DEMO: BTLEmap: Nmap for Bluetooth Low Energy

Alexander Heinrich
Secure Mobile Networking Lab
Department of Computer Science
TU Darmstadt, Germany
aheinrich@seemoo.de

Milan Stute
Secure Mobile Networking Lab
Department of Computer Science
TU Darmstadt, Germany
mstute@seemoo.de

Matthias Hollick
Secure Mobile Networking Lab
Department of Computer Science
TU Darmstadt, Germany
mhollick@seemoo.de

ABSTRACT

The market for Bluetooth Low Energy (BLE) devices is booming and, at the same time, has become an attractive target for adversaries. To improve BLE security at large, we present BTLEmap, an auditing application for BLE environments. BTLEmap is inspired by network discovery and security auditing tools such as Nmap for IP-based networks. It allows for device enumeration, Generic Attribute Profile (GATT) service discovery, and device fingerprinting. It also features a BLE advertisement dissector, data exporter, and a user-friendly UI including a proximity view. BTLEmap currently runs on iOS and macOS using Apple's CoreBluetooth API but also accepts alternative data inputs such as a Raspberry Pi to overcome the restricted vendor API. The open-source project is under active development and will provide more advanced capabilities such as long-term device tracking (in spite of MAC address randomization) in the future.

CCS CONCEPTS

• **Security and privacy** → **Mobile and wireless security**; Software reverse engineering; • **Software and its engineering** → *Software creation and management.*

KEYWORDS

Bluetooth Low Energy, privacy analysis, device enumeration, network monitoring, reverse engineering

ACM Reference Format:
Alexander Heinrich, Milan Stute, and Matthias Hollick. 2020. DEMO: BTLEmap: Nmap for Bluetooth Low Energy. In *13th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '20), July 8–10, 2020, Linz (Virtual Event), Austria.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3395351.3401796

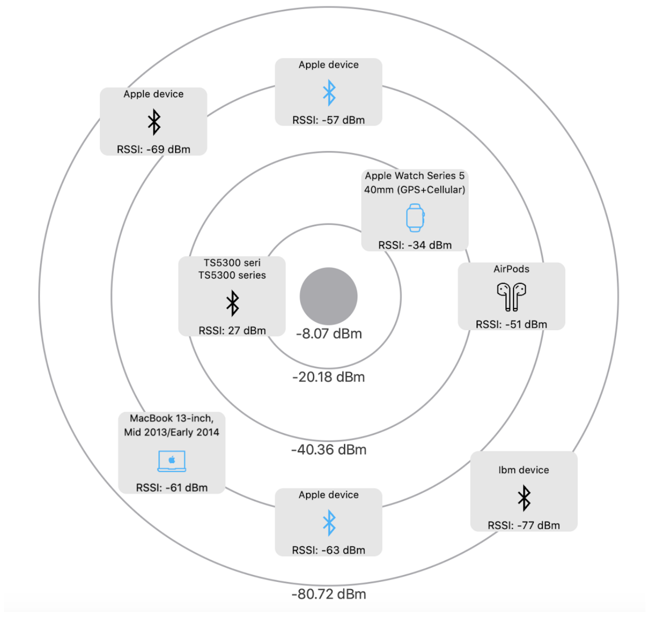


Figure 1: BTLEmap's proximity view

1 INTRODUCTION AND BACKGROUND

Since the introduction of Bluetooth Low Energy (BLE) in 2010 [1], the technology has become widely adopted in smartphones, wearables, and other IoT devices. In a modern household, there can be more than a dozen devices that supporting this technology. They are constantly sending BLE advertisements to inform devices in their surrounding about their presence but typically without the owner noticing. Most recently, BLE has been proposed to be used for contact tracing during the COVID-19 pandemic. Most of the tracing solutions use BLE advertisements containing a custom identification token that can be linked to an infected person.

BLE advertisements pose a vast surface for privacy-compromising attacks. In the past, several researchers found that BLE advertisements may contain fixed identifiers that allow for device tracking despite MAC address randomization [6] and could contain personally identifiable information about the user such as their phone number or email address [9]. Besides, an adversary might place a tracking device based on BLE, such as the rumored Apple *AirTags* [4], in a person's pocket and leverage a crowd-sourced finder network to track their target. In essence, BLE devices share potentially sensitive data with devices in proximity. To improve our understanding of the (privacy-related) attack surface at large, e. g., by analyzing privacy leaks or detecting malicious devices, we as a security community require better application support.

We propose BTLEmap (pronounce as *beetle map*), a network discovery and security auditing tool in the spirit of Nmap [5] but for BLE environments. Currently, BTLEmap supports device enumeration, advertisement dissection, rudimentary device fingerprinting, a proximity view, and more. With its extensible and modular design, it will support more advanced features such as long-term device tracking and comprehensive fingerprinting capabilities in the future.

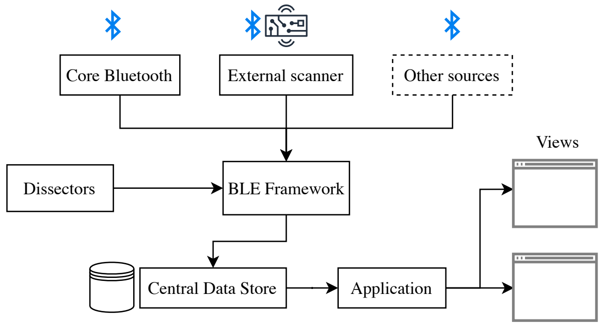


Figure 2: BTLEmap's software architecture

We make BTLEmap for iOS and macOS available as open-source software on GitHub and distribute the application bundle for iOS and macOS via Apple's official App Store for free.

2 BTLEMAP

Our initial version of BTLEmap runs on iOS and macOS using SwiftUI and macOS Catalyst to enable rich animations combined with a unified user experience across multiple platforms. Our tool should be the first approach to analyze surroundings for BLE devices and for researchers who want to discover new targets that should be analyzed. It automatically scans the environment for BLE devices, lists their advertisements, connects to them to detect the supported services. All received advertisements are stored for long term analysis and can be exported.

Our design favors modularity, as shown in Fig. 2. At its core, BTLEmap uses a data store for recording advertisements. It supports both internal and external input sources (more in Section 2.4) and an extensible dissector module that already implements multiple dissectors for Apple's BLE advertisements. Finally, different views, such as a proximity view (Section 2.1), present the advertisement data in different ways. In the following, we highlight some of BTLEmap's features, in particular, the proximity view, the advertisement dissector, the automated device detection, and support for external scanning sources. Also, we discuss areas for future work.

2.1 Proximity view

Using the central data store, BTLEmap can create different representations of the BLE readings. Figure 1 shows a proximity view that estimates the distance to devices by using the Received signal strength indication (RSSI) values in combination with the advertised transmission power of each advertisement that has been received. The scanner can be used to discover devices in proximity quickly and to estimate how far away devices are. Even though RSSI values cannot be used to measure an exact distance, it is possible to estimate a relative distance between the devices shown on the circular plane. At the moment, the angles shown are random because the current chips and APIs do not allow to measure the angle-of-arrival (AoA). As this feature is part of the current Bluetooth Core Specification [2, 1.A.8], we plan to implement it as soon as AoA becomes available to the public.

2.2 Advertisement dissector

BLE advertisements contain up to 31 bytes of information about the emitting device or its services. Existing BLE scanners such as Nordic's nRF Connect [7] display this information as a simple byte string as there is no standard on what information can be encoded or how it is encoded. BTLEmap leverages the reverse engineering efforts of several researcher teams [3, 6, 9] to present the user with additional information about the state of the device. For example, Apple AirPods indicate the model name, colors, and battery status. We depict our Wireshark-inspired dissector view in Fig. 4.

2.3 Automated device detection

Apart from analyzing BLE advertising packets, BTLEmap has the option to automatically connect to all devices in range, query the supported services, and read the values of characteristics. This results in automated device detection, listing the manufacturer, device type, device names, and more. Many users tend to rename their Bluetooth enabled speakers or headphones, which can result in a privacy leak and tracking possibilities because BLE often allows unauthenticated connections. BTLEmap can be used to make such issues visible for researchers and for non-scientific users.

2.4 External scanning sources

The iOS and macOS APIs for BLE scanning are limited. We have found that the *CoreBluetooth* API on iOS does not report any manufacturer data that starts using Apple's company identifier `0x4c00` (little-endian). On macOS, the manufacturer data is not stripped, but the API will not retrieve all Generic Attribute Profile (GATT) services supported by a device. The frameworks strip away Apple-specific services that should not be accessible from other devices. Furthermore, it is not possible to retrieve a device's (randomized) BLE MAC address, because *bluetoothd* replaces all MAC addresses with an on-demand generated UUID.

These inherent restrictions would limit the usefulness of BTLEmap. Therefore, we support external Bluetooth scanner inputs such as a Raspberry Pi (3 or newer/Zero W). The external device implements a simple Bluetooth scanner and forwards all received advertisements and discovered services in real-time to a connected macOS or iOS device. The Raspberry Pi connects over a direct Ethernet connection or by using Ethernet-over-USB to receive power and set up a connection, as shown in Fig. 3. BTLEmap announces itself as services via multicast DNS such that the Raspberry Pi can automatically connect to it. The modular setup allows to support most functionality on the device running the app, but increase the feature spectrum by adding additional hardware.

2.5 Additional features

BTLEmap currently supports several additional features: (1) An RSSI graph displaying all devices and the received RSSI values at a specific time. (2) An RSSI recorder that allows recording RSSI readings and exporting them to CSV files later. (3) Several filters, such as manufacturer or RSSI. (4) Highlighting the recently active devices. (5) Import and export of `pcap` files.

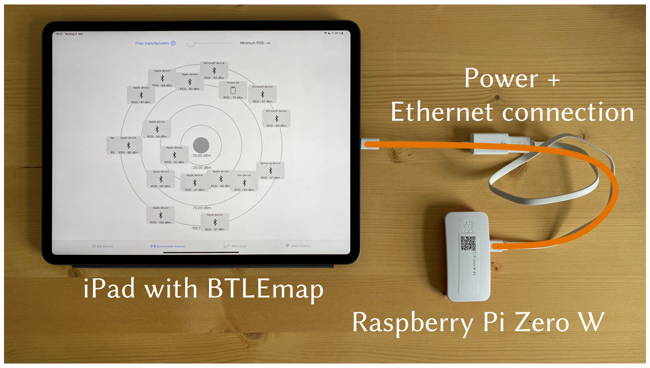
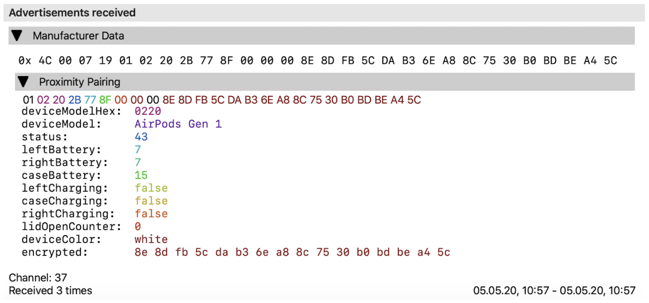


Figure 3: BTLEmap setup with an external scanner source



```
Advertisements received
▼ Manufacturer Data
0x 4C 00 07 19 01 02 20 2B 77 8F 00 00 00 8E 8D FB 5C DA B3 6E A8 8C 75 30 B0 BD BE A4 5C
▼ Proximity Pairing
01 02 20 2B 77 8F 00 00 00 8E 8D FB 5C DA B3 6E A8 8C 75 30 B0 BD BE A4 5C
deviceModelHex:  0220
deviceModel:     AirPods Gen 1
status:          43
leftBattery:     7
rightBattery:    7
caseBattery:     15
leftCharging:    false
caseCharging:    false
rightCharging:   false
lidOpenCounter:  0
deviceColor:     white
encrypted:       8e 8d fb 5c da b3 6e a8 8c 75 30 b0 bd be a4 5c

Channel: 37
Received 3 times                                   05.05.20, 10:57 - 05.05.20, 10:57
```

Figure 4: Decoding of an BLE advertisement emitted by Apple *AirPods* earphones

2.6 Future work

At the moment, we are fine-tuning our implementation to be ready for a public release. Additionally, we work on algorithms that allow further automatic analysis of BLE advertisements by identifying "trackable" identifiers, thus, revealing potentially private data.

We started with macOS and iOS, because it allows us to combine BLE with Apple's custom protocol Apple Wireless Direct Link (AWDL) [8]. Apple devices always send out BLE advertisements before they setup a peer-to-peer connection over AWDL, e. g., for exchanging files. In the future we want to visualize those connections over AWDL by monitoring both interfaces in parallel.

If Bluetooth chips adopt AoA support and it becomes available through public APIs, BTLEmap could leverage the AoA of wireless signals to align devices in the proximity view in Section 2.1 with actual angles. Finally, as our core framework is written in Swift, portability to other platforms such as Linux is possible with some caveats.

3 DEMONSTRATION SETUP

Our demonstration provides every participant with an application binary running on macOS and gives them the ability to join a TestFlight Beta for an iOS build. We provide an introduction to BTLEmap and present the available features during a live session or via a pre-recorded video. The demonstration starts with the general scanning and dissection features: we use Apple AirPods advertisements to detect the device model and the state, as shown in Fig. 4. We also demonstrate Apple AirDrop advertisements. We continue the demonstration by showing the distance estimations when devices are moving around the scanning device. Then we use the scanning device to find hidden BLE devices. Finally, we show how to enumerate devices in combination with a Raspberry Pi as an external scanning source. Participants can test the software immediately and report any recommendations for future releases on GitHub.

AVAILABILITY

BTLEmap is available open-source on https://github.com/seemoo-lab/BTLEmap. All necessary frameworks are also published on our GitHub account. Furthermore, the app will be made available on TestFlight as soon as possible. After our Beta test, we will distribute BTLEmap via the official Apple App Store for free. It allows other researchers without a development background to get easy access to our tool and it includes non-scientific users that are curious about their BLE environment.

ACKNOWLEDGEMENTS

This work has been funded by the German Federal Ministry of Education and Research and the Hessen State Ministry for Higher Education, Research and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

REFERENCES

[1] Bluetooth SIG 2010. *Specification of the Bluetooth System.* Bluetooth SIG. Vol 0.
[2] Bluetooth SIG 2019. *Bluetooth Core Specification.* Bluetooth SIG. Vol 0.
[3] Guillaume Celosia and Mathieu Cunche. 2020. Discontinued Privacy: Personal Data Leaks in Apple Bluetooth-Low-Energy Continuity Protocols. *Proceedings on Privacy Enhancing Technologies* 2020, 1 (Jan. 2020), 26–46. https://doi.org/10.2478/popets-2020-0003
[4] Juli Clover. 2020. *AirTags: Everything We Know So Far.* MacRumors.com LLC. https://www.macrumors.com/guide/airtags/
[5] Gordon Lyon. 1997. *Nmap: the Network Mapper.* Insecure.Com LLC. https://nmap.org
[6] Jeremy Martin, Douglas Alpuche, Kristina Bodeman, Lamont Brown, Ellis Fenske, Lucas Foppe, Travis Mayberry, Erik Rye, Brandon Sipes, and Sam Teplov. 2019. Handoff All Your Privacy – A Review of Apple's Bluetooth Low Energy Continuity Protocol. *Proceedings on Privacy Enhancing Technologies* 2019, 4 (Oct. 2019), 34–53. https://doi.org/10.2478/popets-2019-0057
[7] Nordic Semiconductor. 2016. *nRF Connect for Mobile.* Nordic Semiconductor ASA. https://www.nordicsemi.com/Software-and-tools/Development-Tools/nRF-Connect-for-mobile
[8] Milan Stute, David Kreitschmann, and Matthias Hollick. 2018. One Billion Apples' Secret Sauce: Recipe for the Apple Wireless Direct Link Ad hoc Protocol. *CoRR* abs/1808.03156 (2018). arXiv:1808.03156 http://arxiv.org/abs/1808.03156
[9] Milan Stute, Sashank Narain, Alex Mariotto, Alexander Heinrich, David Kreitschmann, Guevara Noubir, and Matthias Hollick. 2019. A Billion Open Interfaces for Eve and Mallory: MitM, DoS, and Tracking Attacks on iOS and macOS Through Apple Wireless Direct Link. In *28th USENIX Security Symposium (USENIX Security 19).* USENIX Association, Santa Clara, CA, 37–54. https://www.usenix.org/conference/usenixsecurity19/presentation/stute