Bluetooth Low Energy Data Security Testing

Mustafa İPEKBAYRAK
İstanbul Üniversitesi-Cerrahpaşa
mustafa.ipekbayrak@istanbul.edu.tr

Ebu Yusuf GÜVEN
İstanbul Üniversitesi-Cerrahpaşa
eyguven@istanbul.edu.tr

Muhammed Ali AYDIN
İstanbul Üniversitesi-Cerrahpaşa
aydinali@istanbul.edu.tr

***Abstract*—Wireless connected devices bring our lives many advantages. BLE (Bluetooth Low Energy) is one of the most useful wireless connection protocols because of its low energy consumption. Although it is known that BLE is a secure peer to peer protocol, many attacks are achieved. While attacker involves Bluetooth range, devices can be compromised. In this work, security test scenarios on BLE devices communication are empathized. From the flow chart below, BLE security test steps are shown.**

Keywords— Bluetooth; BLE; Bluetooh Low Energy; Security; Mobil Wirelless Security Introduction

I. INTRODUCTION

Bluetooth is a wireless technology standard for exchanging data between fixed and mobile devices over short distances using short-wavelength UHF radio waves in the industrial, scientific and medical radio bands, from 2.400 to 2.485 GHz, and building personal area networks (PANs). It was originally conceived as a wireless alternative to RS-232 data cables [1].

Bluetooth has many security features such as frequency hopping and pairing. However, there are much vulnerability like bluejacking, bluebugging and car whispering. Bluejacking uses vCard sending feature of Bluetooth. These messages are automatically opened. Bluebugging is about accessing a Bluetooth device remotely. Car Whispering is about receiving and sending audio data to car stereo system.

In this work, developed BLE test classification is explained. Then, Bluetooth analyzing and sniffing method with creating Bluetooth proxy using two Bluetooth dongles on a computer is mentioned. Lastly, BLE security test with an android operating system mobile phone is explained.

II. EXPERIMENTAL SETTINGS

There was a dish washer with Bluetooth connection. Bluetooth connection was for home use monitoring purposes. The machine was also our test subject. We tried to copy its data communication and replying to see if the device malfunction or something ominous happen.

The other device was a rooted android operating system phone. It is used to create a debug application and collect communication data between the dish washer and mobile application.

Two Bluetooth dongles are used with a laptop installed Linux operating system. Virtual machine installation was also used in our setup in order to use second Bluetooth dongle in another virtual Linux environment. Two dongles were used to create a proxy to read data. In this test path, some devices are used to create a Bluetooth test environment as seen in **Figure 1.**

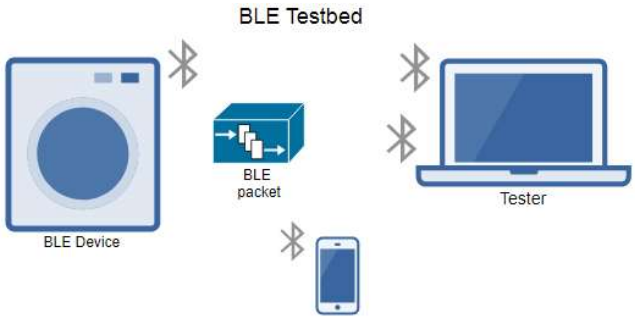


Fig. 1: BLE testbed include BLE device, smartphone and tester computer with double BLE adaptor

III. BLE TEST CLASSIFICATION

To be able to handle a flawless Bluetooth test, we created test classifications according environments that we can listen the data traffic. Three types of test environments can be seen in **Figure 2**. One of them is using only Bluetooth radio waves, other one is using low end microcontroller device which supports both peripheral and master Bluetooth mode and last one in on a UNIX system.

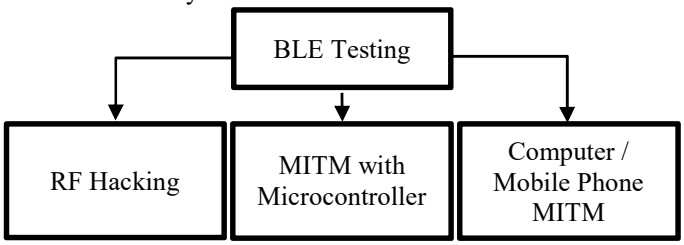


Fig. 2: BLE testing classification.

A. Radio Frequency Hacking

These kinds of attacks require tools like HackRF which can sniff Bluetooth radio waves and reply the same frequencies. You can see this device in **Figure 3.** Currently we are not able to create these kinds of tests.



Fig. 3: HackRF Device which captures and emits radio frequencies.

B. MITM Attack with Microcontroller

Using a microcontroller like ESP, it is possible to test Bluetooth communication and carry out security tests because ESP can act as both master and peripheral Bluetooth device. This test is out of our scope for now [8][9][10].

C. Computer / Mobile Phone MITM

This test step separates to two parts as it's name; computer and mobile phone tests. However desired test types can make both of these devices. But for ease of use, it is divided to two parts. In the computer test, we use two Bluetooth dongle and use one of these dongle as master and the other one as peripheral. A proxy is created between these dongles [12]. Target Bluetooth device is connected to master dongle. Target Bluetooth application device should be connected to peripheral dongle. Data between these dongles can be read and manipulated.

The other test which requires an Android mobile phone consists of build in features. A debug application is created to make more use of these features. Bluetooth communication log data can be collected using an android phone with developer settings enabled.

IV. MITM USING A COMPUTER

An interesting approach for analyzing and replying BLE signals is creating a Bluetooth proxy. This method makes it possible to read several of devices Bluetooth data just like listening wireless network data. This test can work with using Bluetooth 4.0 or higher Bluetooth adapter only. **Figure 4** shows a sample Bluetooth 4.0 USB dongle.



Fig. 4: A USB Bluetooth 4.0 dongle.

To be able to accomplish this test, two Bluetooth dongles are required. This method called Bluetooth proxy. A computer with virtual operating system or two computers are necessary. Data flow from peripheral BLE device to application can be seen in **Figure 5** during the MITM attack.

To start explaining the test; first Bluetooth dongle which acts as a client BLE device, should connect to the victim Bluetooth device. Generally mobile phone applications act as clients, but in our test, with the help of a computer program, computer can connect to the device. All connection data between computer and Bluetooth device are copied. Now data passes through sniffing and manipulating codes if it is necessary. After that, data replicates to the other computer's Bluetooth dongle. Now, mobile phone application should connect to the second Bluetooth dongle. In this position, second Bluetooth dongle acts as server. Actual BLE application which should connect to the device must connect to computer's dongle which mimics as actual Bluetooth device. This will allow tester to read and change Bluetooth data that flows between device and application. This method can also help for a reply attack.

To be able to use this technique with only one computer; a virtual machine and a second Bluetooth dongle for the virtual machine is necessary. Sometimes it is challenging to make virtual machine to see the second Bluetooth dongle. Removing and plugging adapter back to USB makes virtual machine software to see external adapters.

There are tools like btproxy, gattacker, noble, btlejuice to start proxy testing directly [1][11][13]. They are totally open-source projects which can be developed. There are also essential tools like bluez, hcitool, gattool to create custom Bluetooth proxy code [14][15][16].

Bluetooth Low Energy is one of the most popular technologies in IoT world. Wearables, smartphones, smartwatches, car locks, smart locks and smart home gadgets are bonded with BLE. In Bluetooth GATT (Generic Attribute Profile) protocol, services and characteristics are introduced in order to improve security [2][3][4][5]. Nothing to else, this protocol can easily be compromised by cloning services and characteristics [6][7]. There are many attacks scenarios using this attack. Passwords, authentications tokens can be stolen.

Denial of service attacks and eavesdropping attacks are possible. Battery consumption is the biggest risk in BLE devices. Fuzzing attacks are able to crash system. Attacker can send malformed data.

However, there are some difficulties about this attack. Some devices have costume connection protocol. In one of our tests, a device was connecting to a mobile application without pairing with the phone. This Bluetooth device was dropping a connection in five minutes. To be able to handle this kind of situations, custom proxy servers should be developed.

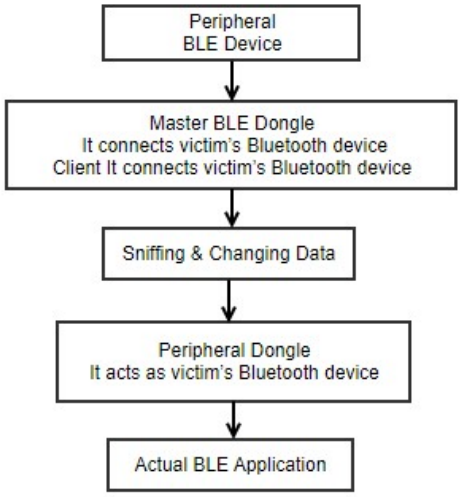


Fig. 5: BLE Hacking

The actual peripheral BLE device should be connected to fake master BLE dongle. Actual BLE application, should also connect to false peripheral BLE dongle. Proxy between BLE dongles can be sniffed.

V. Sniffing Bluetooth Data With Android Phone

In mobile phones with android operating system, there is a build in feature that you can capture Bluetooth connection log data. To have it, developer settings should be activated firstly. And then Bluetooth HCI capture option should be activated. Next, Bluetooth application can be used to capture some data. After that a file named btsnoop_hci.log should be found in storage. Phones have different destinations for his. A command can help to find its location. You can use this code from a terminal emulator in the phone or android adb command can be used to access terminal: "# find / -name btsnop_hci.log". After this part, downloading this file to a computer is required to open in Wireshark which is a packet capturing and analyzing program. Below code can be used to pull log folder to a computer. Android adb tool is required for that: "# adb pull /path/btsnop_hci.log".

Wireshark is a handy tool to explore Bluetooth data packages. It is possible to see many data as: mac address, Bluetooth adapter names, BLE commands (Read, Write, Notify, and Indicate), characteristic UUID information, and service UUID information.

To be able to improve this data, an android debug application was developed. In this application, Bluetooth log data is collected and devices from this data are listed. This application was built for Bluetooth Low Energy protocol, so services, characteristics and data are stored. It is possible to connect found devices again and send sniffed connection data.

There are many BLE services which defined in Bluetooth official website, however many are these services are custom made. BLE protocol sends raw binary data. Solving the meaning of this data can be challenging sometimes.

To understand meaning of the binary raw data, actual BLE mobile application is decompiled. Jadx-gui software was used for decompile process. Then decompiled project imported back to Android Studio development environment. This is helpful for obfuscated java codes to reach correct references of variables. When java code obfuscated, all variable names changed to letters.

Variable type names were not obfuscated. "UUID" variable type searched as string in code to find something belongs to services, characteristics or writable data. A service name and two characteristics are found using mobile application reverse engineering.

There were many log strings which stay still against obfuscations. From these log strings we understand a little bit about the process.

VI. Conclusion

In this work, during the BLE devices security test and evaluation steps are explained as flow charts. It is observed that BLE devices can be compromised as attacks like MITM, DOS, Replay; when an attacker able to enter its range. It is also seen that enhanced security protocols improves security levels. This work shows that BLE communication protocol is not enough for secure the devices by itself.

Acknowledgements

This work was made by workers and researchers of IoT Test and Evaluation Center (ISTEC) which supported by Istanbul Development Agency (İSTKA).

References

[1] ec.europa.eu, 2019. [Online]. Available: https://ec.europa.eu/eip/ageing/standards/home/indoor-localization/ieee-802151_en. [Accessed: 30- Sep- 2019].

[2] Gattack.io, 2019. [Online]. Available: http://gattack.io/whitepaper.pdf. [Accessed: 30- Sep- 2019].

[3] Generic Access Profile assigned numbers, Bluetooth SIG, [Online]. Available:https://www.bluetooth.org/enus/specification/assigned-numbers/generic-accessprofile [Accessed: 30- Sep- 2019].

[4] Bluetooth GATT specification, Bluetooth SIG, [Online]. Available: https://www.bluetooth.com/specifications/generic -attributes-overview [Accessed: 30- Sep- 2019].

[5] Bluetooth GATT Services specification, Bluetooth SIG, [Online]. Available: https://developer.bluetooth.org/gatt/services/Pages/ServicesHome.aspx. [Accessed: 30- Sep- 2019].

[6] GATT descriptors specification, Bluetooth SIG, [Online]. Available: https://developer.bluetooth.org/gatt/descriptors/Pages/DescriptorsHomePage.aspx [Accessed: 30- Sep- 2019].

[7] Bluetooth Smart Security, Bluetooth SIG, [Online]. Available: https://developer.bluetooth.org/TechnologyOverview/Pages/LE-Security.aspx. [Accessed: 30- Sep- 2019].

[8] M. Ryan, „Crackle - cracking Bluetooth Smart encryption," [Online]. Available: http://lacklustre.net/projects/crackle/ [Accessed: 30- Sep- 2019].

[9] Ubertooth One, Great Scott Gadgets, [Online]. Available: https://greatscottgadgets.com/ubertoothone/ [Accessed: 30- Sep- 2019].

[10] Bluefruit LE sniffer, Adafruit, [Online]. Available: https://www.adafruit.com/product/2269 [Accessed: 30- Sep- 2019].

[11] Bluetooth Smart community forums: random function," BlueGiga, [Online] Available: https://bluegiga.zendesk.com/entries/59399217-Random-function [Accessed: 30- Sep- 2019].

[12] Web Bluetooth, [Online], Available: http://webbluetoothcg.github.io/web-bluetooth [Accessed: 30- Sep- 2019].

[13] Mohamed Imran Jameel, Jeffrey Dungen, Low-Power Wireless Advertising Software Library for Distributed M2M and Contextual IoT http://reelyactive.com/science/reelyActiveIoT2015.pdf [Accessed: 30- Sep- 2019].

[14] Sandeep Mistry, A Node.js BLE (Bluetooth Low Energy) central module https://github.com/sandeepmistry/noble [Accessed: 30- Sep- 2019].

[15] [Online], Available: https://github.com/securing/gattacker [Accessed: 30- Sep- 2019].

[16] [Online], Available: https://www.npmjs.com/package/gattacker [Accessed: 30- Sep- 2019].

[17] [Online], Available: https://github.com/conorpp/btproxy [Accessed: 30- Sep- 2019].