India, Mar. 2017, pp. 2141–2146.

[75] J.-R. Lin, T. Talty, and O. K. Tonguz, "On the potential of Bluetooth low energy technology for vehicular applications," *IEEE Commun. Mag.*, vol. 53, no. 1, pp. 267–275, Jan. 2015.

[76] A. Rose, J. Bindewald, B. Ramsey, M. Rice, and B. Mullins, "Securing Bluetooth low energy locks from unauthorized access and surveillance," in *Proc. Int. Conf. Crit. Infrastruct. Protect.*, Arlington, VA, USA, Mar. 2017, pp. 319–338.

[77] R. Karani *et al.*, "Implementation and design issues for using Bluetooth low energy in passive keyless entry systems," in *Proc. IEEE Annu. India Conf.*, Bangalore, India, Dec. 2016, pp. 1–6.

[78] A. Rose and B. Ramsey, "Picking Bluetooth low energy locks a quarter mille away," 2016. Accessed: Feb. 9, 2022. [Online]. Available: https://doi.org/10.5446/36217

[79] R. Faragher and R. Harle, "Location fingerprinting with Bluetooth low energy beacons," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 11, pp. 2418–2428, Nov. 2015.

[80] Q. Zhang and Z. Liang, "Security analysis of Bluetooth low energy based smart wristbands," in *Proc. 2nd Int. Conf. Front. Sens. Technol. (ICFST)*, Shenzhen, China, Apr, 2017, pp. 421–425.

[81] W. K. Zegeye, "Exploiting Bluetooth low energy pairing vulnerability in telemedicine," in *Proc. Int. Telemetering Conf.*, Oct. 2015, pp. 1–10.

[82] D. Antonioli, N. O. Tippenhauer, K. B. Rasmussen, and M. Payer, "BLURtooth: Exploiting cross-transport key derivation in Bluetooth classic and Bluetooth low energy," 2020, *arXiv:2009.11776*.

[83] S. Sevier and A. Tekeoglu, "Analyzing the security of Bluetooth low energy," in *Proc. Int. Conf. Electron. Inf. Commun. (ICEIC)*, Jan. 2019, pp. 1–5.

[84] S.-C. Cha, C.-Y. Dai, and J.-F. Chen, "Is there a tradeoff between privacy and security in BLE-based IoT applications: Using a smart vehicle of a major taiwanese brand as example," in *Proc. IEEE 5th Global Conf. Consum. Electron.*, Kyoto, Japan, Oct 2016, pp. 1–4.

[85] A. K. Das, P. H. Pathak, C.-N. Chuah, and P. Mohapatra, "Uncovering privacy leakage in BLE network traffic of wearable fitness trackers," in *Proc. 17th Int. Workshop Mobile Comput. Syst. Appl.*, Feb. 2016, pp. 99–104.

[86] H. O'Sullivan, *Security Vulnerabilities of Bluetooth Low Energy Technology (BLE)*, Tufts Univ., Medford, MA, USA, 2015.

[87] Y. Zhang, J. Weng, R. Dey, and X. Fu, "Bluetooth low energy (BLE) security and privacy," in *Encyclopedia of Wireless Networks*, Springer, Oct. 2019, pp. 1–12.

[88] J. C. Haartsen, "The Bluetooth radio system," *IEEE Personal Commun.*, vol. 7, no. 1, pp. 28–36, Feb. 2000.

[89] N. Wang, N. Zhang, and M. Wang, *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)*, IEEE Standard 802.15, 2005, pp. 1–700, doi: 10.1109/IEEESTD.2005.96290.

[90] "Bluetooth Low Energy—Wikipedia, the Free Encyclopedia." Wikipedia Contributors. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Bluetooth_Low_Energy&oldid=942789780 (accessed Jul. 17, 2020).

[91] "Bluetooth vs BLE-Difference Between Bluetooth and BLE (Bluetooth Low Energy)." RF Wireless World. [Online]. Available: https://www.rfwireless-world.com/Terminology/Bluetooth-vs-BLE.html (accessed Jul. 17, 2020).

[92] S. Morehead. "How to Pick the Best Bluetooth Protocol for Your Application." 2019. [Online]. Available: https://www.mwrf.com/technologies/systems/article/21849843/how-to-pick-the-best-bluetooth-protocol-for-your-application (accessed Jan. 10, 2022).

[93] K. Scarfone and J. Padgette, *Guide to Bluetooth Security*, vol. 800, NIST, Gaithersburg, MD, USA, 2008, p. 121.

[94] M. Bon. "A Basic Introduction to BLE Security." [Online]. Available: https://www.digikey.com/eewiki/display/Wireless/A+Basic+Introduction+to+BLE+Security (accessed Jul. 17, 2020).

[95] K. Haataja, *Security Threats and Countermeasures in Bluetooth-Enabled Systems*, Univ. Kuopio, Kuopio, Finland, 2009.

[96] T. Melamed, "An active man-in-the-middle attack on Bluetooth smart devices," *Safety Security Stud.*, vol. 8, no. 2, pp. 200–211, 2018.

[97] S. Sarkar, J. Liu, and E. Jovanov, "A robust algorithm for sniffing BLE long-lived connections in real-time," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Waikoloa, HI, USA, USA, Dec. 2019, pp. 1–6.

[98] A. C. Santos, J. L. Soares Filho, Á. Í. Silva, V. Nigam, and I. E. Fonseca, "BLE injection-free attack: A novel attack on Bluetooth low energy devices," *J. Ambient Intell. Humanized Comput.*, vol. 20, pp. 1–11, Sep. 2019.

[99] Q. Zhang, Z. Liang, and Z. Cai, "Developing a new security framework for Bluetooth low energy devices," *Comput. Mater. Continua*, vol. 59, no. 2, pp. 457–471, 2019.

[100] Z. Cekerevac, Z. Dvorak, L. Prigoda, and P. Cekerevac, "Internet of Things and the man-in-the-middle attacks-security and economic risks," *MEST J.*, vol. 5, no. 2, pp. 15–25, 2017.

[101] M. A. Albahar, K. Haataja, and P. Toivanen, "Bluetooth MiTM vulnerabilities: A literature review, novel attack scenarios, novel countermeasures, and lessons learned," *Int. J. Inf. Technol. Security*, vol. 8, no. 4, pp. 25–49, 2016.

[102] D.-Z. Sun, Y. Mu, and W. Susilo, "Man-in-the-middle attacks on secure simple pairing in Bluetooth standard V5.0 and its countermeasure," *Pers. Ubiquitous Comput.*, vol. 22, no. 1, pp. 55–67, 2018.

[103] M. Yaseen *et al.*, "MARC: A novel framework for detecting MITM attacks in eHealthcare BLE systems," *J. Med. Syst.*, vol. 43, no. 11, p. 324, 2019.

[104] D. Antonioli, N. O. Tippenhauer, and K. Rasmussen, "Low entropy key negotiation attacks on Bluetooth and Bluetooth low energy," *ACM Trans. Privacy Security*, vol. 23, no. 3, pp. 2471–2566, 2020.

[105] A. Ray, V. Raj, M. Oriol, A. Monot, and S. Obermeier, "Bluetooth low energy devices security testing framework," in *Proc. 11th Int. Conf. Softw. Test. Verif. Validation (ICST)*, Vasteras, Sweden, Apr. 2018, pp. 384–393.

[106] B. Cyr, W. Horn, D. Miao, and M. Specter, *Security Analysis of Wearable Fitness Devices (Fitbit)*, vol. 1, Massachusetts Inst. Technol., Cambridge, MA, USA, 2014.

[107] P. Syverson, "A taxonomy of replay attacks [cryptographic protocols]," in *Proc. Comput. Security Found. Workshop VII*, Franconia, NH, USA, Jun. 1994, pp. 187–191.

[108] J. D. Guttman, "Security protocol design via authentication tests," in *Proc. 15th IEEE Comput. Security Found. Workshop (CSFW)*, Cape Breton, NS, Canada, Jun. 2002, pp. 92–103.

[109] E. D. Cardenas, *MAC Spoofing–An Introduction, GIAC Security Essentials Certification (GSEC)*, SANS Inst., Bethesda, MD, USA, 2003.

[110] W. Oliff, A. Filippoupolitis, and G. Loukas, "Evaluating the impact of malicious spoofing attacks on Bluetooth low energy based occupancy detection systems," in *Proc. IEEE 15th Int. Conf. Softw. Eng. Res. Manag. Appl. (SERA)*, London, U.K., Jun. 2017, pp. 379–385.

[111] A. Kumar, N. Saxena, G. Tsudik, and E. Uzun, "Caveat eptor: A comparative study of secure device pairing methods," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, Galveston, TX, USA, Mar. 2009, pp. 1–10.

[112] Y. Zhang, J. Weng, R. Dey, Y. Jin, Z. Lin, and X. Fu, "Breaking secure pairing of Bluetooth low energy using downgrade attacks," in *Proc. 29th USENIX Security Symp.*, Boston, MA, USA, Aug. 2020, pp. 37–54.

[113] J. Wu, Y. Nan, V. Kumar, M. Payer, and D. Xu, "BlueShield: Detecting spoofing attacks in Bluetooth low energy networks," in *Proc. 23rd Int. Symp. Res. Attacks Intrusions Defenses (RAID)*, Donostia-San Sebastian, Spain, Oct. 2020, pp. 397–411.

[114] Y. Shaked and A. Wool, "Cracking the Bluetooth pin," in *Proc. 3rd Int. Conf. Mobile Syst. Appl. Serv.*, Seattle, WA, USA, Jun. 2005, pp. 39–50.

[115] T. Rosa, "Bypassing passkey authentication in Bluetooth low energy," IACR Cryptol. ePrint Archive, Lyon, France, Rep. 2013/309, 2013.

[116] F. Stajano and R. Anderson, "The resurrecting duckling: Security issues for ad-hoc wireless networks," in *Proc. Int. Workshop Security Protocols*, Cambridge, U.K., Apr. 1999, pp. 172–182.

[117] T. Martin, M. Hsiao, D. Ha, and J. Krishnaswami, "Denial-of-service attacks on battery-powered mobile computers," in *Proc. 2nd IEEE Annu. Conf. Pervasive Comput. Commun.*, Orlando, FL, USA, Mar. 2004, pp. 309–318.

[118] E. Mackensen, M. Lai, and T. M. Wendt, "Bluetooth low energy (BLE) based wireless sensors," in *Proc. IEEE Sensors*, Taipei, Taiwan, Oct. 2012, pp. 1–4.

[119] J. Hughes, J. Yan, and K. Soga, "Development of wireless sensor network using Bluetooth low energy (BLE) for construction noise monitoring," *Int. J. Smart Sens. Intell. Syst.*, vol. 8, no. 2, pp. 1379–1405, 2015.

[120] Z. Guo, I. G. Harris, Y. Jiang, and L.-F. Tsaur, "An efficient approach to prevent battery exhaustion attack on BLE-based mesh networks," in *Proc. IEEE Int. Conf. Comput. Netw. Commun. (ICNC)*, Silicon Valley, CA, USA, Jan. 2017, pp. 1–5.

[121] S. Brauer, A. Zubow, S. Zehl, M. Roshandel, and S. Mashhadi-Sohi, "On practical selective jamming of Bluetooth low energy advertising," in *Proc. IEEE Conf. Stand. Commun. Netw. (CSCN)*, Berlin, Germany, Oct./Nov. 2016, pp. 1–6.

[122] A. A. Pammi, "Threats, countermeasures, and research trends for BLE-based IoT devices," M.S. thesis, Dept. Comput. Sci., Arizona State Univ., Tempe, AZ, USA, 2017.

[123] P. Stirparo, J. Loeschner, and M. Cattani, "Bluetooth technology: Security features, vulnerabilities and attacks," JRC Sci., Ispra, Italy, Rep. JRC 68414, Apr. 2011, p. 27.

[124] C. Zuo, H. Wen, Z. Lin, and Y. Zhang, "Automatic fingerprinting of vulnerable BLE IoT devices with static UUIDs from mobile apps," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, London, U.K., Nov. 2019, pp. 1469–1483.

[125] G. Balakrishnan and T. Reps, "Analyzing memory accesses in x86 executables," in *Proc. Int. Conf. Compiler Constr.*, Barcelona, Spain, Mar./Apr. 2004, pp. 5–23.

[126] G. Celosia and M. Cunche, "Fingerprinting Bluetooth-low-energy devices based on the generic attribute profile," in *Proc. 2nd Int. ACM Workshop Security Privacy Internet Things*, London, U.K., Nov. 2019, pp. 24–31.

[127] C. Wang, X. Guo, Y. Wang, Y. Chen, and B. Liu, "Friend or foe?: Your wearable devices reveal your personal pin," in *Proc. 11th ACM Asia Conf. Comput. Commun. Security*, Xi'an China, May 2016, pp. 189–200.

[128] K.-Y. Lam, N. W.-H. Tsang, S. Han, W. Zhang, J. K.-Y. Ng, and A. Nath, "Activity tracking and monitoring of patients with alzheimer's disease," *Multimedia Tools Appl.*, vol. 76, no. 1, pp. 489–521, 2017.

[129] A. Korolova and V. Sharma, "Cross-app tracking via nearby Bluetooth low energy devices," in *Proc. 8th ACM Conf. Data Appl. Security Privacy*, Tempe, AZ, USA, Mar. 2018, pp. 43–52.

[130] S. Soderi, "Cybersecurity assessment of the polar Bluetooth low energy heart-rate sensor," in *Proc. EAI Int. Conf. Body Area Netw.*, Florence, Italy, Nov. 2019, pp. 252–265.

[131] P. Wang, "Bluetooth low energy—Privacy enhancement for advertisement," M.S. thesis, Dept. Telematics, Inst. Telematics, Lübeck Germany, 2014.

[132] K. Fawaz, K.-H. Kim, and K. G. Shin, "Protecting privacy of BLE device users," in *Proc. 25th USENIX Security Symp.*, Austin, TX, USA, Aug. 2016, pp. 1205–1221.

[133] S.-C. Cha, M.-S. Chuang, K.-H. Yeh, Z.-J. Huang, and C. Su, "A user-friendly privacy framework for users to achieve consents with nearby BLE devices," *IEEE Access*, vol. 6, pp. 20779–20787, 2018.

[134] M. Herfurt and C. Mulliner, "Remote device identification based on Bluetooth fingerprinting techniques," Trifinite Group, Amsterdam, The Netherlands, Rep., 2004.

[135] "nRF Connect for Mobile." Nordic Semiconductor ASA. [Online]. Available: https://play.google.com/store/apps/details?id=no.nordicsemi.android.mcp&hl=en (accessed Jul. 17, 2020).

[136] P. Sivakumaran and J. Blasco, "A study of the feasibility of co-located app attacks against BLE and a large-scale analysis of the current application-layer security landscape," in *Proc. 28th USENIX Security Symp. (USENIX Security)*, Santa Clara, CA, USA, Aug. 2019, pp. 1–18.

[137] M. Naveed, X.-Y. Zhou, S. Demetriou, X. Wang, and C. A. Gunter, "Inside job: Understanding and mitigating the threat of external device MIS-binding on android," in *Proc. NDSS Symp.*, San Diego, CA, USA, Feb. 2014, pp. 23–26.

[138] Z.-M. Lin, C.-H. Chang, N.-K. Chou, and Y.-H. Lin, "Bluetooth low energy (BLE) based blood pressure monitoring system," in *Proc. Int. Conf. Intell. Green Build. Smart Grid*, Taipei, Taiwan, Apr. 2014, pp. 1–4.

[139] M. Garbarino, M. Lai, D. Bender, R. W. Picard, and S. Tognetti, "Empatica E3—A wearable wireless multi-sensor device for real-time computerized biofeedback and data acquisition," in *Proc. IEEE 4th Int. Conf. Wireless Mobile Commun. Healthcare-Transforming Healthcare Through Innov. Mobile Wireless Technol. (MOBIHEALTH*, Athens, Greece, Nov. 2014, pp. 39–42.

[140] M. E. Garbelini, S. Chattopadhyay, and C. Wang, "SweynTooth: Unleashing mayhem over Bluetooth low energy," in *LL Encryption Procedure, Channels*, vol. 37, USENIX Assoc., 2020, pp. 39–40.

[141] W. Zhou and S. Piramuthu, "Security/privacy of wearable fitness tracking IoT devices," in *Proc. 9th Iberian Conf. Inf. Syst. Technol. (CISTI)*, Barcelona, Spain, Jun. 2014, pp. 1–5.

[142] M. Rahman, B. Carbunar, and M. Banik, "Fit and vulnerable: Attacks and defenses for a health monitoring device," in *Proc. IEEE Symp. Security Privacy*, San Francisco, CA, USA, May 2013, pp. 447–459.

[143] O. Arias, J. Wurm, K. Hoang, and Y. Jin, "Privacy and security in Internet of Things and wearable devices," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 1, no. 2, pp. 99–109, Apr.–Jun. 2015.

[144] H. Wang, T. T.-T. Lai, and R. R. Choudhury, "MoLe: Motion leaks through smartwatch sensors," in *Proc. 21st Annu. Int. Conf. Mobile Comput. Netw.*, Paris, France, Sep. 2015, pp. 155–166.

[145] T. O. Arney, "A literature review on the current state of security and privacy of medical devices and sensors with Bluetooth low energy," M.S. thesis, Dept. Master Sci. Med. Informat., Michigan Technol. Univ., Houghton, MI, USA, 2018.

[146] A. Hilts, C. Parsons, and J. Knockel, "Every step you fake: A comparative analysis of fitness tracker privacy and security," *Open Effect Report*, vol. 76, no. 24, pp. 31–33, 2016.

[147] M. L. Hale, D. Ellis, R. Gamble, C. Waler, and J. Lin, "Secu wear: An open source, multi-component hardware/software platform for exploring wearable security," in *Proc. IEEE Int. Conf. Mobile Serv.*, New York, NY, USA, Jun./Jul. 2015, pp. 97–104.

[148] M. Kumar, "Security issues and privacy concerns in the implementation of wireless body area network," in *Proc. Int. Conf. Inf. Technol.*, Bhubaneswar, India, Dec. 2014, pp. 58–62.

[149] M. B. Barcena, C. Wueest, and H. Lau, *How Safe Is Your Quantified Self*, vol. 16, Symantech, Mountain View, CA, USA, 2014.

[150] "The Attack Vector "BlueBorn" Exposes Almost Every Connected Device." ARMIS®. [Online]. Available: https://www.armis.com/blueborne/ (accessed Jul. 17, 2020).

[151] Y. Ojha. "I Hacked MiBand 3, and Here Is How I Did It." Medium Blog. 2018. [Online]. Available: https://medium.com/@yogeshojha/i-hacked-xiaomi-miband-3-and-here-is-how-i-did-it-43d68c272391 (accessed Jul. 17, 2020).

[152] Y. Ojha. "MiBand3." 2018. [Online]. Available: https://github.com/yogeshojha/MiBand3 (accessed Jul. 17, 2020).

[153] D. Olenick. "Design Flaw Leaves Bluetooth Devices Vulnerable." [Online]. Available: https://www.scmagazine.com/home/security-news/mobile-security/design-flaw-leaves-bluetooth-devices-vulnerable/ (accessed Jul. 17, 2020).

[154] "Ubertooth One." Great Scott Gadgets. 2020. [Online]. Available: https://greatscottgadgets.com/ubertoothone/ (accessed Jul. 17, 2020).

[155] M. Ryan. "Crackle, Crack Bluetooth Smart (BLE) Encryption." 2015. [Online]. Available: http://lacklustre.net/projects/crackle/ (accessed Jul. 17, 2020).

[156] S. Jasek. "A Node.js Package for BLE (Bluetooth Low Energy) Security Assessment Using Man-in-the-Middle and Other Attacks." [Online]. Available: https://github.com/securing/gattacker (accessed Jul. 17, 2020).

[157] M. Ossmann and D. Spill. "Software, Firmware and Hardware Designs for Ubertooth." [Online]. Available: https://github.com/greatscottgadgets/ubertooth (accessed Jul. 17, 2020).

[158] "Bluefruit LE Sniffer." Adafruit. [Online]. Available: https://www.adafruit.com/product/2269 (accessed Jul. 17, 2020).

[159] "SmartRF Protocol Packet Sniffer." Texas Instruments. [Online]. Available: http://www.ti.com/tool/PACKET-SNIFFER (accessed Jul. 17, 2020).

[160] "BtleJuice." Btlejuice. 2016. [Online]. Available: https://github.com/DigitalSecurity/BtleJuice (accessed Jul. 17, 2020).

[161] "Bluetooth Mesh Security Overview." Bluetooth SIG. 2017. [Online]. Available: https://www.bluetooth.com/blog/bluetooth-mesh-security-overview/ (accessed Jul. 17, 2020).

[162] H. Wang, M. Xi, J. Liu, and C. Chen, "Transmitting IPv6 packets over Bluetooth low energy based on BlueZ," in *Proc. 15th Int. Conf. Adv. Commun. Technol.*, PyeongChang, South Korea, Jan. 2013, pp. 72–77.

[163] J. Nieminen, T. Savolainen, M. Isomaki, B. Patil, Z. Shelby, and C. Gomez, "IPv6 over BLUETOOTH(R) low energy," IETF, RFC 7668, 2015.

[164] M. Spörk, C. A. Boano, M. Zimmerling, and K. Römer, "BLEach: Exploiting the full potential of IPv6 over BLE in constrained embedded IoT devices," in *Proc. 15th ACM Conf. Embedded Netw. Sens. Syst.*, New York, NY, USA, Nov. 2017, pp. 1–14.

[165] P. Kriz, F. Maly, and T. Kozel, "Improving indoor localization using Bluetooth low energy beacons," *Mobile Inf. Syst.*, vol. 2016, Apr. 2016, Art. no. 2083094.

[166] Y. Zhuang, J. Yang, Y. Li, L. Qi, and N. El-Sheimy, "Smartphone-based indoor localization with Bluetooth low energy beacons," *Sensors*, vol. 16, no. 5, p. 596, 2016.

[167] Y. Tian, B. Huang, B. Jia, and L. Zhao, "Optimizing AP and beacon placement in WiFi and BLE hybrid localization," *J. Netw. Comput. Appl.*, vol. 164, 2020, Art. no. 102673.

[168] M. Bocca, V. Jain, C. Lang, and H. Lee, "Bluetooth low energy (BLE) passive vehicle access control system for defending the system against relay attacks and method thereof," U.S. Patent 10 124 768, Nov. 13, 2018.

[169] Y.-C. Pu and P.-C. You, "Indoor positioning system based on BLE location fingerprinting with classification approach," *Appl. Math. Model.*, vol. 62, pp. 654–663, Oct. 2018.

[170] D. Baldie, "System and method for providing a Bluetooth low energy mobile payment system," U.S. Patent 10 062 073, Aug. 28, 2018.

[171] A. Yohan, N.-W. Lo, and D. Winata, "An indoor positioning-based mobile payment system using Bluetooth low energy technology," *Sensors*, vol. 18, no. 4, p. 974, 2018.

[172] M. R. Ghori, T.-C. Wan, and G. C. Sodhy, "Bluetooth low energy Mesh networks: Survey of communication and security protocols," *Sensors*, vol. 20, no. 12, p. 3590, 2020.

[173] S. M. Darroudi and C. Gomez, "Bluetooth low energy mesh networks: A survey," *Sensors*, vol. 17, no. 7, p. 1467, 2017.

[174] L. Leonardi, G. Patti, and L. L. Bello, "Multi-hop real-time communications over Bluetooth low energy industrial wireless mesh networks," *IEEE Access*, vol. 6, pp. 26505–26519, 2018.

[175] K. Lotfy and M. L. Hale, "Assessing pairing and data exchange mechanism security in the wearable Internet of Things," in *Proc. Int. Conf. Mobile Serv.*, San Francisco, CA, USA, Jun./Jul. 2016, pp. 25–32.

[176] J. Seo, K. Cho, W. Cho, G. Park, and K. Han, "A discovery scheme based on carrier sensing in self-organizing Bluetooth low energy networks," *J. Netw. Comput. Appl.*, vol. 65, pp. 72–83, Apr. 2016.

[177] J. Yang, C. Poellabauer, P. Mitra, and C. Neubecker, "Beyond beaconing: Emerging applications and challenges of BLE," *Ad Hoc Netw.*, vol. 97, Feb. 2020, Art. no. 102015.

[178] S.-C. Cha, J.-F. Chen, C. Su, and K.-H. Yeh, "A blockchain connected gateway for BLE-based devices in the Internet of Things," *IEEE Access*, vol. 6, pp. 24639–24649, 2018.

[179] B. Farzaneh, M. A. Montazeri, and S. Jamali, "An anomaly-based IDS for detecting attacks in RPL-based Internet of Things," in *Proc. IEEE 5th Int. Conf. Web Res.*, Tehran, Iran, Apr. 2019, pp. 61–66.

[180] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.

[181] R. V. Kulkarni and G. K. Venayagamoorthy, "Neural network based secure media access control protocol for wireless sensor networks," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Atlanta, GA, USA, Jun. 2009, pp. 1680–1687.

[182] M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1996–2018, 4th Quart., 2014.
[183] C. Shi, J. Liu, H. Liu, and Y. Chen, "Smart user authentication through actuation of daily activities leveraging WiFi-enabled IoT," in *Proc. 18th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, New York, NY, USA, Jul. 2017, pp. 1–10.
[184] M. Ozay, I. Esnaola, F. T. Y. Vural, S. R. Kulkarni, and H. V. Poor, "Machine learning methods for attack detection in the smart grid," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 8, pp. 1773–1786, Aug. 2016.
[185] L. Xiao, Q. Yan, W. Lou, G. Chen, and Y. T. Hou, "Proximity-based security techniques for mobile users in wireless networks," *IEEE Trans. Inf. Forensics Security*, vol. 8, pp. 2089–2100, 2013.
[186] M. Kuzlu, C. Fair, and O. Guler, "Role of artificial intelligence in the Internet of Things (IoT) cybersecurity," *Discover Internet Things*, vol. 1, no. 1, pp. 1–14, 2021.
[187] M. J. Baucas and P. Spachos, "Permissioned Blockchain-driven Internet of Things gateway using Bluetooth low energy," in *Proc. Int. Conf. Commun.*, Dublin, Ireland, Jun. 2020, pp. 1–6.
[188] J. Wang *et al.*, "A blockchain-based eHealthcare system interoperating with WBANs," *Future Gener. Comput. Syst.*, vol. 110, pp. 675–685, Sep. 2020.
[189] B. K. Mohanta, D. Jena, U. Satapathy, and S. Patnaik, "Survey on IoT security: Challenges and solution using machine learning, artificial intelligence and blockchain technology," *Internet Things*, vol. 11, Sep. 2020, Art. no. 100227.

ARUP BARUA received the B.Sc. degree in computer science and engineering from the Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh, in 2019. He is currently working as a Software Engineer with Samsung R&D Institute, Bangladesh, and also a Graduate Researcher with BUET. His research interests include computer network, network security and privacy, machine learning, and AI.

MD ABDULLAH AL ALAMIN (Graduate Student Member, IEEE) received the B.Sc. degree in computer science and engineering from the Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2017. He is currently pursuing the graduate degree with the University of Calgary, Canada. He has three years of working experience as a Software Engineer. His research interests include software security, privacy in the social network, data privacy, and IoT. He is interested in applying machine-learning technology in building tools that would help developers to be more productive and assist them to create secured privacy-preserving software.

MD. SHOHRAB HOSSAIN (Member, IEEE) received the B.Sc. and M.Sc. degrees in computer science and engineering from the Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh, in 2003 and 2007, respectively, and the Ph.D. degree from the School of Computer Science, University of Oklahoma, Norman, OK, USA, in December 2012. During the Ph.D. degree, he worked under NASA funded projects related to survivability, scalability, and security of space networks. He is currently serving as a Professor with the Department of Computer Science and Engineering, BUET. His research interests include mobile malware detections, cyber security, software-defined networking, security of mobile and ad hoc networks, and Internet of Things. He has published more than 85 technical research papers in leading journals and conferences, including *Journal of Computers & Security*, IEEE ACCESS, *Journal of Network and Computer Applications*, *Journal of Telecommunication Systems*, *Wireless Personal Communication*, *PLoS ONE*, IEEE GLOBECOM, IEEE ICC, IEEE MILCOM, IEEE HPSR, and IEEE HPCC. He has been serving as the TPC Member for IEEE GLOBECOM, IEEE ICC, and IEEE VTC, and a reviewer for *Wireless Personal Communication*, (Springer), *Journal of Network and Computer Applications* (Elsevier), and IEEE WIRELESS COMMUNICATIONS.

EKRAM HOSSAIN (Fellow, IEEE) is a Professor with the Department of Electrical and Computer Engineering, University of Manitoba, Canada (http://home.cc.umanitoba.ca/∼hossaina). His current research interests include design, analysis, and optimization of wireless networks with emphasis on beyond 5G cellular networks. He was elevated to an IEEE Fellow "for contributions to spectrum management and resource allocation in cognitive and cellular radio networks." He was listed as a Clarivate Analytics Highly Cited Researcher in Computer Science in 2017–2021. He served as the Editor-in-Chief for IEEE PRESS from 2018 to 2021 and the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS from 2012 to 2016. He is a member (Class of 2016) of the College of the Royal Society of Canada, a Fellow of the Canadian Academy of Engineering, and a Fellow of the Engineering Institute of Canada.