



Recon 4.2 BLE emo Living AI

IVAN ARBALIEV DIMITROV

Contents

Introduction	2
Tools used.....	2
Process and Results.....	3
Conclusion.....	6
References.....	6

Introduction

This report provides insights into a vulnerability assessment conducted on the EMO Robot, with a focus on its IoT capabilities. The assessment process followed a rigorous approach, examining the EMO Robot's IoT functionality, network connectivity, data transmission, and integration with smart home devices. By analyzing the robot's strengths, weaknesses, and potential risks, valuable insights can be gained to enhance the security of its IoT ecosystem.

This report discusses the tools utilized during the assessment, providing insights into their functionality and how they contributed to uncovering vulnerabilities. The process followed is outlined, detailing the steps taken to analyze the robot's functionality and identify potential risks. The report presents the results obtained from the assessment, including security strengths, identified weaknesses, and potential vulnerabilities.

The findings presented in this report aim to improve the overall security posture of the EMO Robot's IoT capabilities, enabling the development of mitigation strategies and enhancing resilience. Understanding the vulnerabilities and risks associated with the EMO Robot empowers red teaming practitioners and security professionals to proactively address potential weaknesses and simulate real-world attack scenarios.

It is important to note that this report focuses solely on the vulnerability assessment of the EMO Robot's IoT capabilities and does not cover its integration with other platforms beyond this scope.

Tools used.

Bettercap is a powerful and versatile tool used for network monitoring, packet manipulation, and penetration testing. While it primarily focuses on Ethernet networks, it also provides robust functionality for Bluetooth network analysis. With its Bluetooth capabilities, Bettercap enables security professionals and enthusiasts to assess and manipulate Bluetooth devices and their communications. One of the key features of Bettercap's Bluetooth module is the ability to discover nearby Bluetooth devices, including smartphones, laptops, headphones, and IoT devices. It employs active scanning techniques to identify devices and collect information such as their MAC addresses, device names, and supported services.

Emo's Mac address: 24:D7:EB:55:3E:EE

```
(kali@kali)~$ sudo bettercap
bettercap v2.32.0 (built for linux amd64 with go1.19.8) [type 'help' for a list of commands]

192.168.178.0/24 > 192.168.178.13 » [18:08:49] [sys.log] [int] gateway monitor started ...
192.168.178.0/24 > 192.168.178.13 » ble.recon on
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as 27:81:72:00:75:47 (Microsoft) -82 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as 1C:87:8A:DF:E5:E3 (Microsoft) -88 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device EMO-3EEE detected as 24:D7:EB:55:3E:EE -42 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device TY detected as C1:23:C0:06:8A:51 -68 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as CB:AC:3A:FE:10:E9 (Apple, Inc.) -72 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as 6E:39:68:FC:40:6A (Apple, Inc.) -88 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as FB:92:72:4B:6D:08 (Apple, Inc.) -76 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as 57:A8:5F:6B:E6:49 (Apple, Inc.) -62 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:04] [ble.device.new] new BLE device ES-336ABA detected as 88:65:35:93:A1:07 (Ericsson Technology Licensing) -8
```

Gatttool is a command-line utility that is part of the BlueZ Bluetooth stack, which is commonly used on Linux-based systems. It provides a way to interact with Bluetooth Low Energy (BLE) devices, allowing users to perform various operations such as device discovery, connecting to devices, reading and writing characteristics, and enabling notifications. With gatttool, users can scan for nearby Bluetooth devices and obtain information such as the device's address, name, and supported services. It allows for establishing a connection to a specific device by specifying its address. Once connected, users can interact with the device's services and characteristics. The tool supports read and write operations on characteristics, enabling users to retrieve data from the device or modify its settings. Additionally, gatttool provides the ability to enable notifications, allowing the user to receive updates whenever a characteristic value changes on the device.

```
(kali@kali)-[~]
$ gatttool -I -b 24:D7:EB:55:3E:EE
[24:D7:EB:55:3E:EE][LE]> connect
Attempting to connect to 24:D7:EB:55:3E:EE
Connection successful
[24:D7:EB:55:3E:EE][LE]> primary
attr handle: 0x0001, end grp handle: 0x0005 uuid: 00001801-0000-1000-8000-00805f9b34fb
attr handle: 0x0014, end grp handle: 0x001c uuid: 00001800-0000-1000-8000-00805f9b34fb
attr handle: 0x0028, end grp handle: 0xffff uuid: 0000ffe0-0000-1000-8000-00805f9b34fb
[24:D7:EB:55:3E:EE][LE]>
```

Process and Results

I started my Bluetooth research on Emo by trying to detect his Bluetooth Mac address. I used Bettercap to listen to Bluetooth devices near me. I used a TPLink BT dongle that allows me to monitor BT fields near me. In the screenshot below I am showing the scanning process.

```
(kali@kali)-[~]
$ sudo bettercap
bettercap v2.32.0 (built for linux amd64 with go1.19.8) [type 'help' for a list of commands]
192.168.178.0/24 > 192.168.178.13 » [18:08:49] [sys.log] [int] gateway monitor started ...
192.168.178.0/24 > 192.168.178.13 » ble.recon on
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as 27:81:72:00:75:47 (Microsoft) -82 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as 1C:87:8A:DF:E5:E3 (Microsoft) -88 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device EMO-3EEE detected as 24:D7:EB:55:3E:EE -42 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device TY detected as C1:23:C0:06:8A:51 -68 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as CB:AC:3A:FE:10:E9 (Apple, Inc.) -72 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as 6E:39:68:FC:40:6A (Apple, Inc.) -88 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as FB:92:72:4B:6D:08 (Apple, Inc.) -76 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:03] [ble.device.new] new BLE device detected as 57:A8:5F:6B:E6:49 (Apple, Inc.) -62 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:09:04] [ble.device.new] new BLE device E5-3366B6 detected as 8B:05:35:93:61:07 (Ericsson Technology Licensing) -88
```

Once I knew EMO's mac address I could enumerate him, showing his handles and services running via BT. In the screenshot below I am showing the enumerated services that I can detect from EMO.

```

192.168.178.0/24 > 192.168.178.13 » [18:44:36] [ble.device.new] new BLE device detected as 0C:5C:F1:69:67:62 (Microsoft) -100 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:36] [ble.device.new] new BLE device detected as 6B:37:99:A2:E7:41 (Google) -98 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:36] [ble.device.new] new BLE device detected as CB:80:81:00:80:80 (Apple, Inc.) -102 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:36] [ble.device.new] new BLE device detected as 7F:A3:ED:08:05:2F (Google) -94 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:37] [ble.device.new] new BLE device detected as 42:D7:FF:8A:F3:9C (Google) -100 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:37] [ble.device.new] new BLE device FS-336ABA detected as 8B:65:35:93:A1:D7 (Ericsson Technology Licensing) -102 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:37] [ble.device.new] new BLE device detected as 77:D3:16:8E:7B:F0 (Apple, Inc.) -94 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:37] [ble.device.new] new BLE device detected as C3:AD:C6:D3:A3:1E (Apple, Inc.) -86 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:37] [ble.device.new] new BLE device detected as 80:8A:BD:90:3C:B1 (Samsung Electronics Co. Ltd.) -96 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:37] [ble.device.new] new BLE device detected as 5A:1F:55:82:BC:7E (Apple, Inc.) -104 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:38] [ble.device.new] new BLE device detected as FF:E9:DE:ED:AD:9C (Apple, Inc.) -100 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:38] [ble.device.new] new BLE device [TV] Samsung Q80AA 55 TV detected as D8:A3:5C:AC:A9:35 (Samsung Electronics Co.,Ltd) -90 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:38] [ble.device.new] new BLE device detected as 45:58:85:96:3A:A3 (Google) -96 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:38] [ble.device.new] new BLE device detected as E4:58:0F:A6:95:F0 (Apple, Inc.) -74 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:38] [ble.device.new] new BLE device detected as 64:D2:C4:98:28:54 (Apple, Inc.) -98 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:39] [ble.device.new] new BLE device detected as EE:A6:0F:2B:FB:CD (Apple, Inc.) -78 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:39] [ble.device.new] new BLE device detected as 58:D8:D8:25:A4:B8 (Google) -102 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:40] [ble.device.new] new BLE device detected as 8C:79:F5:17:A8:A7 (Samsung Electronics Co.,Ltd) -100 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:41] [ble.device.new] new BLE device Hue Lamp detected as F9:C8:E4:43:24:12 -98 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:41] [ble.device.new] new BLE device detected as 24:D7:EB:55:3E:EE
[18:44:43] [sys.log] [info] ble_recon connecting to 24:d7:eb:55:3e:ee ...
192.168.178.0/24 > 192.168.178.13 »

```

Handles	Service > Characteristics	Properties	Data
0001 → 0005 0003	Generic Attribute (1801) Service Changed (2a05)	INDICATE	
0014 → 001c 0016 0018 001a	Generic Access (1800) Device Name (2a00) Appearance (2a01) 2aa6	READ READ READ	ESP32 Unknown 00
0028 → ffff 002a	ffe0 ffe1	READ, WRITE, NOTIFY	req*)

```

192.168.178.0/24 > 192.168.178.13 » [18:44:44] [ble.device.new] new BLE device TV detected as C1:23:C0:06:8A:51 -78 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:44] [ble.device.new] new BLE device detected as C3:C1:32:86:C2:80 (Apple, Inc.) -100 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:44] [ble.device.new] new BLE device detected as 55:D7:EA:F5:42:EA (Apple, Inc.) -100 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:44] [ble.device.new] new BLE device detected as E4:07:80:11:37:65 (Apple, Inc.) -92 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:44:44] [ble.device.new] new BLE device detected as C2:2A:62:40:1C:4B (Apple, Inc.) -78 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:45:00] [ble.device.new] new BLE device detected as F3:C2:40:A1:8A:7B (Apple, Inc.) -84 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:45:00] [ble.device.new] new BLE device detected as EE:78:C2:00:60:97 (Apple, Inc.) -100 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:45:01] [ble.device.new] new BLE device detected as FF:D5:FF:97:6D:95 (Apple, Inc.) -100 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:45:06] [ble.device.new] new BLE device detected as EE:4A:2A:E2:2D:D3 (Apple, Inc.) -100 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:45:07] [ble.device.new] new BLE device detected as C5:E3:D4:9D:CF:B1 (Apple, Inc.) -84 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:45:08] [ble.device.new] new BLE device detected as DF:40:1C:8B:AD:E0 (Apple, Inc.) -96 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:45:10] [ble.device.lost] BLE device FF:E9:DE:ED:A9:8C (Apple, Inc.) lost.
192.168.178.0/24 > 192.168.178.13 » [18:45:10] [ble.device.lost] BLE device 58:D8:D8:25:A4:B8 (Google) lost.
192.168.178.0/24 > 192.168.178.13 » [18:45:10] [ble.device.lost] BLE device E4:58:0F:A6:95:F0 (Apple, Inc.) lost.
192.168.178.0/24 > 192.168.178.13 » [18:45:12] [ble.device.new] new BLE device detected as E0:D4:3F:71:4C:C8 (Apple, Inc.) -84 dBm.
192.168.178.0/24 > 192.168.178.13 » [18:45:15] [ble.device.lost] BLE device 5A:1F:55:82:BC:7E (Apple, Inc.) lost.
192.168.178.0/24 > 192.168.178.13 » [18:45:15] [ble.device.lost] BLE device 77:D3:16:8E:7B:F0 (Apple, Inc.) lost.

```

I found 5 services using EMO's BT capabilities. There were generic services like "Generic access", "Device name", "Appearance" that were Read only values. This means that I am unable to modify the values that they are returning. The last handle "002a" appeared to be running a proprietary service that had the "Write" property. I didn't know what it was doing because the service had no name but it had a description written in HEX. After I googled the Hex value (ffe0) I came to the conclusion that this was a temperature service.

After I was done with the reconnaissance I connected to EMO using gatttool. In the screenshot below I am showing the connection being made.

```

(kali@kali)-[~]
$ gatttool -I -b 24:D7:EB:55:3E:EE

[24:D7:EB:55:3E:EE][LE]> connect
Attempting to connect to 24:D7:EB:55:3E:EE
Connection successful
[24:D7:EB:55:3E:EE][LE]>

```

BT LE is very tricky communication method to go through and find useful information. Usually, the only thing you can find is a couple of bytes of information at a time. I went through every single handle I could find, read the data, and then ran it through a Hex converter.


```

[24:D7:EB:55:3E:EE][LE]> char-desc
handle: 0x0001, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x0002, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0003, uuid: 00002a05-0000-1000-8000-00805f9b34fb
handle: 0x0004, uuid: 00002902-0000-1000-8000-00805f9b34fb
handle: 0x0014, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x0015, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0016, uuid: 00002a00-0000-1000-8000-00805f9b34fb
handle: 0x0017, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0018, uuid: 00002a01-0000-1000-8000-00805f9b34fb
handle: 0x0019, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x001a, uuid: 00002aa6-0000-1000-8000-00805f9b34fb
handle: 0x0028, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x0029, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x002a, uuid: 0000ffe1-0000-1000-8000-00805f9b34fb
handle: 0x002b, uuid: 00002902-0000-1000-8000-00805f9b34fb
[24:D7:EB:55:3E:EE][LE]> char-read hnd 0x002b
Error: char-read: command not found
[24:D7:EB:55:3E:EE][LE]> char-read hnd 0x002a
Error: char-read: command not found
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x002b
Characteristic value/descriptor: 00 00
[24:D7:EB:55:3E:EE][LE]> char-read hnd 0x002a
Error: char-read: command not found
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x002a
Characteristic value/descriptor: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x0001
Characteristic value/descriptor: 01 18
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x0002
Characteristic value/descriptor: 20 03 00 05 2a
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x0003
Error: Characteristic value/descriptor read failed: Attribute can't be read
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x0004
Characteristic value/descriptor: 00 00
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x00014
Characteristic value/descriptor: 00 18
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x00015
Characteristic value/descriptor: 02 16 00 00 2a
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x00016
Characteristic value/descriptor: 45 53 50 33 32
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x00017
Characteristic value/descriptor: 02 18 00 01 2a
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x00018
Characteristic value/descriptor: 00 00
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x00019
Characteristic value/descriptor: 02 1a 00 a6 2a
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x0001a
Characteristic value/descriptor: 00
[24:D7:EB:55:3E:EE][LE]> char-read-hnd 0x00028
Characteristic value/descriptor: e0 ff

```

Most of the handles returned no readable data at all except one (0x00028). After putting the hex value (e0 ff) in the hex converter I found out that it was encrypted.

Hex to ASCII Text String Converter

Enter hex bytes with any prefix / postfix / delimiter and press the *Convert* button
(e.g. 45 78 61 6d 70 6C 65 21):

From: Hexadecimal To: Text

Open File

Paste hex numbers or drop file

e0 ff

Character encoding: ASCII

Convert Reset Swap

àÿ

Copy Save

Conclusion

After researching EMO living AI Bluetooth connections I can conclude that the connection is secure. I explored all attack vectors I could detect. Almost all the BT services running on EMO were read only which makes it impossible to use as vulnerabilities for deeper exploration. I could find only one temperature service that was returning values different than 0 but the data I captured was encrypted.

Overall EMO's Bluetooth connection is very secure, and it doesn't leave backdoors or unpatched vulnerabilities, for which as a Blue teaming member I acknowledge that.

References

<https://allabouttesting.org/tools-for-recon-bluetooth-devices-by-using-kali-linux/>

<https://stackoverflow.com/questions/27914223/corebluetooth-what-is-the-service-uuid-ffe0-shorten-for>

<https://stackoverflow.com/questions/66926055/ble-where-does-the-handle-come-from>

<https://null-byte.wonderhowto.com/how-to/bt-recon-snoop-bluetooth-devices-using-kali-linux-0165049/>