BL(u)E CRAB: A User-Centric Framework for Identifying Suspicious Bluetooth Trackers

Dylan Conklin
Portland State University
Portland, OR USA
dconklin@pdx.edu

Primal Pappachan
Portland State University
Portland, OR USA
primal@pdx.edu

Roberto Yus
University of Maryland, Baltimore County
Baltimore, MD USA
ryus@umbc.edu

***Abstract*—Given the pervasiveness of Bluetooth Low Energy (BLE)-based devices, detecting unwanted or suspicious trackers is challenging, especially due to their heterogeneity, cross-platform compatibility issues, and inconsistent detection methods. BL(u)E CRAB identifies suspicious BLE trackers based on various risk factors within minutes. It does so by collecting information including the number of encounters, time with the user, distance traveled with the user, number of areas each device appeared in, and device proximity to user. After collecting this information, BL(u)E CRAB performs an outlier detection analysis to flag suspicious devices. BL(u)E CRAB presents this information in a simple, intuitive, and customizable way for the user to determine which devices pose the biggest threat to them based on their context.**

***Index Terms*—Bluetooth Low Energy (BLE), BLE trackers, Suspicious device detection, Context-aware monitoring**

I. Introduction

Devices that use Bluetooth Low Energy (BLE) surround us, including, among many others, smartphones, smartwatches, and audio devices. Among them, trackers are small, low-power devices that use a crowd-sourced model to communicate their location to a centralized server, relying on nearby higher-powered devices to relay location updates to a server. Users who own these trackers can view the tracker's location via an app or web service and interact with the device remotely. BLE trackers are commonly used to track a user's possessions for convenience and to counter theft. However, assailants, which may include friends, family, coworkers, and current or ex-romantic partners, can misuse these devices for stalking. Stalkers commonly use BLE trackers to stalk their victims by placing them in cars or backpacks[1] to receive updates on their victim's location, revealing their daily habits, where they live, and when they are likely to be alone.

Manufacturers have taken measures to prevent stalking via BLE trackers, such as alerting victims when a tracker is detected and installing speakers to help victims find it when detected. However, these features are sometimes limited to specific devices capable of running the manufacturer's apps. There has been some progress from the industry in developing protocols, such as Apple's Find My and Google's Find My Device protocols[2], which manufacturers can adopt to fight unwanted tracking. However, many other third-party services rely on manufacturer-specific networks that cannot adequately alert users of unwanted trackers in their vicinity, and these services often are incompatible with each other[1], as multiple manufacturers, devices, and services are available. Many of these services are also proprietary, so their methods of detecting trackers are not transparent. Stalking victims may be unaware of tracking unless or until their cellular device identifies the tag as a tracker, and these features are unavailable to victims until their device makes this identification. These incompatibilities are problematic because a stalker could use a tracker that would not be detected by the victim's phone, requiring the victim to have multiple smartphones from different manufacturers with many BLE tracker apps installed on each phone to detect unwanted trackers adequately.

AirGuard[3] and BLE-Doubt[4] do not require manufacturer participation and are transparent about how they decide whether or not a device is suspicious. While the behavior of suspicious devices changes with time and context, these apps rely on only a few metrics with fixed thresholds. AirGuard only uses fixed time and distance thresholds. In contrast, BLE-Doubt uses fixed time and distance thresholds using an approach to group data points based on time and categorize devices based on their trajectory. Evaluating a device's threat potential should consider a broader set of factors, and a flexible approach to establishing thresholds can provide better insight. BL(u)E CRAB is a decentralized and manufacturer-neutral approach that attempts to improve upon prior works by collecting several adaptable risk factors (outlined in Section II-B) that do not rely on fixed time and distance thresholds to show users why it flags specific devices as suspicious. It introduces an adaptable risk assessment framework that evaluates various risk factors relevant to the user's context and nearby devices. Our contributions include this framework and a mobile application that realizes this framework.

II. Overview of BL(u)E CRAB

Figure 1 provides an overview of BL(u)E CRAB's architecture. BL(u)E CRAB scans for Bluetooth Low Energy (BLE) devices in the vicinity of a user device (such as AirTag, SmartTag, and Tile) while recording the time, location, and signal strength for each tracker when they are detected. This information is stored on-device in a local database and

[1]For example, Tile devices only work with the Tile app and Chipolo devices only work with any of Apple's Find My, Google's Find My Device, or Chipolo's proprietary apps.

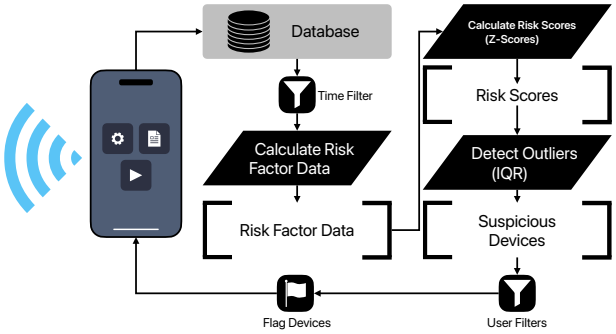

Fig. 1. Architecture of BL(u)E CRAB.

periodically retrieved with filters on time to obtain the most recent data. It then calculates the risk scores associated with each risk factor (as defined in Section II-B) for every device using outlier detection. BL(u)E CRAB marks devices with an outlier score in any metric as suspicious. Users can turn on or off various risk factors relevant to their context to filter the results, and BL(u)E CRAB displays details of any devices marked as suspicious based on the selected factors. BL(u)E CRAB relies on the Flutter SDK, which enables cross-platform deployment from a single codebase.

A. Scanning Method

BL(u)E CRAB collects data using the device's Bluetooth scanner and GPS location (if the user permits location scanning) by scanning for Bluetooth signals broadcast by surrounding devices. BL(u)E CRAB logs the transmitting device's MAC address and optional identifiers (services, manufacturers, and platform), the time the signal was received, the user's location, and the transmitting device's signal strength (RSSI). BL(u)E CRAB utilizes foreground scanning instead of background scanning because it helps to collect more accurate data by allowing for more frequent updates[2], at the expense of higher power consumption. While scanning, BL(u)E CRAB checks suspicious devices and filters stale data to prevent devices that do not change their identifiers over time from skewing risk scores toward flagging non-threatening devices.

B. Risk Factors

BL(u)E CRAB calculates risk scores based on factors derived from data collected during scanning to determine each device's likelihood of threatening the user. BL(u)E CRAB evaluates all these risk factors using a static threshold with a default value that the researchers experimentally determined, and the user can modify as necessary.

1) Time With User: Measures the sum of durations between each timestamp shorter than the threshold by sorting the data points associated with the device by time. BL(u)E CRAB removes each duration exceeding the time threshold because they indicate periods when it did not detect the device.

[2]Apple devices hide Find My devices scanned in the background[1], making data collection difficult for devices like AirTags.

2) Incidence: Counts the number of non-overlapping timestamp clusters where a device is detected. Any two consecutive timestamps whose difference is under the time threshold belong to the same cluster.

3) Distance Traveled: Measures the sum of the distances between locations where BL(u)E CRAB detects the device, provided the time difference between those two detections is under the threshold.

4) Area Count: Counts the amount of non-overlapping location clusters where a device is detected. Each cluster includes any two locations whose distance is under the threshold.

5) Device Proximity: Measures the scanned devices' average signal strength (RSSI) over time. Devices measure RSSI in dBm, with values close to 0 representing strong signals and below -90 representing weak signals.

C. Threat Potential Calculation

A device's threat potential derives from risk scores computed from the risk factors across all recently scanned devices. BL(u)E CRAB uses single-metric scores to compartmentalize risk based on a factor's context. BL(u)E CRAB evaluates the scores for each risk factor using the Z-score formula, $z_{rf} = \frac{x_{rf} - \mu_{rf}}{\sigma_{rf}}$, where x_{rf} is the factor value, μ_{rf} is the mean of the factor values across all devices, and σ_{rf} is the standard deviation of the factor values.

Using these risk scores, BL(u)E CRAB identifies outlier values by applying two multipliers (e.g., 1.5 and 3 in our demo) of the interquartile range ($IQR = Q_3 - Q_1$) above Q_3. Using the equation $threshold = Q_3 + (IQR + multiplier)$, mild outliers exceed the threshold for the smaller multiplier, and extreme outliers exceed the threshold for the greater multiplier. BL(u)E CRAB flags all devices with an outlier value, indicating a more significant threat for that risk factor.

III. Demo Description

This demonstration intends to showcase BL(u)E CRAB's capabilities in scanning and detecting unwanted BLE trackers in a user's vicinity. The demo scenario involves a smartphone installed with BL(u)E CRAB and multiple BLE-based trackers (such as AirTag, Tile, SmartTag, or Pebblebee). We will set up the app with sample scan data from BLE-Doubt[4] for users who wish to skip scanning and observe the metrics. The screenshots in the Figures below derive from a modified version of this dataset. The original dataset anonymized the devices, so we added device identifiers to differentiate between them. The BL(u)E CRAB source code and sample datasets are available on GitHub[3]. A brief video showcasing the functionality of BL(u)E CRAB can seen on YouTube[4]. The steps in the demonstration scenario are as follows:

1) Users start with the scanner view and can begin scanning by pressing the "Start Scanning" button (Figure 2). A notification appears once a suspicious device is detected, and BL(u)E CRAB displays the report view upon tapping the notification or the report button.

[3]https://github.com/DIPrLab/BLuE-CRAB
[4]https://youtu.be/J9vjPuSkJyU

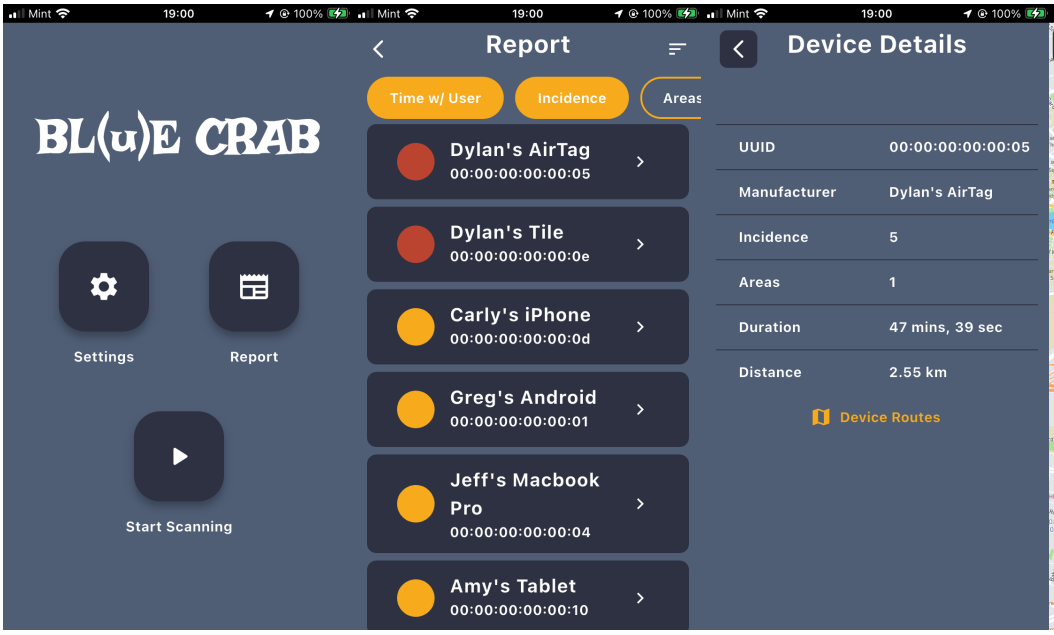

Fig. 2. Scanner view with the scan button at the bottom.

Fig. 3. Report view with tiles for flagged devices.

Fig. 4. Device detail view displaying identifiers along with risk factors.

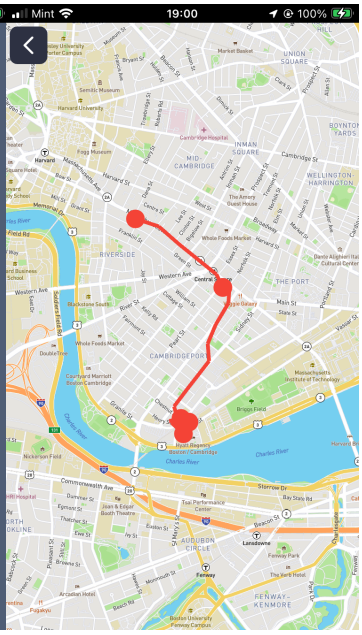

Fig. 5. Map of where the selected device traveled with the user.

2) At the top of the report view are filter buttons where users can toggle each risk factor based on their context (Figure 3). For example, users may want to turn off the incidence factor at an event with many people, like a conference, where they may run into the same devices multiple times. After BL(u)E CRAB flags a device as suspicious, it appears on the report page with a colored circle indicating how many risk factors exceeded the threshold. If the device broadcasts its name, it will appear with its ID. The user can press the device tile to open the device details view, see more information about a device, and find out why BL(u)E CRAB flagged it as suspicious.
3) Once a suspicious device is selected, BL(u)E CRAB will present the user with a table displaying the device identifier (UUID), device name (if provided), device platform (if provided), device manufacturer(s) (if provided), and risk factors (Figure 4). At the bottom of the device detail view is a button labeled "Device Routes," which opens a map showing all the locations where the suspicious device appeared while scanning.
4) The map view helps users identify where and how the suspicious device entered their vicinity (Figure 5). The map shows red lines representing the user's approximate location when BL(u)E CRAB detected the selected device, with dots marking each path's starting and ending points. It is fully interactive, so the user can inspect specific areas to see more details and scroll along the path the device traveled with them.

IV. Privacy Considerations

BL(u)E CRAB protects user privacy by not collecting user login data, remaining independent from any first-party or third-party web services, and storing all data in an on-device database rather than uploading it elsewhere. BL(u)E CRAB does not require location permissions to operate nor block functionality should the user decline location access.

V. Conclusions and Future Work

BL(u)E CRAB aims to improve existing methods of detecting suspicious BLE devices based on risk factors with dynamic thresholds, unlike the previous methods, which rely on limited information and static thresholds. BL(u)E CRAB focuses on an adaptable, user-centric approach to evaluating suspicious devices, which can be adjusted to match the user's preferences based on their context. Avenues for future work include conducting detailed empirical evaluations, adding more filters based on the user's context, and explaining why a device is tagged suspicious in natural language.

References

[1] H. et al., "Please unstalk me: Understanding stalking with bluetooth trackers and democratizing anti-stalking protection," *PoPETS Proceedings*, 2024.
[2] I. Apple. "Apple and google lead initiative for an industry specification to address unwanted tracking." (2023).
[3] H. et al., "Airguard - protecting android users from stalking attacks by apple find my devices," ser. WiSec '22, Association for Computing Machinery, 2022, ISBN: 9781450392167.
[4] B. et al., "Ble-doubt: Smartphone-based detection of malicious bluetooth trackers," in *SafeThings 2022*, IEEE.