Location Privacy Protection in the Mobile Era and Beyond

by

Kassem Fawaz

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in The University of Michigan
2017

Doctoral Committee:

Professor Kang G. Shin, Chair
Professor J. Alex Halderman
Associate Professor Qiaozhu Mei
Professor Atul Prakash

Kassem Fawaz

kmfawaz@umich.edu

ORCID iD: 0000-0002-4609-7691

To my brother, sister, father and mother for their unwavering love and support.

ACKNOWLEDGEMENTS

"*Kassem, I can see the light at the end of the tunnel.*"

Kang G. Shin
sometime during my 3^{rd} PhD year

First and foremost, I would like to express my sincere gratitude to my advisor Professor Kang G. Shin. Prof. Shin has been there for me for the past six years; he has given me immense support and fostered an environment where I can grow and mature as an independent researcher. His support extended well beyond technical matters; he was a real mentor about the different aspects of life. For this, I am very grateful.

I am also grateful to my Ph.D. committee members: Professors J. Alex Halderman, Qiaozhu Mei and Atul Prakash for their constructive comments in shaping this dissertation.

I would like to acknowledge Dr. Kyu-Han Kim, my mentor for two years as an intern at HP labs. Kyu-Han offered me freedom and was very patient while I explored different ideas and topics that resulted in the last two chapters of this dissertation. I also would like to acknowledge Professors Hassan Artail and Fadi Zaraket, my undergraduate and Masters research advisors. Prof. Artail introduced me to academic research during my second undergraduate year and was my mentor for the following four years. Prof. Zaraket advised my research for half a year and helped me greatly when I applied for Ph.D. positions.

During my Ph.D., I was fortunate to have had the company and support of the members of the Real-Time Computing Laboratory (RTCL). I would like to single out several RTCL members: Krishna Garikipati, Huan Feng, Arun Ganesan and Yu-Chih Tung. Krishna, Huan, Arun and Yu-Chih have been close friends throughout and after my graduate studies. I cherish the time, the many discussions and the trips I had with them over the years. I am

truly fortunate to have had the company of these great friends during my stay in Ann Arbor. I would also like to thank two RTCL members: Caoxie (Michael) Zhang and Eugene Chai for showing me the ropes during the first two years of my Ph.D. Finally, I would like to thank the rest of RTCL members for providing such a friendly and intellectual environment during my graduate studies: Seunghyun, Xiaoen, Liang, Yuanchao, Hamed, Youngmoon, Dongyao, Sunmin, Chun-Yu, Tak, Taeju and Juncheng.

I am also very grateful to three of my closest friends: Ali Ghandour, Alamjad Salami, and Hamza Harkous who I met during my undergraduate studies at AUB. Over the past ten years, Ali has been like an older brother to me; I often seek his advice on many of the decisions I make. Alamjad has been a true companion throughout this Ph.D. journey. I don't think there is anything I did not discuss with Alamjad throughout the years. I have also been very fortunate to have Hamza as a close friend and collaborator. Hamza is very smart, nice and responsible; I always enjoy working with him.

Most importantly, I would like to express my deep gratitude to my brother, sister, mother and father. My family bore the responsibility of supporting me during the bleakest days. I thank my brother, Ahmed, most of all for never hanging up the phone when I gave him so many reasons to do so. Ahmed is more than a brother; he is my best friend. He can always me make me feel good and I highly value his judgment and positive perspective. I am very fortunate to have had my sister, Mariam, by my side during my graduate studies. Mariam is a very caring and kind sister as well as a critical thinker. She read all my papers before submission, managed my online presence and was a constant source of encouragement. No words can express my gratitude towards my father and mother. My father has made immense sacrifices over the years so that we can receive the best education. He has never asked for anything in return. Above all, I am eternally indebted to my mother for her love, support, and care throughout the years. Everything I have achieved, I owe to her.

Finally, I would like to acknowledge the National Science Foundation for their support through grants 1114837, 1505785 and 1646130.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

Location Privacy Protection
in the Mobile Era and Beyond

by

Kassem Fawaz

Chair: Kang G. Shin

As interconnected devices become embedded in every aspect of our lives, they accompany many privacy risks. Location privacy is one notable case, consistently recording an individual's location might lead to his/her tracking, fingerprinting and profiling. An individual's location privacy can be compromised when tracked by smartphone apps, in indoor spaces, and/or through Internet of Things (IoT) devices. Recent surveys have indicated that users genuinely value their location privacy and would like to exercise control over who collects and processes their location data. They, however, lack the effective and practical tools to protect their location privacy. An effective location privacy protection mechanism requires real understanding of the underlying threats, and a practical one requires as little changes to the existing ecosystems as possible while ensuring psychological acceptability to the users. This thesis addresses this problem by proposing a suite of effective and practical privacy preserving mechanisms that address different aspects of real-world location privacy threats.

First, we present LP-Guardian, a comprehensive framework for location privacy protection for Android smartphone users. LP-Guardian overcomes the shortcomings of existing approaches by addressing the tracking, profiling, and fingerprinting threats posed by dif-

ferent mobile apps while maintaining their functionality. LP-Guardian requires modifying the underlying platform of the mobile operating system, but no changes in either the apps or service provider. We then propose LP-Doctor, a light-weight user-level tool which allows Android users to effectively utilize the OS's location access controls. As opposed to LP-Guardian, LP-Doctor requires no platform changes. It builds on a two year data collection campaign in which we analyzed the location privacy threats posed by 1160 apps for 100 users. For the case of indoor location tracking, we present PR-LBS (Privacy vs. Reward for Location-Based Service), a system that balances the users' privacy concerns and the benefits of sharing location data in indoor location tracking environments. PR-LBS fits within the existing indoor localization ecosystem whether it is infrastructure-based or device-based. Finally, we target the privacy threats originating from the IoT devices that employ the emerging Bluetooth Low Energy (BLE) protocol through BLE-Guardian. BLE-Guardian is a device agnostic system that prevents user tracking and profiling while securing access to his/her BLE-powered devices. We evaluate BLE-Guardian in real-world scenarios and demonstrate its effectiveness in protecting the user along with its low overhead on the user's devices.

CHAPTER I

Introduction

Since the smartphone was introduced, a revolution started; users discovered the convenience of getting driving directions before asking for them, and the comfort of a connected thermostat adjusting the temperature on their way home. The mobile market is expanding rapidly; the number of smartphones expected to exceed the world's population by the end of 2020 [5]. People use these devices to connect with others, read the news, check the weather, find nearby points of interest (PoIs), perform localized search, or even do shopping.

These and other context-aware services are enabled by the wealth of collected sensory data and inferred personal habits – a trend that will not slow down with the rise of the internet of things (IoT) computing paradigm. Beyond the smartphone era, the IoT promises new applications that will further improve the quality of life. With the number of connected devices, sensors, and actuators expected to hit 50 billion in 2020 [6], fitness devices, thermostats, door locks, heart pacemakers, cars, and appliances are becoming connected.

This convenience, however, comes at a considerable privacy cost. Our Devices, apps and advertisement libraries leak our movements to location-based service providers, as well as curious and malicious parties. Recording location over time risks our privacy by revealing the places that we visit [7], such as in the following cases:

- Location-aware apps running on smartphones access location to provide an array of location-based services. While useful, such services do not come for free. Service

providers use collected location information to push location-based advertisements and analyze the behavior of their customers through location-based analytics. A weather app, for instance, continuously accesses fine-grained location, while running. This weather app indeed uses location to provide weather information but does not need continuous fine-grained tracking to perform its functionality.

- In the past few years, major retailers have been investing in infrastructure to monitor shoppers' movements. An app such as Shopkick allows customers to check-in inside the store; for which in return, they receive coupons or promotions. At the same time, the retailer analyzes the shoppers' mobility to learn more about their aisles of interest and behaviors.

- More recently, there has been an ongoing trend of wearable devices. A health monitoring device, such a Bluetooth Low Energy (BLE) powered Glucose monitor, broadcasts unencrypted periodic beacons to let a gateway device connect to it. Any party monitoring Bluetooth traffic can track the bearer of the health monitor over time as well as identify a his/her sensitive health condition.

In these situations, a curious or even a malicious entity observes and potentially records our movements over time. Beyond tracking, such an entity by using our location data can draw inferences about our behavior, preferences, interests, or identities – all which we refer to as *location privacy threats*. In this thesis, we define the different scenarios where individuals' electronic location privacy is compromised, highlight state of the art in location privacy protection along with their shortcomings, and propose a set of location privacy-enhancing mechanisms that address those deficiencies.

1.1 System Model

In the current smartphone and the evolving IoT eras, there are three major scenarios where our location privacy is compromised, as evident from Fig. 1.1.

(a) Smartphone Apps. (b) Indoor Environment. (c) IoT Wearable Devices.

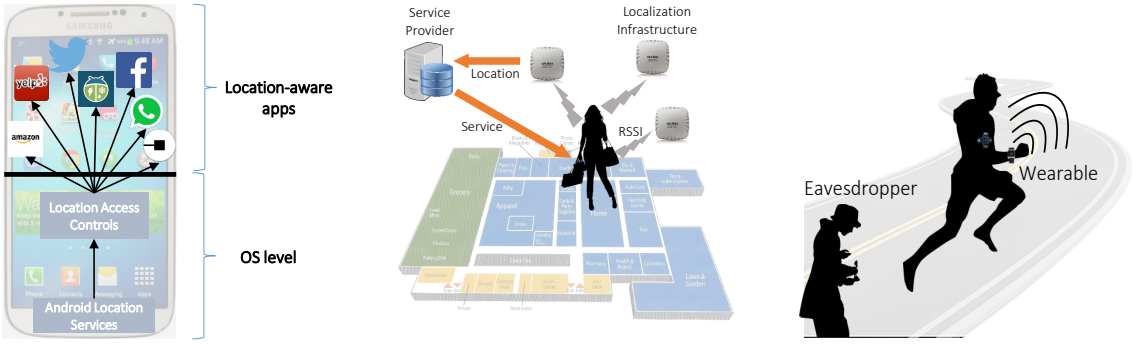


Figure 1.1: Location access scenarios in the mobile environments.

1.1.1 Location-aware Apps

Smartphone platforms (Fig. 1.1a), such as iOS and Android, offer location as a service for the mobile apps. A mobile device estimates the user's *outdoor* location via GPS or triangulation of nearby cell towers and/or access points. The mobile platform exposes the estimated user's location through a set of APIs for the mobile apps. In addition, the mobile platform provides location access controls to limit the user's location exposure. These controls come in the form of install-time permissions or run-time prompts.

The location-aware app can link user's location over time through the use of consistent identifiers. Depending on its location access pattern, each app will have a unique snapshot of the user's mobility. Advertisement and Analytics (A&A) libraries constitute another location access dimension that runs across the different location-aware apps. These libraries can aggregate the location traces of the set of apps in which they are packed.

1.1.2 Indoor Scenarios

In indoor environments (Fig. 1.1b), traditional localization techniques are inapplicable due to their poor coverage and inaccuracy [8]. Advanced localization technologies utilize a multitude of sensors (e.g., accelerometer, WiFi, Bluetooth low energy, and RFID) to locate the user with an error less than 10m in indoor environments, such as retail stores, malls,

airports, museums, and hospitals.

An *indoor service provider* (e.g., retail store owner) works with a *localization solution provider* (e.g., Cisco, Apple, HP) to track customers' location. The service provider collects the customers' indoor location information to infer their preferences and interests. It then uses the inferred information to deliver improved services back to the customers. Retail shopping is one notable example; the retailer (service provider in this case) tracks the shoppers' mobility inside the store to learn more about their regular routes in the store, wait and dwell times, aisles of interest, etc. The retailer aims to provide a more personalized service to the shopper (coupons, deals, promos, etc.) which helps improve the shopping experience and eventually increasing sales [9].

In indoor environments, localization could be infrastructure-based in which the location is computed outside the user's device. The localization provider uses wireless beacons (e.g., WiFi or Bluetooth) emitted from the user's device(s) to estimate her location [10]. Alternatively, indoor localization could be a device-based in which the device utilizes sensed data from infrastructure (e.g., BLE or WiFi) to compute its location (without other entities' involvement).

1.1.3 The Internet of Things

Bluetooth Low Energy [11] has emerged as the *de facto* communication protocol in the new computing paradigm of the Internet of Things (IoTs) [12, 13, 14, 15, 16, 17]. The BLE (Bluetooth 4.0 and newer) protocol has been developed by the Bluetooth SIG to support low power devices such as sensors, fitness trackers, health monitors, etc. Currently, more than 75,000 devices in the market support this protocol along with most of more capable devices such as smartphones, tablet, PCs, and recently access points [18]. BLE-equipped products are embedded and used in every aspect of our lives (Fig. 1.1c); they sense nearby objects, track our fitness, control smart appliances and toys, and provide physical security.

Advertisements (periodic wireless beacons) are instrumental to the operation of the

BLE protocol and constitute the only means by which others can discover the BLE-equipped device. Each advertisement message contains the device's address, along with some of the services offered by the device and their respective values, such as its name and type.

1.2 Privacy Threats

Any party that systematically observes and records the user's mobility has the potential to pose an array of privacy threats. In the following, we discuss the different sources of location privacy threats and their implications for the mobile users.

1.2.1 Sources of Threats

In this thesis, we focus on two major parties capable of posing the location privacy threats: the location-based service providers and the entities eavesdropping wireless traffic from wearable and mobile devices. In particular, we are concerned with the systems (apps, libraries and devices) leaking location information, as opposed to the mobile users publicly revealing, posting or sharing their location information with others.

1.2.1.1 Location-Based Services

To provide a location-based service, be it in the indoor or outdoor cases, the service provider has to be aware of the user's location. In the outdoor case, a service provider obtains the users' location samples through the apps running on their devices. In the indoor case, on the other hand, the service provider can either record users' mobility through mobile apps running on their devices or through the deployed infrastructure. Through utilizing long-term identifiers, such as phone number, IMEI, MAC address, software-generated identifiers, service providers can link different user interactions over time [19]. Linking user's mobility over time enables the service provider to track the user's whereabouts, and hence his/her behavior, creating many privacy issues [20].

While it is safe to assume that a significant fraction of service providers are trustful, many are curious and interested in drawing inferences, that reveal sensitive attributes about the users. Others, to the users' dismay, may share location information with third parties, such as advertisement and analytics libraries. Some service providers may misuse the data or may not employ sufficient safeguards to protect the confidentiality of the users' location information. The mobile users' location traces might fall into the wrong hands, such as an adversary hijacking this information from the service provider's databases.

1.2.1.2 Eavesdropping Wearable Traffic

A wearable device, such as BLE-equipped one, advertises its presence to let interested parties initiate connections and glean relevant information. These advertisements, however, are a double-edged sword. An unauthorized, potentially malicious, party can collect these advertisements through an off-the-shelf 2.4 GHz radio (e.g., Ubertooth) or range extending antenna (e.g., BlueSniper[1]). Such a party can use these advertisements to learn more about the BLE-equipped devices of a certain user or in a particular environment [21]. Enumerating the set of devices that the user is employing is generally referred to in literature as the *inventory attack* [22]. BLE advertisements, due to poor design, implementation or configuration, leak an alarming amount of information that allows the tracking, profiling, and fingerprinting of the users.

1.2.2 Types of Threats

The privacy implications of systematically monitoring an individual's locations extend well beyond low-level tracking; they include fingerprinting the users as well as profiling their interests and behaviors.

[1] http://www.tomshardware.co.uk/how-to-bluesniper-pt1,review-1224-9.html

1.2.2.1 Tracking Threat

The most primary privacy threat arising from collecting user's location over time is that of tracking. The adversary might receive continuous location updates that enable locating the user in real time. The adversary might also be able to identify the user's mobility patterns (frequently traveled routes) and predict his/her future location with high accuracy by leveraging the typical consistency of people's movement patterns [23]. The tracking threat arises from all three scenarios of Section 1.1 through: a **location-aware app** regularly accessing the user's location (e.g., weather app) in the background; an **indoor service provider** utilizing infrastructure-based localization mechanisms (e.g., WiFi based localization); or an **eavesdropper** collecting BLE advertisements leaking consistent identifiers (e.g., Bluetooth address).

1.2.2.2 Fingerprinting Threat

Another issue pertaining to the long-term tracking of location information is the fingerprinting of users. Through sporadic location access, the adversary might isolate spatiotemporal patterns representing the locations that a user visits frequently and are consistent in time. These spatiotemporal patterns define a user's points of interest, including home and work locations among others. The adversary can use these places as quasi-identifiers [24] to fingerprint the users from anonymous location traces [20, 25, 26].

Golle and Partridge [20] showed that home and work locations can be used to identify most of the US residents from the census records. Bettini *et al.* [27] applied this concept to mobile networks, showing that a sequence of spatiotemporal patterns serves as quasi-identifiers. Researchers at Sprint [28] analyzed users' location records and tried to assess the anonymization level sufficient to publish the operator's location information without risking the identification of users. They concluded that a city-level anonymization is required to achieve a decent privacy level. De Montjoye *et al.* [26] performed a similar study and concluded that a handful of location samples are enough to identify a user.

In the particular case of BLE advertisements, we show later that the device types, typically part of the advertisement, can help fingerprint the user by revealing the types of devices s/he is carrying. We found that the combination of the devices a user carries can serve as a "quasi-identifier" for him/her. They enable user tracking and fingerprinting regardless of whether the Bluetooth address is randomized or not. Moreover, BLE advertisements might contain identifying information (the owner's name). The advertisements can enable unauthorized access to the device's attributes that reveals the user's identity as well.

1.2.2.3 Profiling Threat

The user's mobility trace might not include places that would reveal his/her identity, but information that the adversary can use to draw inferences about him/her; we refer to such inference as *location-based profiling*. In the outdoor location monitoring case, places with special significance can reveal sensitive attributes of the users. Examples include health clinics, religious places, certain entertainment venues, ethnic grocery shops, among others.

The same applies to indoor scenarios where the user mobility patterns are tightly coupled with potentially private personality traits and/or shopping habits. For example, a retailer can infer from the frequently-visited aisles the shopper's gender (men's vs. women's clothing), ethnicity (ethnic food aisles), socioeconomic status (expensive vs. inexpensive clothing and accessories), health condition (pharmacy aisles), sensitive interests (sporting goods, adult magazines and films), or religious beliefs (clothing, particular food aisles).

Even in the case of Bluetooth Low Energy, devices that users wear or deploy reveal a considerable amount of information about them. Different BLE-equipped devices[2] serve different purposes, typically coupled with various personal aspects, behaviors or needs. Through advertising its presence, each device leaks its type and attributes. Monitoring these advertising messages allows for profiling the user by revealing: a health condition (e.g., glucose monitor), personal lifestyle (e.g., fitness trackers), preferences (brands of

[2]http://www.bluetooth.com/Pages/Bluetooth-Smart-Devices-List.aspx

devices), interests (toys, cameras, Pet activity trackers, etc.),or user behavior (e.g., smart home sensors).

1.3 State of the Art

Location privacy protection has received considerable attention, mostly from research, in the past decade. Unfortunately, state of the art in location privacy research for mobile systems fails on the fronts of deployability, providing theoretical guarantees, and/or balancing between the privacy and utility requirements of users.

1.3.1 Deployability

One of the most important requirements of a location privacy protection mechanism (LPPM) is its deployability in a mobile system. An LPPM must be compatible with the ecosystem it targets and has to be practical to employ within the ecosystem.

1.3.1.1 Compatibility with the Target Ecosystem

To start with, an LPPM must operate in an online fashion in a manner compatible with the ecosystem. It should control every location access while maintaining a long-term privacy objective. For example, an LPPM targeting Android must intercept, and potentially modify, each location access by a location-aware app. Many of the existing approaches, however, have been evaluated on traces but were neither implemented on mobile platforms nor tested with actual apps.

These include offline private publishing of mobility traces such as the works of Rastogi and Nath [29], Abul *et al.* [30], Terrovitis and Mamoulis [31], and Chen *et al.* [32]. An online LPPM does not have the privilege of knowing the entire location trace beforehand. It has to enact privacy protection one location access at a time.

Moreover, some of the LPPMs hinge on unrealistic assumptions, such as trusted infrastructure to provide the privacy protection [33, 34], requiring a set of users of the same app

at the same time and same place (e.g., mixzones [35, 36]), or focusing on a small subset of location accessing apps [37].

1.3.1.2 Practicality

Even when LPPMs are compatible with the mobile ecosystem, they might require introducing changes to the ecosystem itself, the APIs or the apps making them unpractical for the average user to deploy. For example, Koi [38] relies on a cloud-based service to achieve location privacy protection. It also requires developers to use a different API for location access. Other mechanisms, such as Caché[39] and the work by Micinski *et al.* [40], provide apps with coarsened locations but require modifications to the apps.

Similarly, approaches proposed to improve the privacy of BLE device owners necessarily include changes to the protocol itself or to the way the BLE-equipped devices function [41, 42]. Amending the operation of such devices, post-production, requires their patching by securely pushing a firmware update. With thousands of manufacturers and developers around the world, it is very challenging, sometimes impossible, to guarantee firmware patches to the millions of already deployed devices [43]. Even a security-aware user might lack the ability to update the firmware of a BLE-equipped device. As such, these privacy enhancements for the BLE protocol are highly unpractical to employ.

1.3.2 Privacy criteria

Almost all of the proposed *online* LPPMs focus on low-level location tracking as the single location privacy threat. They ignore the longer term implications of location access, namely location-based fingerprinting and profiling. Addressing the higher-level privacy threats requires designing novel location privacy threat models and criteria. Such privacy models and criteria have to satisfy two requirements: (1) they have to adapt for the online operation, i.e. mitigate the risk on-the-go; and (2) not place a usability burden on the user by frequently prompting for privacy-related decisions.

Influenced by offline mobility datasets (vehicular traces, cellular traces, etc.), location privacy studies focus only modeling on the tracking threat by viewing mobile location privacy as if there were only one party *continuously* accessing a user's location [26, 28, 33, 27, 44, 45, 46]. These proposals lack the models for higher-level privacy threats.

Researchers proposed mechanisms to address the resulting tracking threats [33, 47, 48, 44, 49, 50, 51, 35] by hiding the user's *raw* location. In particular, they try to maximize the error between and adversary's estimate of the user's location and the real location. These mechanisms still reveal the high-level features of the user's mobility [28], thus becoming not or less useful. These higher-level movement patterns could eventually lead to user profiling and even identification [46, 26].

As for BLE, we find that the built-in address randomization scheme fails to combat device tracking in practice due to poor design and/or implementation. An adversary can still utilize information from the advertisements to track the user in many cases, regardless of the (randomized) address. Furthermore, some devices allow external connections without an existing trust relationship. Unauthorized entities can access unsecured data on the BLE-equipped devices that might leak sensitive information about their owners.

1.3.3 Privacy vs. Utility

Last but not least, a location privacy enhancing technology must offer a balance between the privacy protection and utility requirements for users. In both indoor and outdoor location tracking scenarios, proposed solutions to protect mobile users' location privacy fail to provide such balance. They often consider only one extreme of the privacy–utility spectrum. They provide coarse location access controls; the user either enjoys full privacy without utility or vice versa.

For the outdoor case, MockDroid [52] provides users with OS-based controls to disable access to certain resources in Android, including location. The app will never receive location updates. This is a solution that provides full privacy but zero utility. Similarly,

Micinski *et al.* [40] coarsen the location supplied to the apps without considering the threat level or the location granularity required by the app. Last but not least, mobile apps such as PlaceMask [53] and Fake GPS Location Spoofer allow users to supply fake locations for apps rendering them unusable, and hindering the apps' functionality. Even the location access controls of Android and iOS allow the users two options of either enabling or disabling location access. The users lack the usable tools enabling them to decide when and where should an app be allowed to access location, and with which granularity.

In the indoor case, existing approaches [54, 55, 56, 57] attempt to blindly prevent indoor location tracking for the sake of privacy. They, however, fail to recognize the mutual benefits between users and the service provider through proper location sharing. Complete blocking of tracking deprives the service provider from understanding the users and the users from receiving more useful services.

1.4 Thesis Contributions

Although users repeatedly express concern about the location privacy threats associated with their mobile and wearable devices [58, 59, 60, 61, 62], they still lack the effective and practical tools to mitigate these threats. Effective privacy protection mechanisms should provide theoretically sound tools to address threats. Practical mechanisms should implement effective theoretical tools with as little changes to the ecosystems they target as possible, while ensuring psychological acceptability to the users. As indicated above, the state of the art in location privacy protection fails meet these requirements. In this thesis, we address the following question:

Can we bring usable, practical, and theoretically sound location privacy protection mechanisms for mobile users in the different scenarios of location-aware smartphone apps, indoor location-based services, and for the Internet of Things manifested by prevalent BLE-equipped devices?

To answer this question, we improve over the state-of-the-art by proposing a set of

Table 1.1: A summary of the thesis contributions.

System	Target Ecosystem	Tracking Protection	Fingerprinting Protection	Profiling Protection	Deployment
LP-Guardian	Outdoor Smartphone Apps	✔	✔	✔	platform changes
LP-Doctor	Outdoor Smartphone Apps	—	✔	✔	user-level app
PR-LBS	Indoor Localization	✔	✔	✔	platform changes
BLE-Guardian	BLE-equipped Devices	✔	✔	✔	external hardware

online location privacy protection mechanisms (Table 1.1) that target location-aware apps in outdoor scenarios: `LP-Guardian` and `LP-Doctor`, indoor location-based services: `PR-LBS`, and BLE-equipped devices: `BLE-Guardian`. These location privacy protection mechanisms reduce the privacy threats from exposing the user's location. Each of them pushes state of the art in location privacy protection research by providing theoretical privacy guarantees, practical and usable protection, and a balance between the user's privacy and utility.

1.4.1 `LP-Guardian` [1]:

Most location privacy protection mechanisms, proposed in the literature, do not handle the privacy threats as posed by location-aware apps. To fill this gap in location privacy research, we propose `LP-Guardian`, the first *app-aware* framework for location privacy protection on Android platforms, including, but not limited to, smartphones. `LP-Guardian` anonymizes the user's location before the app can access it. It guarantees that the observed (by the service provider) mobility pattern, modeled as the frequency of visits to locations, of a certain user is indistinguishable from a general set of individuals. An adversary cannot attribute the observed location information to the real user. `LP-Guardian` only patches the Android framework, so that users have to employ a custom ROM or root their devices. It, however, requires no changes in the apps or the service providers. As such, privacy-aware users can install `LP-Guardian` and use the same mobile apps from

Google Play. LP-Guardian overcomes the shortcomings of existing approaches by addressing the tracking, profiling, and identification threats. We have implemented and evaluated LP-Guardian on Android 4.3.1. Our evaluation results show that LP-Guardian thwarts the privacy risks, without deteriorating the user's experience (less than 10% overhead in delay and energy). Also, LP-Guardian's achieves privacy protection at a the tolerable loss in app functionality.

1.4.2 LP-Doctor [2]:

In a follow-up project, we investigate the deployment challenges facing location privacy protection mechanisms. They all require changes either to the underlying platform or the infrastructure making them untenable to be deployed. Therefore, users are left with the location access control of mobile operating systems. We analyze the efficacy of these controls in combating the location privacy threats. For this analysis, we conducted the first location measurement campaign of its kind, analyzing more than 1000 free apps from Google Play and collecting detailed usage of location by more than 400 location-aware apps and 70 Advertisement and Analytics (A&A) libraries from more than 100 participants over a period ranging from 1 week to 1 year. We found that, without a location-privacy protection mechanism, 70% of the apps and the A&A libraries pose significant profiling threats even when they sporadically access the user's location. Existing OS controls (those that are built-in the mobile operating systems) are found ineffective and inefficient in mitigating these threats, thus calling for a finer-grained location access control. We propose LP-Doctor, a lightweight and open-source Android tool[3] which enables users to be aware of the underlying location privacy threats and exercise fine-grained location access control. As opposed to LP-Guardian, LP-Doctor runs completely in the user-level, and requires no changes to the underlying platform in the mobile operating system. This, however, entails a trade-off between usability and privacy. To run completely in user-level,

[3] https://github.com/kmfawaz/LP-Doctor

LP-Doctor sacrifices the ability to protect against the tracking threat posed by apps running in the background. LP-Doctor picks a novel privacy criterion that limits information leakage from the location data released to different apps. A user study of 227 participants indicated LP-Doctor's ease of deployability and usability.

1.4.3 PR-LBS [3]:

While LP-Guardian and LP-Doctor focus on outdoor location privacy, indoor environments exhibit different dynamics between users and service providers. Our survey of 200 individuals highlighted their concerns about this tracking for potential leakage of their personal/private traits, but also showed their willingness to accept reduced tracking for improved service. We propose PR-LBS (Privacy vs. Reward for Location Based Service), a system that addresses these seemingly conflicting requirements by balancing the users' privacy concerns and the benefits of sharing location information in indoor location tracking environments. PR-LBS includes three novel online location release mechanisms that achieve differential privacy guarantees and ensure that the user engages in a fair location-service exchange with the service provider. PR-LBS is a general framework; it acts as a broker between users and service providers when the service provider monitors users' mobility through tracking their wearable devices. We implement and evaluate PR-LBS extensively with various real-world user mobility traces. Results show that PR-LBS has little overhead, protects the users' privacy, and makes a good tradeoff between the quality of service for the users and the utility of shared location data for service providers.

1.4.4 BLE-Guardian [4]:

Our study of more than 200 types of BLE-equipped devices has revealed that the BLE protocol, despite its privacy provisions, fails to address the most basic threat of all – hiding the device's presence from curious and malicious eavesdroppers. To combat these threats, we propose BLE-Guardian, a novel device-agnostic system that protects ac-

cess to BLE-equipped devices and prevents unauthorized eavesdroppers from tracking them. BLE-Guardian opportunistically invokes reactive jamming to determine the entities that can observe the device existence through the BLE advertisements (device hiding module), and those that can issue connection requests in response to advertisements (access control module). BLE-Guardian combats security and privacy threats, has little overhead, and incurs minimal or no disruption to the legitimate BLE devices of the user. BLE-Guardian achieves its objectives with minimum requirements from an external radio that offers only the basic capabilities of reception and transmission on the 2.4GHz band. As a result, it avoids employing sophisticated and customized (thus impractical) radios and signal processing approaches.

This thesis is organized as follows. The first chapter presents LP-Guardian and the second proposes LP-Doctor. The third chapter discusses indoor location privacy and presents PR-LBS. The fourth chapter considers location privacy in the new era of Internet of Things and proposes BLE-Guardian. Finally, we conclude the thesis in the fifth chapter where we also discuss future research directions.

CHAPTER II

LP-Guardian

2.1 Introduction

Location privacy has been a hot topic in research and media over the last decade or so [58, 63, 64, 65, 26, 28]. The popularity of location-aware smartphones has led to the prevalence of apps that access users' location in order to provide them personalized/customized services. As we indicated before, location access has introduced a new class of privacy threats that users are increasingly becoming aware of. These threats range from an adversary's ability to localize an individual to fingerprinting and profiling him/her based on the places s/he visits.

Motivation:

To assess users' perceptions of location privacy and location-aware apps, we surveyed[1] 180 smartphone users. We recruited 70 participants through social network announcements and the rest through Amazon Mechanical Turk. We chose Mechanical Turk workers who have achieved "master qualification," i.e., those who have shown high competency of performing tasks.

We find the survey results supporting the deployment of a location privacy protection

[1] https://docs.google.com/forms/d/1VFKlSa3Heq7Wz_mY4MmL7YNu8D74Gtl-lDNlnXesUTo/viewform

mechanism. 78% of the participants believe that apps accessing their location can pose privacy threats. Also, 85% of them reported that they care about who accesses their location information, compared to 87% reported by a Microsoft survey [58] in 2012. Users are even expected to be more sensitive towards this issue in relation to the recent revelations on government accessing their location as collected by apps [64]. Interestingly, 52% of the surveyed individuals stated no problem in supplying apps with imprecise location information to protect their privacy. Only 18% of the surveyed people objected to supplying apps with imprecise location information.

There have been numerous research proposals for location privacy protection from various angles and in various scenarios. Unfortunately, the vast majority of them have not found their way to the common users. Existing mechanisms (e.g., see surveys by Krumm [63] and Shin [65]) suffer several shortcomings that hinder their deployment in the real world. These shortcomings can be best described in terms of effectiveness, efficiency, and practicality as will be more evident in Section 2.2. Existing mechanisms address the tracking threats without guaranteeing protection against profiling or fingerprinting threats. Also, they impose the same protection measures regardless of the app and the privacy threat it poses. Finally, most of these mechanisms rely on unrealistic assumptions, making their real-world deployment difficult.

In this chapter, we present the design, implementation, and evaluation of a new location privacy protection framework, called the *Location Privacy Guardian* (LP-Guardian), that addresses the shortcomings of the existing mechanisms. We show that privacy protection can be brought to the masses through a client-side solution at a minimal cost. Although we focus on the Android platform, `LP-Guardian` is applicable to other platforms that utilize the permission model to authorize location access (e.g., Windows Phone, BlackBerry OS) as well as those that rely on explicit user authorization for every location access (iOS).

Our design philosophy is based on seven main features as discussed below.

A. The app only accesses location when the user expects it to do so: A user ex-

pects the app to access his/her location only when a location-based functionality is required, e.g., localized-search, place check-in, etc. Most Android apps, however, engage in location acquisition without explicit user authorization/awareness. `LP-Guardian` addresses this issue by anonymizing location access in the background and enabling the user to choose the appropriate anonymization strategy for foreground location access. Finally, `LP-Guardian` feeds Advertising and Analytics (A&A) libraries anonymized location samples to prevent tracking through third parties.

B. The app only accesses location with a granularity sufficient to produce the location-based functionality: A majority of Android apps request location with a finer granularity than actually required to deliver necessary service to the user. Our analysis of the top 1150 location-aware apps in Google Play revealed that 68% of them can accommodate coarser location without significannot losses in quality of service. As a result, `LP-Guardian` feeds every app with the location granularity necessary to serve the user. `LP-Guardian` can thus safely anonymize location for the majority of the apps without hindering their functionality.

C. An anonymous app cannot identify/fingerprint the user based on his/her frequently visited places: As not all apps can afford an anonymized location, feeding them with an accurate location might lead to user identification or fingerprinting from frequently visited places (e.g., home and work) [27, 20, 25]. This is applicable even in apps that do not require explicit user identity to function, such as games, search apps, etc. In this chapter, we formalize this fingerprinting threat and propose a novel mechanism that addresses this threat.

D. A single app alone poses no significant profiling threats based on the collected location information: Some places the user visits are not sensitive to his/her identity/privacy but might assist in profiling him/her (e.g., health clinics, religious places, bars, etc.). `LP-Guardian` relies on the user to learn these places and anonymize the location, if needed, by applying the mechanism of Andrés *et al.* [66].

E. An app cannot track the user all the time even when tracking is required to perform functionality: Some apps might require constant monitoring of the user's location, such as fitness or speed-monitoring apps. The absolute location matters less than relative mobility. `LP-Guardian` reacts by replacing the real location samples with dummy ones that belong to a synthetic route. This route preserves the actual route properties (mainly speed) and does not reveal the actual user's location to thwart tracking.

F. Privacy protection fits within the existing mobile ecosystem: We implement `LP-Guardian` in the Android framework by instrumenting the `Location` object. Our implementation can be easily incorporated in custom ROM or even in a rooted device through the Xposed framework [67]. Moreover, it neither requires modification of the apps nor it relies on additional entities. Most importantly, it is app-aware as it applies different anonymization strategies independently for different apps.

G. Privacy protection comes at a minimal cost in usability and app functionality: Anonymization naturally comes at a cost in terms of usability, delay, energy, and loss of app functionality. `LP-Guardian` minimizes the interaction with the user and makes the anonymization decisions on his/her behalf. It also incurs minimal delay and energy overhead (less than 10%). Finally, `LP-Guardian` minimizes the instances of anonymization to preserve app functionality (more than 60% of the sessions are not anonymized). According to a user study that we performed, such loss of app functionality is tolerable as it comes in places where users do not usually require location-based functionality. This is the first time that a location privacy protection mechanism is evaluated on real-world app usage data.

Contributions:

This chapter makes the following main contributions: `LP-Guardian`

1. provides privacy protection on a per-app basis; the protection level is proportionate to the threat posed and location granularity requirements of the app;

2. prevents fingerprinting by leveraging the notion of indistinguishability; and

3. is relatively practical to deploy as it achieves protection for each app independently without modifying the apps or the service provider's infrastructure.

Organization:

The chapter is organized as follows. Section 2.3 defines the threat model, while Section 2.2 reviews the related work. Section 2.4 gives an overview and then details the design of the components of `LP-Guardian`. Section 2.5 details the architecture of `LP-Guardian`, while Section 2.6 describes its implementation. We evaluate `LP-Guardian` in Section 2.7 and make concluding remarks and discuss future work in Section 2.8.

2.2 Related Work

Approaches addressing location privacy fall into two categories: theoretical and practical.

Theoretical Approaches

These are the approaches that have been evaluated on traces, but were neither implemented on mobile platforms nor tested with actual apps. Most of these mechanisms address the tracking threat [33, 47, 48, 44, 49, 50, 51] in that they hide the user's raw location while still revealing the high-level features of the user's mobility [28], thus becoming not or less effective. These mobility patterns could eventually lead to user profiling and even identification. `LP-Guardian` protects user's privacy at three levels: *tracking*, *profiling*, and *identification* as will be evident later.

Moreover, some of these mechanisms hinge on unrealistic assumptions, such as trusted infrastructure to provide the privacy protection [33, 34], requiring a set of users of the same

app at the same time and same place (e.g., mixzones [35, 36]), or focusing on a subset of location accessing apps [37].

Practical Approaches

Researchers have also proposed more practical approaches that fit within the existing mobile platforms. MockDroid [52] provides users with OS-based controls to disable access to certain resources in Android, including location. The app will never receive location updates. This is a solution that provides full privacy but zero utility. Similarly, Micinski *et al.* [40] coarsen the location supplied to the apps without considering the threat level or the location granularity required by the app. Last but not least, PlaceMask [53] allows users to supply fake locations for apps rendering it unusable. `LP-Guardian` has the advantage of balancing between privacy requirements, usability, and quality of service.

Other proposed systems require changes to the existing mobile ecosystem to provide the privacy guarantees. Koi [38] relies on a cloud-based service to achieve location privacy protection. It also requires developers to use a different API for location access that is based on a location matching criterion, rather than the raw location. Similarly, Caché[39] requires developers to change the way they access location in the apps. `LP-Guardian` requires no modification to the existing apps or infrastructure (it is a completely device-based solution), facilitating its deployment.

Finally, several researchers have studied the problem of private information leakage in mobile devices. For example, TaintDroid [68] tracks the information propagation in mobile devices and detects whether private information (including location) has been leaked. `LP-Guardian` is complementary to such approaches; it controls what location information the app gets access to, while taint tracking reveals how the apps are managing the accessed location information.

2.3 Threat Model

We assume an honest-but-curious and passive adversary who is interested in inferring more information about the user from collected location information. Apps constitute the only mechanism by which the adversary can access the user's location through the available location APIs. The adversary will not attempt to hack into the system or circumvent any privacy controls. We view the security challenges as orthogonal to our work. Our objective is not to implement a solution that prevents a determined adversary from overriding the operating system's controls. The existence of such a solution will further strengthen `LP-Guardian`.

The adversary will collect user's location as part of the app's operation. The collected location information will enable the adversary to pose the following three types of threats:

- *Tracking Threat*: the adversary might receive continuous location updates that enable him/her to locate the user in real time. The adversary might also be able to identify the user's mobility patterns (frequently traveled routes) and predict his/her future location with high accuracy by leveraging the typical consistency of people's mobility patterns [23].

- *Identification/Fingerprinting Threat*: Even if the adversary sporadically accesses user's location, s/he might still be able to isolate the user's frequently visited locations, such as home and work. The adversary can use these places as quasi-identifiers to reveal the user's identity from anonymous location traces [20, 25].

- *Profiling Threat*: The user's mobility trace might not include places that would reveal his/her identity, but places that the adversary can use to profile him/her. Examples include some health clinics, places with religious significance, etc.

We treat all apps belonging to the same developer or having the same signature as one sink of location information. We choose to trust the underlying operating system, since

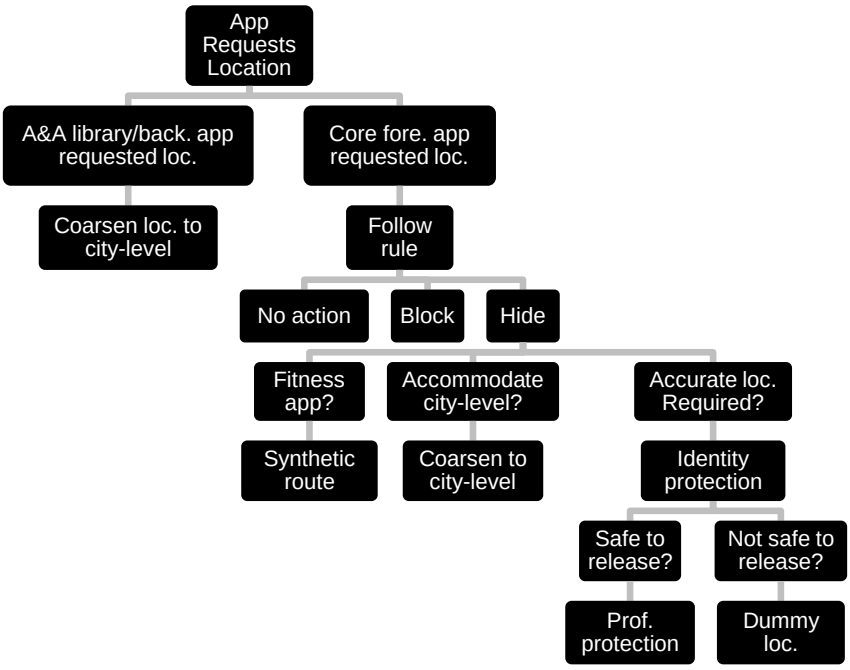


Figure 2.1: The decision diagram highlighting `LP-Guardian`'s main operations when an app receives a new location update

no practical solution can be implemented without such a trust. The user also trusts his/her device, including the underlying OS, to manage and store all his/her personal information.

2.4 Design

2.4.1 High-Level Overview

Before delving into the inner-workings of `LP-Guardian`, we present a high-level overview as shown in Fig. 2.1. This diagram highlights the main operations performed whenever a new location sample is to be delivered to an app (Section **??**).

`LP-Guardian` first determines if an A&A library or the core app is receiving the location update (Section 2.4.4). If the A&A library receives the location update, then the location is automatically coarsened to the city-level. If the app is running in the background,

the location is also coarsened to the city-level (Section 2.4.3). We base this design decision on our analysis of the location-accessing apps. The analysis revealed that only 3% of the apps access user's location while running in the background. Thus, location coarsening in the background has little effect on the functionality of most apps. If the core app, while running in the foreground or perceptible states, is receiving the location update, we rely on the user's preference. The location can be released without modification, completely blocked, or anonymized. In the case of anonymization, there are three available options as follows.

1. The app can accommodate coarsening without loss of service (weather apps), in which case the location is automatically coarsened to the city-level (the location is replaced by a pair of coordinates representing the center of the city).

2. The app is monitoring the user's mobility (fitness app), where `LP-Guardian` feeds the app a synthetic route that preserves some features of the user's actual route (Section 2.4.7).

3. The app requires location with high granularity (e.g., geo-search app). `LP-Guardian` applies a novel mechanism to control release of the location to prevent any possible identification (Section 2.4.5).

If the location is safe to be released, `LP-Guardian` consults with the user to check if s/he is comfortable with release of the location. If the user isn't comfortable, the location is obfuscated (noise added) to hide the visited place (Section 2.4.6), else the location is released as is. On the other hand, if `LP-Guardian` decides that it is not safe to release the location, it replaces the real location with a fake location as described in Section 2.4.5. Finally, `LP-Guardian` minimizes user interaction as much as possible, in order not to hinder the user experience. At the same time, it keeps the user in the loop by informing him/her of the anonymization process and asking for his/her feedback only when needed (Section 2.6.1).

In what follows, we elaborate on the main design decisions that address the various location privacy threats.

2.4.2 Location Sources

The location APIs are not the only way by which apps can access location information. Scanning the nearby WiFi access points (APs) and cellular towers might help locate the user. Skyhook, for example, is an online service that maps the signature of nearby APs to location coordinates. Such location can be fairly accurate (<30 m accuracy). Google localization service relies on a similar concept to provide network-based location samples. `LP-Guardian` addresses this issue by preventing apps from scanning nearby APs to limit the apps' location access to Android's location APIs.

2.4.3 Foreground vs. Background

In Android, an app can assume multiple states depending on its execution status. Android recognizes four app states, three of which are important to us: running in the foreground, background, and perceptible to the user. A foreground app is the one that occupies the screen and the user can interact with. When the user exits the app, Android caches it for faster re-execution and is thus moved to the background. An app can also run persistently and show the user a notification indicating that it is perceptible. For example, apps like Google Now run persistently as a service and are not considered background apps.

Because of elongated periods of location access, tracking threats are more pronounced in apps accessing location in the background or when running as a persistent service. We handle the former case of background location access by coarsening the location to the city-level. That way, the app will still receive relevant location updates, but will not be able to pose any viable location privacy threats. We will also specify how we handle location access in the foreground and as a persistent service.

```
dalvik.system.vmstack
java.lang.thread
android.location.location
com.medialets.analytics.e
com.medialets.analytics.mmanalyticsmanager
com.medialets.analytics.mmanalyticsmanager
com.medialets.analytics.mmanalyticsmanager
com.medialets.analytics.mmanalyticsmanager
com.medialets.advertising.admanagerservice
android.app.intentservice$servicehandler
android.os.handler
android.os.looper
android.os.handlerthread
```

Figure 2.2: The stack trace for a location request by WebMD app with the analytics library method calls highlighted.

2.4.4 A&A Libraries

Most free apps pack A&A libraries to generate revenue by displaying targeted ads to the user running the app. Hence, the apps need to feed these libraries some information about the user, including location [69]. As the number of A&A libraries is limited as compared to the number of apps, an A&A library is most certainly to be packed in multiple apps.

Instead of looking at apps as independent sources of the user's location trace, A&A libraries can aggregate location traces from multiple apps. This implies that the location privacy threats posed by these libraries is more critical. However, apps can thus easily accommodate feeding these libraries with coarsened location samples to the city-level. These libraries will still receive relevant location information to display ads to users, but will not be able to pose any viable tracking/identification/profiling threats.

Theoretically, it is plausible to coarsen location for A&A libraries, but the challenge is how to separate between location requests coming from the A&A library and those coming from the core app. To deal with this challenge, we studied more than 100 A&A libraries in Android. We collected these libraries from the top 1100 apps in Google Play and from different literature sources [70, 69, 71, 72]. It turns out that 98% of these libraries access location information from their code space through two mechanisms: the app (1) enables location collection, or (2) passes them a location object that they can access. In both cases,

the stack trace of every location access request should reveal whether the request is coming from the app or the packed library. Fig. 2.2 shows an example of the stack trace of a location access request from WebMD app; it is evident how the request originates from Medialets Analytics.

2.4.5 Identification/Fingerprinting Protection

As `LP-Guardian` coarsens location accessed in the background, accurate location access is limited to the foreground. The apps will thus only sample the user's location only when running in the foreground. Foreground sessions are short, sporadic, and occur mostly within the same place. These facts will be shown later in Section 2.7 based on three datasets that measure app usage patterns. This implies that there is a one-to-one mapping between an app session and a visited location. For these apps, the mobility information can be best viewed in the form of a histogram of visited places. We model the city which the user is visiting as divided into a 2D grid, where every cell refers to a city block. The set of city blocks is $Bl = \{bl_1, bl_2, bl_3, \ldots\}$. Every resident has a probability distribution of visiting the blocks in the city as $p_i = P(bl_i)$; this probability will be 0 for blocks the user never visits. After a period of app usage, the app records the number of user visits to every block, thus forming the histogram. Each bin in the histogram is the number of times, c_{bl_i}, the app observes that the user was at the block bl_i.

Even if the location information is anonymous, an adversary can map the user's histogram to the user's identity given the background information at the adversary's side. This is widely referred to as the *identification* or *inference* attack in literature [25, 27]. In what follows, we provide the first formal treatment of this attack. We assume that the adversary has access to background information in the form of a mapping between a set of individuals' identities and the probability distribution of visiting each block in the city. The adversary aims to match the anonymous mobility histogram from the app to one of the individuals in its database.

Specifically, the adversary is interested in the probability of the histogram belonging to any of the individuals, x, in its database, $P(h_{app}|x)$. If the app records N user sessions, $P(h_{app}|x)$ is the probability of observing the individual x being at each bl_i for c_{bl_i} times out of a total of N observations, where this individual has a probability p_i of visiting block bl_i. As a result, the probability distribution of the histogram follows the multinomial distribution, assuming that different app sessions are independent, and is given as:

$$P\left(h_{app}|x\right) = \frac{N!}{\prod_{i=1}^{Bl} c_{bli}!} \prod_{i=1}^{Bl} p_i^{c_{bli}}. \tag{2.1}$$

This model holds when $c_{bli} = 0$ as $c_{bli}! = 1$ and $p_i^{c_{bli}} = 1$.

If the app cannot accommodate coarsened location (e.g., geo-search app), then `LP-Guardian` releases the user's location as long as the user's histogram cannot be mapped to his/her identity. In other words, the probability of the histogram originating from the real user should be close to the probability of the histogram originating from another individual within the same city. We define the "closeness" of probability distributions in terms of Eq. 2.2 which is similar to the δ disclosure criterion [73]. We use this notion as a measure of indistinguishability between two histograms. Below, we elaborate on how we leverage this notion to establish an indistinguishability criterion over a theoretical set of individuals. For two individuals x and y, the histogram maps to both individuals with a close probability:

$$e^{-\epsilon} \leq \frac{P(h_{app}|x)}{P(h_{app}|y)} \leq e^{\epsilon}. \tag{2.2}$$

Plugging the probability expression of Eq. 2.1 into Eq. 2.2 we get for two individuals x and y:

$$e^{-\epsilon} \leq \frac{\prod_{i=1}^{|Bl|} \left(P(bl_i|x)\right)^{c_{bl_i}}}{\prod_{i=1}^{|Bl|} \left(P(bl_i|y)\right)^{c_{bl_i}}} \leq e^{\epsilon}. \tag{2.3}$$

For the rest of this chapter, we assume $\epsilon = 0.5$. The user's histogram must obey the

property of Eq. 2.3 to satisfy the privacy criterion. As `LP-Guardian` operates solely on the client side, it has no information about other individuals in the city, and thus cannot fill in the probabilities for other potential individuals. Alternatively, we consider a criterion that other individual must satisfy in the form of a minimum probability of visiting the blocks that the user visits. So, we replace the $P(bl_i|y)$ values in Eq. 2.3 with a value of p_{min}. Now, the privacy criterion states:

The user is indistinguishable within a theoretical set of individuals who have a minimum probability of p_{min} of visiting *all* the places in the user's histogram.

The value of p_{min} is a tunable parameter that controls the level of the user's privacy; the higher p_{min} the lower the privacy guarantee, as less people will visit the same places as the user with high probability. On the other hand, a lower value of p_{min} will indicate a stricter privacy guarantee as the user will be potentially indistinguishable within a larger set of individuals. We rearrange Eq. 2.3, after applying the logarithm, to satisfy the following inequality:

$$-\epsilon \leq \sum_{i=1}^{Bl} c_{bl_i} \left(\ln\left(P(bl_i|x)\right) - \ln\left(p_{min}\right)\right) \leq \epsilon. \tag{2.4}$$

This inequality provides a test for releasing the location from a session or not. For every new app session, the expression of Eq. 2.4 is evaluated given the past released histogram, the user's mobility model, and the value of p_{min}. If the summation value in Eq. 2.4 exceeds ϵ or falls below $-\epsilon$, then the location can be released; otherwise, a dummy location within the city is released. Eq. 2.4 provides an important insight on locations the user visits. If we view ϵ as a privacy budget, places that users frequently visit will exhaust part of the budget. On the other hand, if the user visits a place with a probability lower than p_{min}, this will increase the available privacy budget. The released dummy location is one that the user visits with a very low probability which helps increase the available privacy budget.

So far we have assumed the protection mechanism is blind of background information regarding other users. The availability of such information, however, can assist in improving the trade-off between app functionality and privacy. Our idea is based on hiding the user among a theoretical set of people who have to satisfy a minimum probability constraint of the visiting the places that the user visit. If we know that an actual set of people satisfy a more relaxed constraint, then we can achieve the same privacy level with much improved usability.

Our mechanism relies on the census data that specifies the population in every city block [74]. It identifies the user's home location as the most frequently visited place during night time. It then assigns the home block location a value of p_{min} consistent with the population in the same block. If the number of block residents is above 500, we automatically assign p_{min} for the home location the same value of the user's probability of visiting the home location. The privacy criteria then transforms to the subset of the home block residents who have a minimum probability of p_{min} of visiting the other user places of interest. In that sense, the user enjoys a natural protection level resulting from the fact that s/he lives in a crowded place. Our mechanism can be relaxed more in releasing the home location. This insures protection against the identification threat; the user will still have the option of hiding his/her location (or any other location) as will be evident later.

If a user lives in a sparsely populated area, then s/he naturally suffers lower privacy guarantees. LP-Guardian reacts by assigning very low p_{min} values for these users, meaning they will not be able to release locations from frequently visited places (e.g., home or work) for apps that require location with high precision (e.g., Yelp). Nevertheless, one could also argue that people who live in sparsely populated areas have a lower need for location-based services at home or work as they will tend to be more familiar with the area.

Finally, our modeling of the user's mobility as a histogram removes two pieces of relevant information: the timing information and the ordering between the visits. We can address the first issue of the timing information by associating each histogram bin with a

visit time, such as weekday versus weeknight or daytime versus nighttime. This, however, results in sparser histograms and requires more stringent privacy criteria (lower p_{min}) to avoid underestimating the privacy threat. In this chapter, we do not address the second issue of the ordering between visits. This issue, however, is addressed in the fourth chapter in the context of `PR-LBS`.

2.4.6 Profiling Protection

A user could be profiled based on the places that s/he visits, albeit at a low frequency. A user might not be willing to reveal that s/he is visiting a particular church, a health clinic, a certain bar, or some hotel. Revealing these places might enable an entity with access to the user's location to profile him/her. `LP-Guardian` addresses these threats by putting the user in control, as s/he is the best judge of determining the places which profile him/her. Each time the user invokes an app from a new place, the user has to decide whether s/he is willing to hide the place s/he is currently visiting (more details in Section 2.6). If the user opts to hide his/her place, we leverage the solution of Andrés *et al.* [66] to anonymize his/her location. According to our datasets of app usage (more on that in Section 2.7), this prompt appears for each location-aware app, on average, for 16 times during its lifetime. It is worth noting that `LP-Guardian` applies only to one-fifth of the user's apps, those that are location-aware apps requiring fine location to function correctly.

This approach is well-suited for sporadic location access. It adds noise drawn from a polar Laplacian distribution to make the reported location provably indistinguishable from the actual location within a given radius. Given the user's actual location as a pair $< x, y >$ and an anoymization radius r, the added noise can be computed in terms of a pair $< rad, \theta >$ as follows [66]:

- Draw θ from the uniform distribution $[0, 2\pi)$;
- Draw p from the uniform distribution $[0, 1]$, and set $rad = -\frac{r}{l}(W_{-1}(\frac{r(p-1)}{l}) + 1)$, where W_{-1} is the -1 branch of the Lambert W function and l is a privacy level

typically chosen close to 1.

The new location is just a translation of the original location by rad and θ. The noise level, r, determines the underlying privacy – utility trade-off. In `LP-Guardian`, we rely on a noise value of 200m; the user's actual location is hidden among location within a range of 200m. Still, the location will be of some utility to maintain some app functionality. As long as the user is visiting the same place, the anonymized location reported to the app is kept the same to prevent any additional leakage of information.

Finally, the user's IP address can reveal the place s/he is visiting, especially if s/he accesses the Internet from a public hotspot [75]. Although not included as part of `LP-Guardian`, TOR (Android app is available on Google Play) could be used to anonymize the user's IP address and protect his/her location.

2.4.7 Synthetic Route

Fitness apps track the exercise activity of the users. They provide the user with feedback on the path and distance covered during an exercise session. These apps monitor the user's location in the background for elongated periods of time and with high location precision, thus posing a tracking threat. We approach this type of apps by separating the two objectives: (1) distance and the (2) path of the exercise. Instead of crudely anonymizing location, which will render these apps useless, we sacrifice the second objective to provide privacy while maintaining the first. In other words, we provide the app with a synthetic route that has the same distance of the original route, but has a different shape.

`LP-Guardian` anonymizes the location for these apps by keeping the distance the user covered the same while modifying the actual route. It essentially feeds the fitness app a synthetic route that has the same length of the actual route. Whenever a new fitness tracking session starts (such an app is running as persistent service and location update rate is high), `LP-Guardian` feeds the app with a random location within the current city. Then, as the app receives new location updates, `LP-Guardian` reacts by calculating the

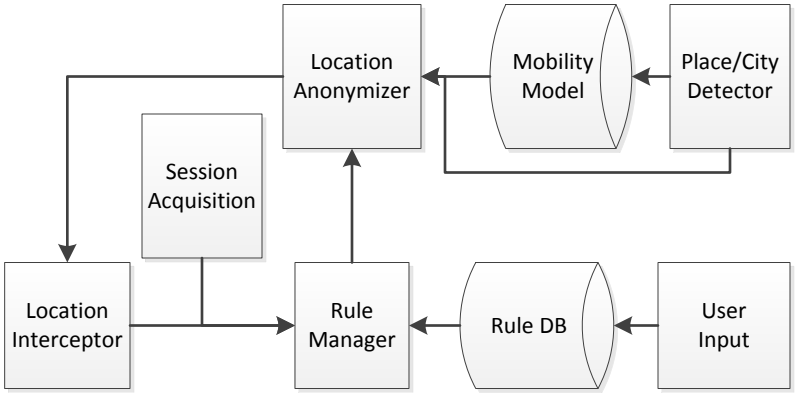


Figure 2.3: LP-Guardian's architecture and interactions of its components

distance covered since the last update, and calculates a new location sample based on this distance and the last reported location.

2.4.8 Navigation Apps

Navigation apps are the most challenging. They require elongated location access periods with high precision. The current version of LP-Guardian doesn't handle this case, but we provide some remedies that balance usability and privacy in navigation apps. Offline navigation (e.g., Garmin, Google Maps) can handle part of the problem by preventing the app from sharing the user's location in real time. However, there are no guarantees against an app leaking this information when it comes online. A possible remedy is to run the navigation app in a "private mode", where it's disbarred from connecting to the Internet, and the stored data is wiped after the session ends. The downside is that the app will be deprived of access to the real-time traffic information that is useful for navigation.

2.5 Architecture

Fig. 2.3 shows the block diagram that implements LP-Guardian. Described below are the main components of LP-Guardian and their interactions.

Location Interceptor

The location interceptor, as its name implies, is responsible for diverting the app's control flow to our service in the event of a location access. This module blocks the app, extracts the newly acquired/created location, and sends it (along with the app's package name) to our service for further processing. It is also responsible for delivering the anonymized location to the app so that the control flow can be resumed. We describe the implementation of this module in Section **??**.

Rule Manager

This module selects the appropriate rule—via interaction with the user—to apply depending on the situation. The rule indicates an appropriate strategy for the location anonymizer module to follow. The rule manager module takes as input the app, location sample, app state (foreground or background), and whether this is a new or ongoing session from the session acquisition module. A foreground/background session is the time period during which the app is continuously executing in the foreground/background. The rule manager also takes input from the place detector module to know which place/city/block the user is currently visiting. This module caches the rule from the database for use in subsequent location accesses of the same session. We define two types of rules as follows.

- *Global Rules*: indicate the protection mechanisms invoked for both the foreground and background states. There is only one global rule per app.
- *Per-place Rules*: control the protection level of individual places the user might be visiting. There is one per-place rule for every app–place combination.

The global rule is always given preference over the per-place rule as it considers the big picture of the user's released location traces. Invoking this rule will determine whether the current location is safe to be released. If it is to be released, the per-place rule is invoked to decide if any further protection is needed.

Place/City Detector

This module is responsible for identifying the current place the user is visiting and for maintaining the user's mobility model. It performs online processing of the user's real location trace to identify spatio-temporal clusters, in a manner similar to that proposed by Bamis *et al.* [76]. Formally, we define a place as a cluster of locations within a radius of 100m and over a minimum duration of 5 minutes. Whenever the place detector receives a new location sample, it tries to map it to one of the existing places in the database. If the mapping is successful, it updates the total visiting time of that place, else it creates a new place. The mobility model is the set of places the user visited along with the total visiting time of every place. This enables computation of the probability of visiting every place. This module also maps the user's location to the nearest city/block.

Location Anonymizer

The location anonymization module is the cornerstone of `LP-Guardian`. It takes as an input the location, the app state (foreground or background), and the rule from the rule manager module, and outputs the anonymized location back to the location interceptor module. The rule dictates the appropriate anonymization strategy to be followed as indicated in Section 2.4.

2.6 Implementation

Android provides apps with two mechanisms with which they can access the user's location: the older Location Manager Service and the newer Google Play Services. In both mechanisms, the apps request regular updates by registering a location listener which acts as a callback function (Fig. 2.4 – left). Whenever a new location is ready, the callback function is invoked, and the app receives a new location object.

To enforce any kind of location protection mechanism, the location passed to the app

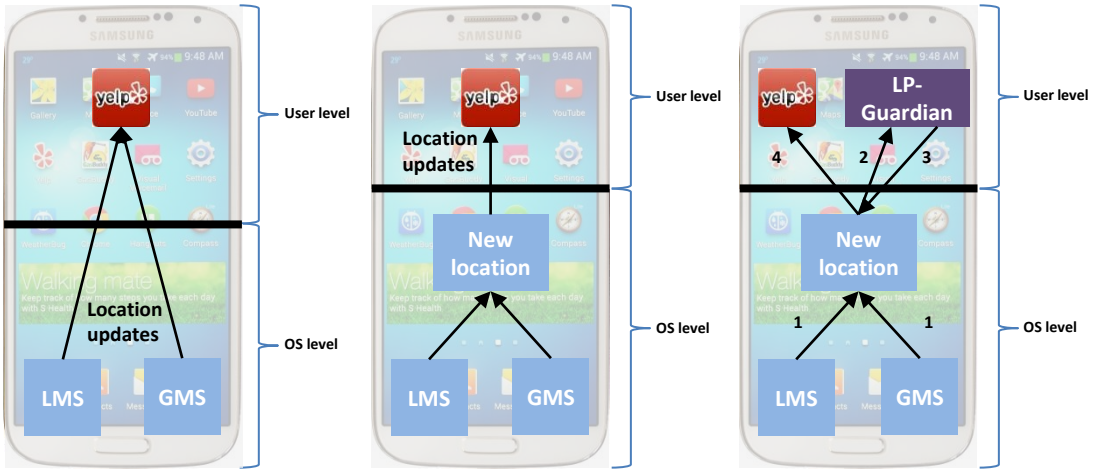


Figure 2.4: Location access mechanisms in Android (left & middle) and `LP-Guardian`'s deployment within Android (right)

has to be modified. Moreover, such a mechanism has to be app-aware so that it may modify the location on a per-app basis. There are two options for modifying the location on a per-app basis. The first involves instrumenting the app's location accessing interfaces and intercepting location updates before they reach the app. While the other option involves instrumenting the platform and changing the location object before reaching the app.

The first option could be implemented with a mechanism similar to Aurasium [77]. Albeit effective, this mechanism requires unpacking, instrumenting, and then repackaging the app. Repackaging the app will change its signature. This will break any future updates to the app, and affect any functionality requiring an authentic app signature. Moreover, the users will have to download the app from a different app store that requires an entity to manage and keep it up-to-date. Instead, we opted to implement a platform-based solution that treats the apps as black boxes and maintains their full functionality.

Android relies on two mechanisms to relay location updates to the apps. So, any platform-based solution has to instrument both mechanisms to intercept and then modify the location object before reaching the app. Nevertheless, Android ships Google Play Services as a closed-source app, and hence, its instrumentation is not straightforward. Fortunately, both mechanisms create a location object prior to propagating it to the app. The

location class is included in Android code and can be altered easily (Fig. 2.4 – middle).

The location object is merely a data holder, but has to be created within the app process space before delivering it to the app. We added a static context field to the location class that is populated when the app is invoked, particularly when a context is created for the app. A context is what enables an app in Android to interact with the OS resources. Enabling the location object with a context enabled us to (1) know what app is currently creating a location object, and (2) communicate with OS or other processes, if needed. We instrumented the location object and pushed the location privacy logic to an independent system app that can be easily updated, if needed. Since the location object has a reference to the app's context, it can communicate with that external service (Fig. 2.4 – right) whenever a new location object is created.

Anonymization incurs a processing cost. It might be prohibitive subjecting every location update to the anonymization operation, especially in case of very high location update rates. It is unlikely that the user's position will change within a second, even with speeds up to 100 mph. Consequently, `LP-Guardian` only communicates with the anonymization service once every 750ms.

2.6.1 User Interface

Our design philosophy is to rely on the user's input as infrequently as possible. `LP-Guardian` makes decisions on his/her behalf to control different configuration and anonymization parameters. Nevertheless, there are some situations where the user needs to interact with `LP-Guardian` to enable three features: bootstrapping, per-place, and per-session controls. Each of these features is elaborated next.

2.6.1.1 Bootstrapping

Initially, the mechanism is blind and doesn't have enough information to function. The two main missing pieces of information are user mobility data and per-app anonymization

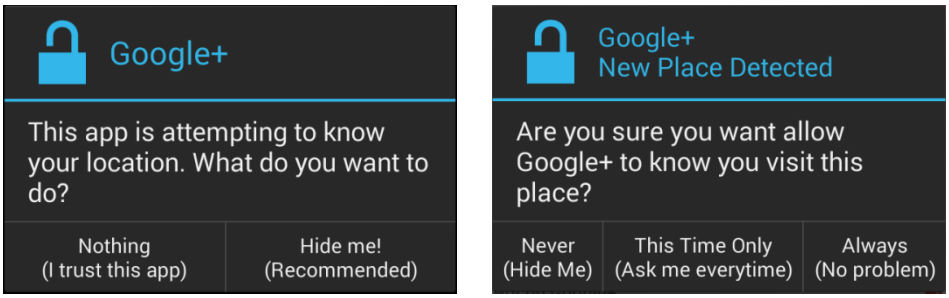


Figure 2.5: The displayed prompts to set the global (left) and per-place (right) rules

rules. As explained earlier, our identification protection mechanism requires mobility data to function. `LP-Guardian` cannot collect mobility data at run-time, as by the time we have this information, other apps will have access to it. At the very first time when the `LP-Guardian` boots, the user has to set his/her top N visited places by tapping on a map (the user still has the option of adding places later through an advanced settings menu). We then set the initial probabilities of visiting these places according to the model proposed by González *et al.* [78]. Specifically, `LP-Guardian` assigns every place a probability of visit inversely proportional to its rank of visit, then normalizes the probabilities to form a PDF.

The second piece of the bootstrapping includes setting the rule for every app. The first time the app accesses location, `LP-Guardian` present prompts (Fig. 2.5 – left) the user to set the app's anonymization rule. The user only gets to choose one of two self-explanatory options: "Do nothing" and "Hide Me." When the user chooses the second option, `LP-Guardian` chooses the appropriate anonymization mechanism depending on the app. It applies city-level coarsening for apps that accommodate such granularity. For the rest of the apps, it chooses identification protection with a default value of p_{min} = 0.0005. Additionally, `LP-Guardian` sets the per-place rule for the location the user is currently visiting. We use 0.0005 as the default value, which amounts to spending 30 minutes at a certain place each 40 days. `LP-Guardian` also has an advanced settings menu that allows the user to set fine-grained p_{min} for each app.

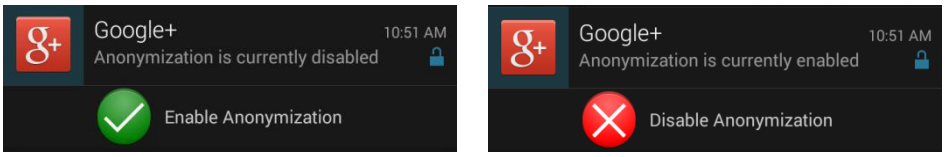


Figure 2.6: The displayed notifications when anonymization is disabled (left) and enabled (right)

2.6.1.2 Per-place/session controls

The per-place rules control the profiling protection level. As `LP-Guardian` cannot infer places sensitive to the user, it relies on explicit user input to set these rules. Whenever an app attempts to access the user's location from a new place (one the app hasn't seen before), `LP-Guardian` will prompt the user to ask for his/her decision (Fig. 2.5 – right). The user has to choose one of three options: hide the place every time (Section 2.4.6, reveal this place only during the current session, and reveal this place always. If the user chooses the second option, `LP-Guardian` will keep on issuing the prompt from the same place until the user chooses a permanent option (either the first or third option).

Finally, `LP-Guardian` always ensures that the user is aware of the anonymization if it takes place. Whenever the user is running an app and anonymization takes place, a notification is displayed (Android notifications are placed in the top left corner and are non-intrusive). This notification, shown in Fig. 2.6, displays the app, a note to the user, and an option to temporarily disable the anonymization during the current session. If the user disables anonymization, the notification will include an option to re-enable location anonymization.

2.6.1.3 Properties of the prompts

Authorizing resource access in software has been studied before. Livshits [79] proposed a set of four requirements for valid placement of user prompts that control for resource access (including location) in mobile platforms. They are: safety, frugality, visibility, and

non-repetitiveness. To achieve safety, a user prompt must precede every resource access. A prompt is frugal if it is *only* displayed in the event of a resource access. Visibility indicates that a user prompt must only be displayed if the app currently executing in the foreground is attempting to access a resource. Finally, a user prompt is not repetitive if it is never displayed for a resource access when a more critical resource of the same type has already been authorized.

`LP-Guardian` satisfies the last three constraints by prompting the user only in the event of a location request and while running in the foreground thus satisfying both frugality and visibility. As for non-repetitiveness, it is already ensured through Android's permission model. If an app is granted fine location permission, it is automatically allowed access for coarse location, but the inverse doesn't hold.

The safety requirement ensures that no location access goes un-checked. Theoretically, this is an optimal requirement to protect the user's privacy. Nevertheless, the users cannot be expected to assess the implications of every resource access and thus cannot always make informed decisions. Second, protecting every resource access with a user prompt will affect the user experience negatively as the app execution will be interrupted frequently. One option, as implemented in iOS, is to prompt the user at the first time location access with an option to remember the decision. Still, that doesn't appropriately address the two issues. If the user chooses to remember the decision, this will certainly reduce the frequency of the prompts but will not necessarily protect the user's privacy.

`LP-Guardian` addresses the above two issues by balancing between the frequency of prompts and privacy guarantees. Only the first location access from every newly visited place is preceded by a prompt to allow the user to make a choice. `LP-Guardian` then makes the subsequent decisions on the behalf of the user. It asks users to make decisions in terms of hiding or revealing places rather than asking them to authorize location accesses. Needless to say, users can relate better to places rather than raw location samples. It is worth noting that the frequency of prompting might be high at the beginning. It should,

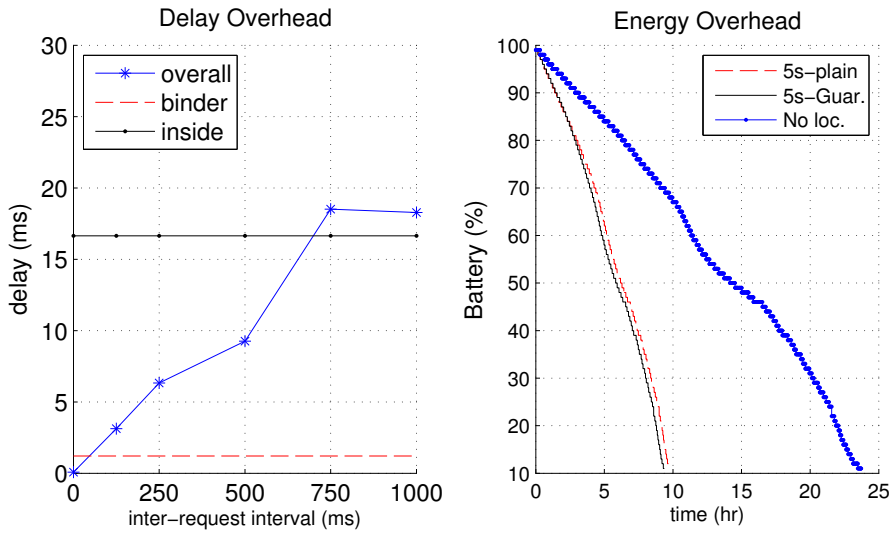


Figure 2.7: The delay overhead (left) and battery depletion rate (right) of `LP-Guardian` for Galaxy Nexus

however, decrease after `LP-Guardian` learns the places the user visits as people tend to exhibit consistency in their mobility.

2.7 Evaluation

We now evaluate `LP-Guardian` in terms of performance, privacy guarantees, and usability.

2.7.1 Performance

First, we validated that `LP-Guardian` can effectively obfuscate the location delivered to the apps and verified that it doesn't cause the apps to crash. We manually installed, ran, and verified 40 representative location-accessing apps. We then evaluated delay and energy overheads on three devices: Google Galaxy Nexus, Samsung Galaxy S3, and Samsung Galaxy S4 running Android 4.3.1.

To evaluate the delay overhead, we simulated location access through an app that accesses location with a varying frequency. We then measured the delay between the time the location is created and the time the modified location is delivered to the app. The left

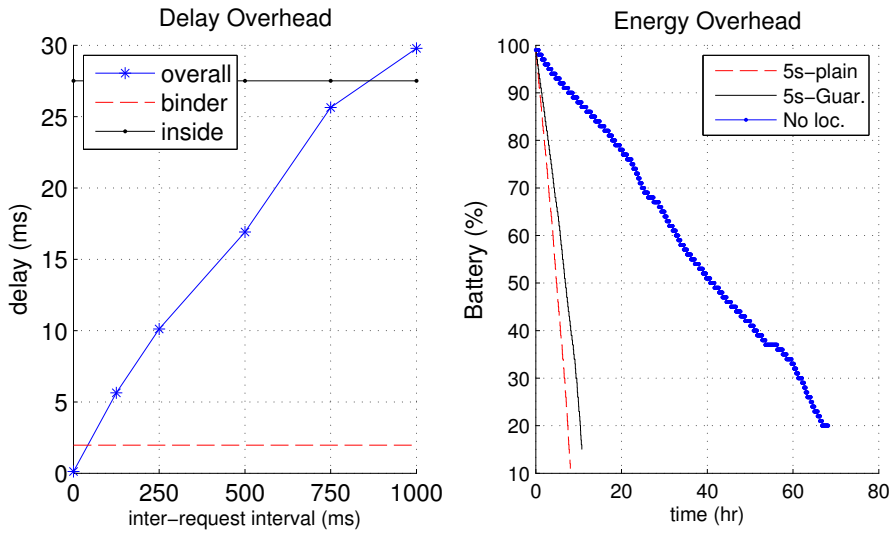


Figure 2.8: The delay overhead (left) and battery depletion rate (right) of `LP-Guardian` for Galaxy S3

plots of Figs. 2.7–2.9 show the average delay for each of the three devices versus the inter-request interval. Bearing in mind that `LP-Guardian` temporarily caches location, the delay value increases with the increase of the inter-request interval.

We also measured the anonymization delay for the worst-case scenario where the anonymization actually takes place (i.e., communication with the anonymization service). The delay was less than 20ms for two devices and less than 30ms for the Galaxy S3 (the curve labeled "inside" in every plot of the delay overhead plots (left plots of Figs. 2.7–2.9). `LP-Guardian` imposes a maximum delay of 30ms every 750ms, which shouldn't impact the app usability.

To evaluate the energy overhead, we considered two scenarios: no location access, and a load of location access of one request per 5 seconds. We measured the rate of battery depletion for each scenario with `LP-Guardian` running and compared it with the rate when the framework is not running. The right plots of Figs. 2.7–2.9 show the rate of battery depletion for each of the three devices and for the two loads.

For a location access rate of 1/5 Hz (one request per 5 sec), the battery depletion rate when `LP-Guardian` is running is very close to the case when it is not. This is evident

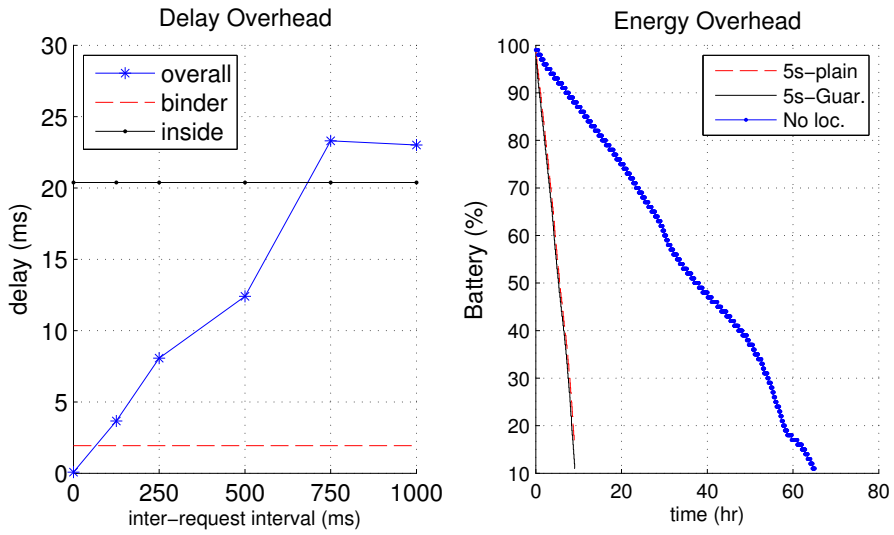


Figure 2.9: The delay overhead (left) and battery depletion rate (right) of `LP-Guardian` for Galaxy S4

from the the curves close to the y-axis in the energy overhead plots (the right plots of Figs. 2.7–2.9) for each of the three devices, with an energy overhead less than 10% for the three devices. In the case of no location access, while `LP-Guardian` was running, the battery lasted more than 24 hours for Galaxy Nexus (Fig. 2.7 – right) and more than 60 hours for both Samsung Galaxy S3 (Fig. 2.8 – right) and S4 (Fig. 2.9 – right). Actually, `LP-Guardian` doesn't do any processing in the background except for location acquisition to maintain the mobility model. All other processes are invoked in response to a location request by the app.

2.7.2 Privacy

In Section 2.4, we proposed a set of mechanisms to cope with the identification and tracking threats. In what follows, we evaluate the privacy protection mechanisms and their impact on quality of service.

2.7.2.1 Datasets

We evaluated `LP-Guardian` over three datasets that contain app usage information for 100 users over a period varying between 1 week and 1 year in three different cities. The first dataset includes 25 Android users (running Android 4.0.3 or higher) that we recruited from our institution over the period of 8 months. We collected app sessions and whether the app collected location while running. The second dataset consists of 49 Android users whom we recruited through PhoneLab [80], a smartphone measurement testbed from the State University of New York, Buffalo. We also collected app sessions from these users over 4 months. Finally, we relied on the dataset from the LiveLab project from Rice University [81]. They collected app sessions from 30 iPhone users for a period of one year. The subsequent analysis assumes a worst-case scenario of an app accessing location whenever it runs and then sharing it with the service provider.

2.7.2.2 Fingerprinting Protection

We evaluate the loss in app functionality when `LP-Guardian` is applied in two scenarios: a conservative privacy requirement of p_{min} = 0.0005, and more relaxed one of 0.05. In what follows, we report the percentage of sessions in which app functionality is negatively affected. For every scenario, we report the distribution of the percentage loss of sessions for every app–user combination. We report this distribution when the mechanism is applied plainly, when we add dummy queries, and when we add the mobility data from the US census data [74].

Figs. 2.10–2.12 show the potential loss in app functionality for each of the three datasets, ours, LiveLab's and PhoneLab's datasets consecutively. We can draw the following conclusions from observing these figures.

- More relaxed privacy constraints will enable releasing more of the user's location (comparing the left and right plots in every figure).

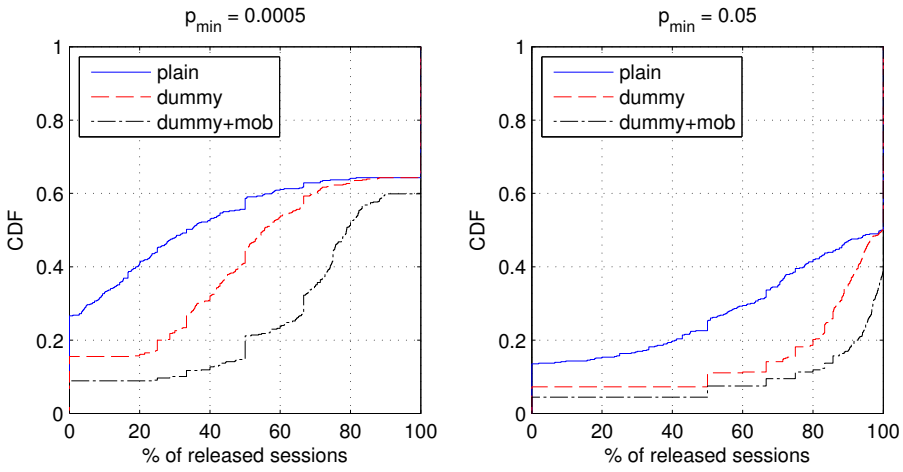


Figure 2.10: The distribution of the released sessions for our dataset

- Adding dummy locations and including the mobility data greatly increases the number of released sessions (comparing the dashed and dotted lines in every plot).

- Users with more diversity in their mobility patterns (visit more places) enjoy naturally higher privacy protection and can thus release more sessions (comparing our and LiveLab's datasets with PhoneLab's).

- The number of unreleased sessions with potential loss in functionality is low. More than 60% of the sessions are released for more than 60% of the users in the more privacy-constraint scenario of $p_{min} = 0.0005$.

- Although not shown in the figures, most of the unreleased sessions belong to top visited places (mostly home and work) for each user. These places are more critical to the user's identity.

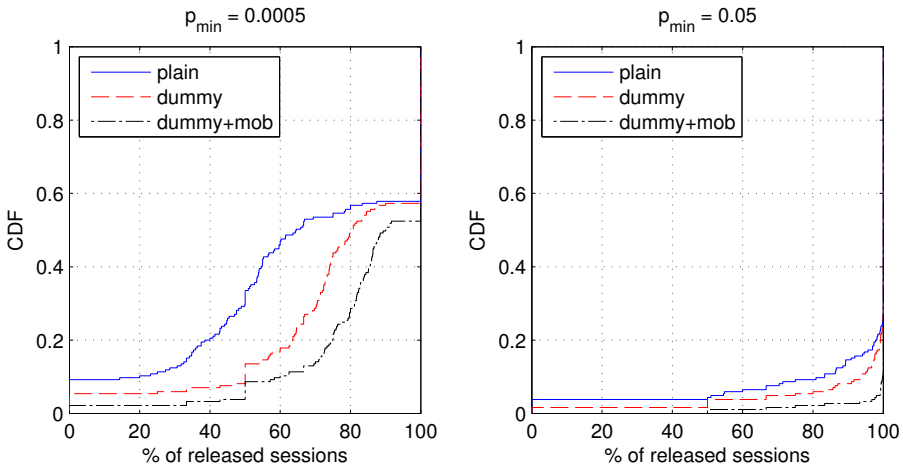


Figure 2.11: The distribution of the released sessions for LiveLab's dataset

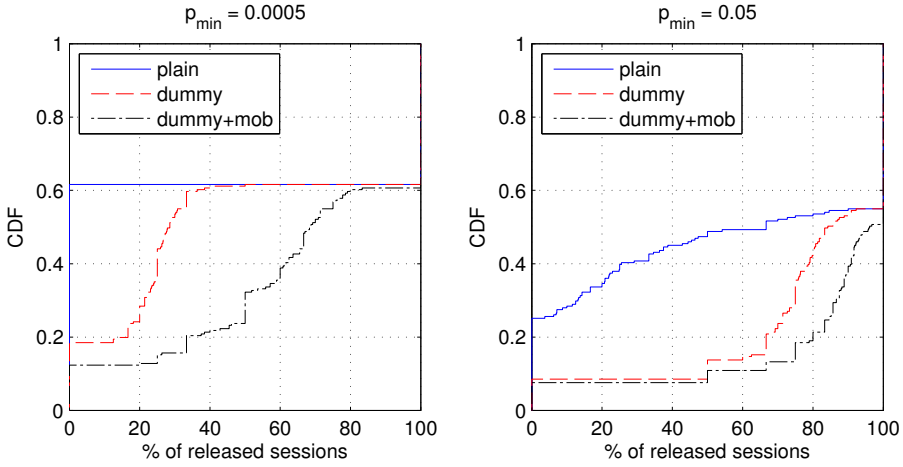


Figure 2.12: The distribution of the released sessions for PhoneLab's dataset

2.7.2.3 Tracking

As for the tracking threat, our mechanism blocks location access in the background, which leaves tracking limited to foreground sessions that are sporadic (mostly invoked at most once a day), short (average session length less than 5 minutes) and invoked from the same location for more than 98% of the time. Furthermore, our high-level privacy protection schemes hide locations in some sessions that are critical to the user's privacy. However, as noticed above, a considerable number of the sessions will be released. We evaluated the tracking threat each app could pose through the time tracked per day (similar to the time-to-confusion metric [82]). We evaluate the total time (in seconds) per day during

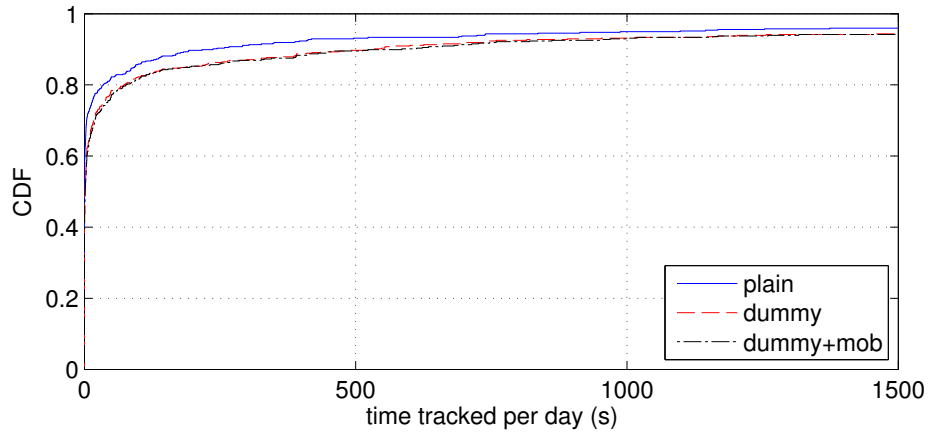


Figure 2.13: The distribution of the time tracked per day metric for our dataset

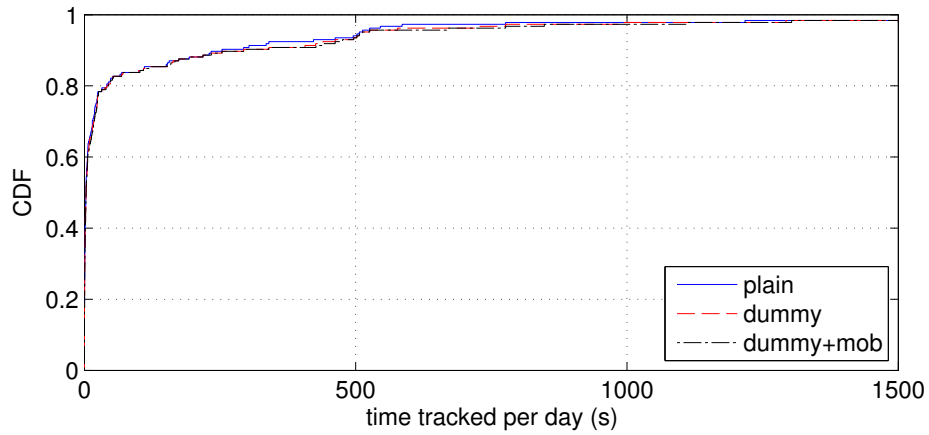


Figure 2.14: The distribution of the time tracked per day metric for LiveLab's dataset

which an app receives accurate location updates. We focus only on the case where most of the sessions will be released (p_{min} = 0.05) as it constitutes the worst-case scenario in terms of tracking.

Figs. 2.13–2.15 show the distribution of the time tracked per day for each of the three datasets (ours, LiveLab, and PhoneLab respectively). In all three plots, the approach with dummy and mobility data (labeled dummy+mob) releases the most of the sessions and thus has the most tracking threat. In all of the cases, 90% of the user–app combinations have tracked time per day of less than 500sec/day; most apps cannot track the user for more than 8 minutes a day. It is important to note that most of the foreground sessions happen when the user is stationary, which prevents the adversary from tracking the user while

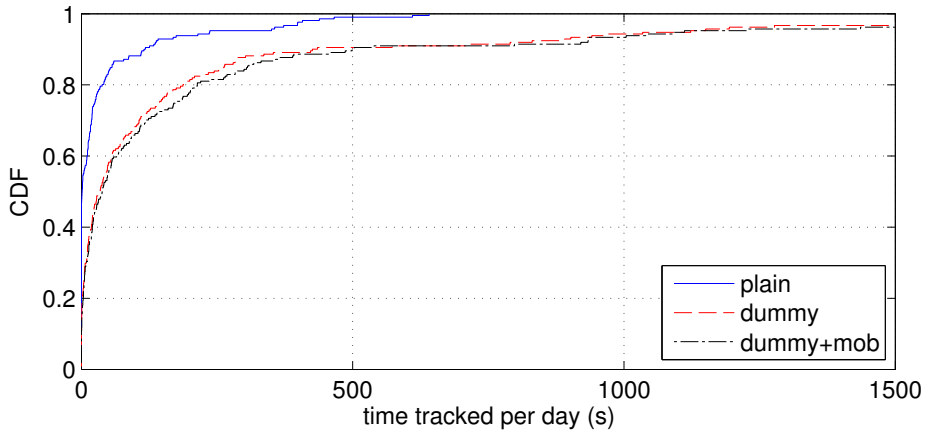


Figure 2.15: The distribution of the time tracked per day metric for PhoneLab's dataset

moving. This limits tracking to counting the places that the user visits. `LP-Guardian`'s identification and profiling prevention schemes handle threats resulting from the inference attacks based on the patterns of visited places.

2.7.3 User Study

`LP-Guardian` achieves theoretical guarantees on privacy with an inevitable loss of quality of service, albeit kept to a minimum. Most of these hidden sessions belong to home and work locations. To assess the user's perception of such loss in app functionality, we asked the participants (same as those of Section 2.3) whether they could accommodate reduced functionality of their apps from home and work locations for six classes of apps.

- Geo-search (e.g., Yelp): We asked participants (107 have such an app installed) about the level of comfort (scale 1–5) if they receive inaccurate search results while performing a search from either home or work. The majority of the participants (57%) indicated a comfort level larger than or equal to 3.

- Social networking (e.g., Facebook): We asked the participants (156 have such an app installed) whether they share location while being either at home or work. Most of the participants (62%) answered that they do not, 29% answered that would sometimes invoke this feature from either places. Also, 82% of the participants answered that they have **no** problem with sharing a city-level location instead of the actual home or work locations. The rest of participants either had no idea (5%), or prefer sharing the actual location (13%).

- Messaging/chatting (e.g., Whatsapp): The participants who reported installing such apps (123) do not normally invoke the location sharing feature from home and work. Only 6% perform such location sharing regularly, and 14% perform it sometimes. Also, most participants (78%) indicated that they have no problem in sharing a city-level location instead of home or work actual locations.

- Sports/fitness tracking (e.g., Runkeeper): Users rely on these apps to monitor their exercising activity. Thirty participants have reported installing and running a fitness tracking app. Only 6% of the participants care about viewing their actual tracks,

while the rest either care more about the distance (36%) or both the distance and viewing the tracks (58%).

- Gaming (e.g., Angry Birds): Most of these apps access location to feed A&A libraries in order to generate revenue. Of the participants who play games on their smartphones (131/180), 88% reported that their gaming experience will not be different despite possible location anonymization.

- Weather (e.g., Weather Bug): More than half the participants reported that weather information will not differ inside the same city, indicating that coarsening location information supplied for weather apps will not hinder their functionality.

It is evident from the survey that users generally will not mind some loss of app functionality at home and work. This is actually expected as users would rely more on location based functionality at unfamiliar places as opposed to places that they live or work at. For example, the user study shows that users would have no problem searching for relatively far places from their home or work locations when utilizing geo-search apps. Also, users seldom share their home or work locations while using social networking or chatting apps. Moreover, users are relatively open to the possibility of loss of quality of service when using sports tracking apps. In summary, `LP-Guardian` provides users with privacy at a tolerable loss of app functionality.

2.8 Conclusion & Future Work

In this chapter, we proposed `LP-Guardian`, a novel location privacy protection framework for Android smartphone users that is practical, effective, and efficient.

- *Practical*: We fully implemented `LP-Guardian` on Android 4.3. It does not break the functionality of the apps, and is not difficult to deploy. `LP-Guardian` also incurs acceptable energy and delay overheads.

- *Effective*: `LP-Guardian` addresses location privacy threats over three levels. It addresses the tracking threats through reducing the time tracked per day. It addresses the profiling threat by enabling the user to hide sensitive places. Finally, it addresses identification threats through a novel mechanism that makes the user's mobility pattern indistinguishable from a set of theoretical individuals satisfying a minimum constraint on all the places the user visits.

- *Efficient*: `LP-Guardian` provides privacy guarantees at a tolerable loss in app functionality. User's location is hidden for a small number of sessions and at places where location-based functionality is less needed.

In the next chapter, we pursue the deployment challenges related to location privacy protection. In the longer term, we are planning to deploy `LP-Guardian` in the phones of a set of diverse participants with limited technical background for a period of six months. This usability study will allow us to fine-tune `LP-Guardian` and identify areas where the privacy–usability tradeoff could be improved. Finally, we will consider incorporating `LP-Guardian` as a part of a custom ROM (e.g., CyanogenMod) to reach a larger audience.

CHAPTER III

LP-Doctor

3.1 Introduction

Mobile users are increasingly aware of the privacy threats caused by apps' access of their location [60, 59]. According to recent studies [61, 62, 59], users are also taking measures against these threats ranging from changing the way they run apps to disabling location services all together on their mobile devices. How to mitigate location-privacy threats has also been researched for some time. Researchers have proposed and even implemented location-privacy protection mechanisms (LPPMs) for mobile devices [52, 39, 40, 38]. However, few of them have been deployed as they require app or system-level modifications, both of which are unappealing/unrealistic to the ordinary users.

In Chapter II, we presented `LP-Guardian` which enacts privacy protection for smartphone users but requires modifying the Android platform. In this chapter, we investigate pushing privacy protection to the user-level by leveraging OS-based location controls. In particular, we propose `LP-Doctor`, a user-level tool, that requires no modification to the underlying operating system.

Faced with location-privacy threats, users are left only with whatever controls the apps and OSes provide. Some, but not all, apps allow the users to control their location access. OSes have been improving on this front. iOS includes a new permission to authorize location access in the background, or when the app is not actively used. Also, iOS, Windows

OS, and Blackberry (Android to follow suit) utilize per-app location-access permissions. The user authorizes location access at the very first time an app accesses his/her location and has the option to change this decision for every subsequent app invocation. We want to answer two important questions related to this: *(i) are these controls effective in protecting the user's location privacy and (ii) if not, how can they be improved at the user level without modifying any app or the underlying OS?*

To answer these questions, we must understand the location-privacy threats posed by mobile apps. This consists of understanding the apps' location-access patterns and their usage patterns. For this, we instrumented and analyzed the top 1165 downloaded free apps (that require location-access permissions) from Google Play to study their location-access patterns. We also studied the behavior of Advertisement and Analytics (A&A) libraries, such as Flurry, embedded in the apps that might access location. We analyzed only those apps/libraries that access location through Android's official location APIs. While some apps/libraries might circumvent the OS in accessing location, it is an orthogonal problem to that addressed in this chapter.

We then analyzed the users' app-usage patterns by utilizing three independent datasets. First, we collected and analyzed app-tagged location traces through a 10-month data collection campaign (Jan. 2013—Nov. 2013) for 24 Android smartphone users. Second, we recruited 95 Android users through PhoneLab [80], a smartphone measurement testbed at New York State University at Buffalo, for 4 months. Finally, we utilized the dataset from Livelab at Rice University [83] that contains app-usage and location traces for 34 iPhone users for over a year.

Ultimately, we were able to evaluate the privacy threats posed by 425 apps and 77 third-party libraries. 70% of the apps are found to have the potential of posing profiling threats that have not yet been adequately studied or addressed before [84, 28, 46, 85]. Moreover, the A&A libraries pose significant profiling threats on more than 80% of the users as they aggregate location information from multiple apps. Most of the users are unaware of these

threats as they cannot keep track of exposure of their location information. The issue becomes more problematic in the case of A&A libraries where users are oblivious to which apps these libraries are packed in and whether they are receiving location updates.

Given the nature of the threats, we studied the effectiveness of the existing OS controls. We found that these controls are capable of thwarting only a fraction of the underlying privacy threats, especially tracking threats. As for profiling, the user only has the options of either blocking or allowing location access. These two options come at either of the two extremes of the privacy–utility spectrum: the user either enjoys full privacy with no utility, or full utility with no privacy. As for A&A libraries, location accesses from a majority of the apps must be blocked to thwart the location-privacy threats caused by these libraries.

The main problem arises from the user's inability to exercise fine-grained control on when an app should receive a location update. The interface provided by existing controls makes it hard for the user to enforce location-access control on a per visited place/session basis. Even if the user can dynamically change the control of location access, s/he cannot estimate the privacy threats at runtime. The location-privacy threat is a function of the current location along with previously released locations. This makes it difficult to estimate the threat for apps and even harder for A&A libraries.

To fill this gap, we propose `LP-Doctor`, a user-level app, to protect the location privacy of smartphone users, which offers three salient features. First, `LP-Doctor` evaluates the privacy threat that the app might pose before launching it. If launching the app from the current location poses a threat, then it acts to protect the user's privacy. It also warns the user of the potential threat in a non-intrusive manner. Second, `LP-Doctor` is a *user-level* app and does not require any modification to the underlying OS or other apps. It acts as a control knob for the underlying OS tools. Third, `LP-Doctor` lets the user control, for each app, the privacy–utility tradeoff by adjusting the protection level while running the app.

`LP-Doctor`, however, fails to thwart the tracking threat posed by mobile apps con-

sistently accessing user's location in the background. This highlights a privacy versus usability trade-off between `LP-Guardian` and `LP-Doctor`. `LP-Guardian` requires modification to the underlying operating system and can offer stronger protection. On the other hand, `LP-Doctor` sacrifices tracking protection in favor of a usable implementation. Nevertheless, we show that the vast majority of the apps do not engage in background location tracking. Also, a native background location access control (such as the one offered by iOS) can complement the operation of `LP-Doctor`.

We implemented `LP-Doctor` as an Android app that can be downloaded from Google Play. The privacy protection that `LP-Doctor` provides comes at a minimal performance overhead. We recruited 227 participants through Amazon Mechanical Turk and asked them to download and use `LP-Doctor` from Google Play. The overwhelming majority of the participants reported little effect on the quality of service and user experience. More than 77% of the participants indicated that they would install `LP-Doctor` to protect their location privacy.

In summary, we make the following main contributions:

- The first location data collection campaign of its kind to measure, analyze, and model location-privacy threats from the apps' perspectives (Sections 3.3–3.6);
- Evaluation of the effectiveness of OS's location privacy controls by anatomizing the location-privacy threats posed by the apps (Sections 3.7–3.8);
- Design, implementation and evaluation of a novel user-level app, `LP-Doctor`, based on our analysis to fill the gaps in existing controls and improve their effectiveness (Section 3.9).

3.2 Related Work

App-Based Studies: To the best of our knowledge, this is the first attempt to quantify and model location privacy from the apps' perspective. Researchers already concluded

that many mobile apps and A&A libraries leak location information about the users to the cloud [86, 87, 88]. These efforts are complimentary to ours; we study the quantity and quality of location information that the apps and libraries locally gather while assuming that they may leak this information outside the device.

Analysis of Location Privacy: Influenced by existing location datasets (vehicular traces, cellular traces, etc.), most of the existing studies view location privacy in smartphones as if there were only one app *continuously* accessing a user's location [26, 28, 33, 27, 44, 45, 46]. Researchers also proposed mechanisms [49, 35, 33] (their effectiveness analyzed by Shokri *et al.* [89]) to protect against the resulting tracking-privacy threats. Such mechanisms have shown to be ineffective in thwarting the profiling threats [28] which are more prevalent as we will show later.

Researchers started considering sporadic location-access patterns as a source of location-privacy threat that calls for a different treatment than the continuous case [90]. Still, existing studies focus mostly on the tracking threat [51, 66]. The only exception to this is the work by Freudiger *et al.* [84]. They assessed the erosion of the user's privacy from sporadic location accesses as the portion of the PoIs identified after downsampling the continuous location trace. In this chapter, we propose a formal metric to model the profiling threats. Also, we show that an app's location-access behavior cannot be modeled as simply downsampling the user's mobility.

Location-Privacy Protection Proposals: Several solutions have been proposed to protect mobile users' location privacy. MockDroid [52] allows for blocking apps' location access to protect the user's location privacy. LP-Guardian [60] is another system aiming at protecting the user's location privacy by incorporating a myriad of mechanisms. Both systems require platform modifications, hindering their deployment. Other mechanisms, such as Caché[39] and the work by Micinski *et al.* [40], provide apps with coarsened locations but require modifications to the apps. Koi [38] proposed a location privacy enhancing system that utilizes a cloud service, but requires developers to use a different API to access

location. Apps on Google Play such as PlaceMask and Fake GPS Location Spoofer rely on the user to manually feed apps with fake locations, which reduce their usability.

Finally, researchers have proposed improved permission models for Android [91, 92]. In their models, the users are aware of how much the apps access their location and have the choice to enable/disable location access for each app (AppOps provided such functionality in Android 4.3). `LP-Doctor` improves on these in three ways. First, it provides a privacy model that maps each app's location access to a privacy metric. This model includes more information than just the number of location accesses by the app. Second, `LP-Doctor` makes some decisions on behalf of the users to avoid interrupting their tasks and to make privacy protection more usable. Third, `LP-Doctor` employs *per-session* location-access granularity which achieves a better privacy–utility tradeoff.

3.3 Background and Data Collection

To study the efficacy of location-access controls of different mobile OSes, we had to first analyze location-privacy threats from the apps' perspectives. This includes studying how different apps collect the user's location. We conduct a data collection campaign to achieve this using the Android platform. Our results, however, can be generalized to other mobile platforms like iOS.

3.3.1 Location-Access Controls

Each mobile platform provides users with a set of location-access controls to mitigate possible location-privacy threats. Android (prior to Android M) provides a one-time permission model that allows users to authorize location access. Once the user approves the permission list (Fig. 3.1–left) for the app, it is installed and the permissions cannot be revoked. It also provides a global location knob (Fig. 3.1–right) to control location services. The user cannot exercise per-app location-access control.

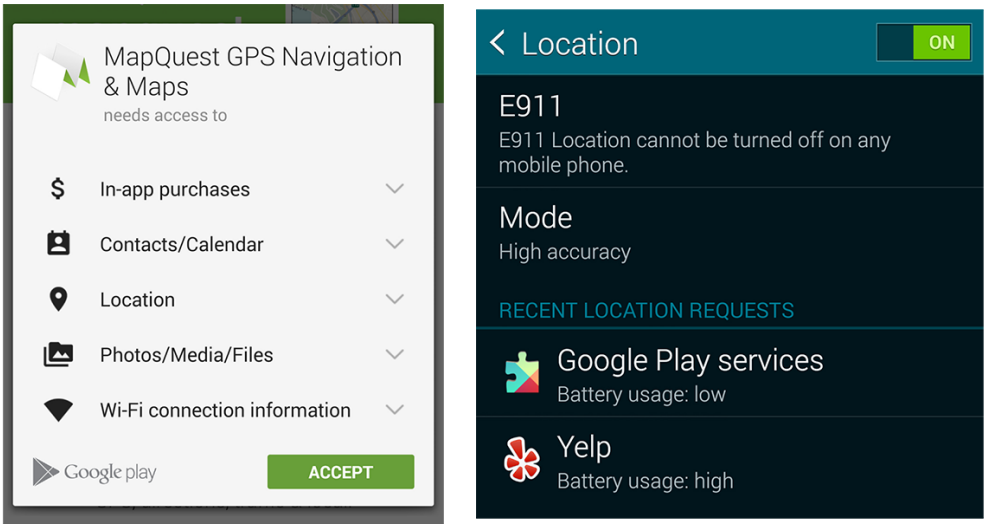


Figure 3.1: Android's permission list (left) and location settings (right).

Other platforms, such as Blackberry OS and iOS, provide finer-grained location permissions. Each app has a settings menu (Fig. 3.2–left) that indicates the resources it is allowed to access, including location. The user can at any point of time revoke resource access by any app. The first time an app accesses location, the OS prompts the user to authorize location access for the app in the current and future sessions (Fig. 3.2–right). Also, Google, starting from Android M, will provide a similar permission model (an evolution of the previously deployed AppOps in Android 4.3) to control access of location and other resources. At present, iOS provides the users with an additional option to authorize location access in the background to prevent apps from tracking users.

In the rest of this chapter, we study the following controls: (1) one-time location permissions, (2) authorization of location access in the background, and (3) finer-grained per-app permissions.

3.3.2 System Model

We study location-privacy threats through apps and A&A libraries that access the user's location. These apps and libraries then provide the service, and keep the location records indexed by a user ID, such as MAC address, Android ID, IMEI, etc.

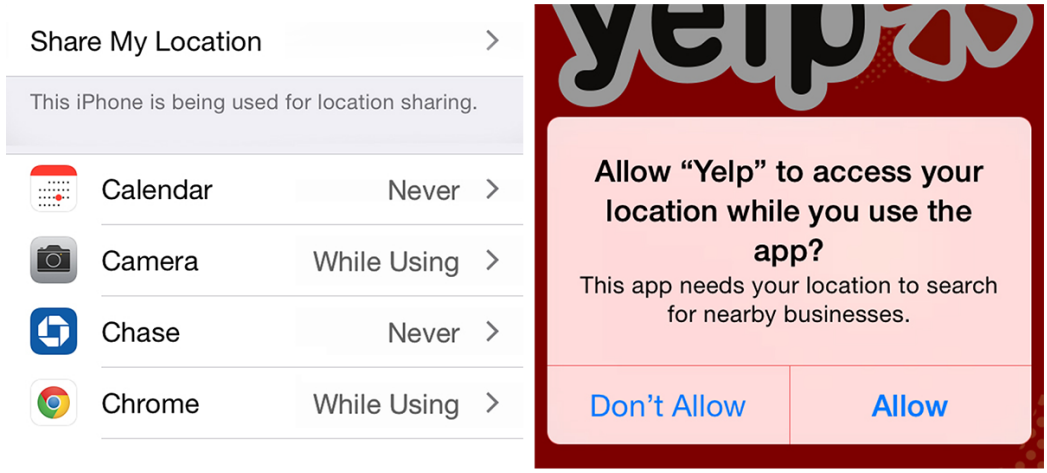


Figure 3.2: iOS’s location settings (left) and prompts (right).

We assume that the app/library communicates all of the user’s location samples to the service provider.[1] This allows us to model the location-privacy threats caused by apps/libraries in the worst-case scenario. The app is the only means by which the service provider can collect the user’s location updates. We do not consider cases where the service provider obtains the user’s location via side channels other than the official API, e.g., an app reads the nearby access points and sends them to a localization service, such as skyhook.

We preclude system and browsing apps from our study for the following reasons. System apps are part of the OS that already has access to the user’s location all the time. Hence, analyzing their privacy implications isn’t very informative. As for the browsing apps, the location sink might be a visited website as well as the browser itself. We decided not to monitor the user’s web history during the data collection for privacy concerns. Also, app-usage patterns differ from browsing patterns. The conclusions derived for the former do not necessarily translate to those for the latter.

[1] We refer to both the app developers and A&A agencies as the service provider.

3.3.3 App and A&A libraries Analysis

In February 2014, we downloaded the top 100 apps of each of Google Play's 27 app categories. We were left with 2588 unique apps, of which 1165 apps request location permissions. We then instrumented Android to intercept every location access invoked by both the app and the packed A&A libraries.

The main goal of this analysis was to unravel the situations in which an app accesses location and whether it feeds a packed A&A library. In Android, the app could be running in the foreground, cached in the background, or as a service. Using a real device, we ran every app in foreground, moved it to background, and checked if it forked a service, while recording its location requests.

Apps running in the foreground can access location spontaneously or in response to some UI event. So, we ran every app in two modes. In the first mode, the app runs for a predefined period of time and then closes, while in the second, we manually interact with each app to trigger the location-based functionality. Finally, we analyzed the functionality of every app and the required location granularity to achieve this functionality.

3.3.4 Data Collection

As will be evident in Section 3.4, the app-usage pattern is instrumental in determining the underlying location-privacy threats. We collected the app-usage data using an app that we developed and published on Google Play. Our study was deemed as not-requiring an IRB oversight by the IRB at our institution; all the data we collected is anonymous. Also, we clustered the participants' location on the device to extract their visited places. We define the "place" as a center location with a radius of 50m and a minimum visit time of 5 min. Then, we logged place IDs instead of absolute location samples to further protect the participants' privacy.

PhoneLab: PhoneLab [80] is a testbed, deployed at the NY State University at Buffalo, composed of 288 smartphone users. PhoneLab aims to free the researchers from recruiting

participants by providing a diverse set of participants, which leads to stronger conclusions.

We recruited 95 participants to download and run our app for the period between February 2014 and June 2014. We collected detailed usage information for 625 apps, of which 218 had location permissions and were also part of the apps inspected in the app-analysis stage.

University of Michigan: The second set consists of 24 participants whom we recruited through personal relations and class announcements. We launched this study from January 2013 till November 2013, with the participation period per user varying between 1 week and 10 months. From this set, we collected usage data of 256 location-aware apps.

We also collected location access patterns of some apps from a subset of the participants. We handed 11 participants Galaxy Nexus devices with an instrumented Android (4.1.2) that recorded app-tagged location accesses. We measured how frequently do ordinary users invoke location-based functionality of apps that do not spontaneously access location (e.g., Whatsapp).

LiveLab: Finally, we utilize the Livelab dataset [83] from Rice University. This dataset contains the app usage and mobility records for 34 iPhone users over the course of a year (2010). We post-processed this dataset to map app-usage records to the location where the apps were invoked. We only considered those apps that overlapped with our Android dataset (35 apps).

3.4 Location-Access Patterns

We address the location-access patterns by analyzing how different apps collect location information while running in foreground and background. The former represents the state where the user actively interacts with the app, while the latter represents the case where the app runs in the background either as cached by Android or as a persistent service.

As evident from Table 3.1, 74% of the apps solely access location when running in the foreground, while only 3% continuously access the user's location in the background.

Table 3.1: Location-access patterns for smartphone apps according to Android location permissions

	Fore. (%)	Cached (%)	Back. (%)	None (%)	Gran. Coarse (%)
Coarse	71	6	1	22	100
Fine	74	14	4	12	48
All	74	12	3	14	66

Table 3.2: Location-access patterns for A&A libraries

Total	No Location Access	App Feeds Location	Auto Location Access		
			Coarse	Fine	Both
77	22	17	3	2	33

Around 70% of the apps accessing location in the foreground spontaneously perform such access preceding any user interaction. Examples of these apps include Angry Birds, Yelp, Airbnb, etc.

Android caches the app when the user exits it; depending on the app's behavior it might still access location; only 12% of the apps access the user's location when they are cached. Interestingly, for 14% of the apps, we didn't find any evidence that they access location in any state.

We also analyzed the location-based functionality of every app and the required location granularity to achieve such functionality. We focused on two location granularity levels: *fine* and *coarse*. A fine location sample is one with block-level granularity or higher, while coarse location is that with zipcode-level granularity or lower. We manually interacted with each app to assess the change in its functionality while feeding it locations with different granularity. We show the percentage of the apps that can accommodate coarse location without noticeable loss of app functionality in Table 3.1 under the column titled *Gran. Coarse*. One can notice that apps abuse the location permissions: 48% of the apps requesting fine location permissions can accommodate locations with coarser granularity without loss of functionality.

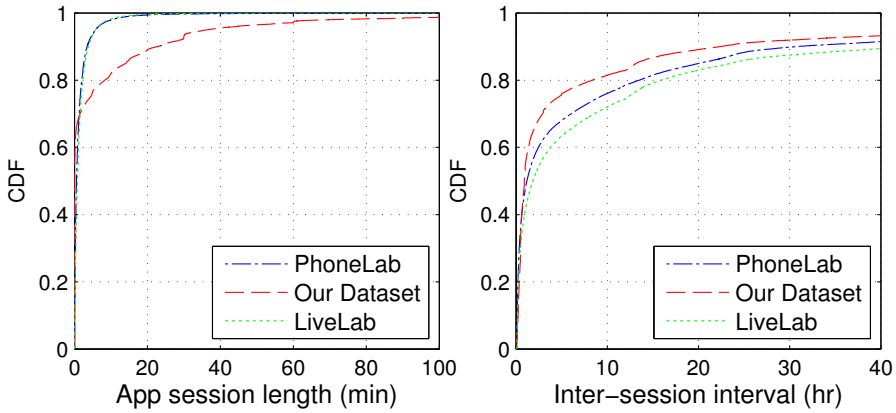


Figure 3.3: The distribution of app session lengths (left) and inter-session intervals (right) for the three datasets.

Finally, we analyzed the packed A&A libraries in these apps. We were able to identify 77 of such libraries packed in these apps. Table 3.2 shows basic statistics about these libraries. Most (more than 70%) libraries require location access where some are fed location from the apps (22%). The rest of the libraries automatically access location where 3 of them require coarse location permissions, 2 require fine permissions, and the rest do not specify a permission. Also, these libraries are included within more than one location-aware app giving them the ability to track the user's location beyond what a single app can do. For example, of 1165 analyzed apps, Google Ads is packed within 499 apps, Flurry within 325 apps, Medialets within 35 apps, etc.

3.5 App-Usage Patterns

As apps mostly access users' location in the foreground, the app-usage patterns (the way that users invoke different apps) help determine how much location information each app collects. Apps are shown to sporadically sample the user's location based on two facts. First, an app session is equivalent to the place visited during the session. Second, apps' inter-session intervals follow a Pareto-law distribution.

For foreground apps, we define a session as a single app invocation—the period of time

in which a user runs the app then exits it. The session lengths are not long enough to cover more than one place the user visits, where 80% of these app sessions are shorter than 10 minutes (the left plot of Fig. 3.3). We confirmed this from our PhoneLab dataset; 98% of the app sessions started and ended at the same place.

This allows for collapsing an app session into one location-access event. It doesn't matter what frequency the app polls the user's location with. As long as the app requests the user's location at least once, while it is running in the foreground, it will infer that the user visited that location. We thus ignore the location-access frequency of foreground apps, and instead focus on the app-usage patterns.

We define the inter-session time as the interval separating different invocations (sessions) of the same app by the same user. The right plot of Fig. 3.3 shows the distribution of the inter-session intervals for the three datasets. More than 50% of the app sessions were separated by at least one hour.

We also found that the inter-session intervals follow a Pareto-law distribution rather than a uniform distribution. This indicates that apps do not sample the user's location uniformly, indicating that existing models for apps' location access do not match their actual behavior.

Fig. 3.4 shows the distribution of the inter-session intervals of a user running Facebook. It is evident that the distribution of the inter-session intervals decays linearly with respect to the increase of inter-session intervals. We observed a similar trend with all other apps. This suggests that the data decays according to a Pareto law (QQ plot in Fig. 3.4). We followed the guidelines outlined by Clauset *et al.* [93] to fit the data to the truncated Pareto distribution. Three parameters (L, H, and α) define the truncated Pareto law distribution:

$$p_X(x) \begin{cases} \frac{(-\alpha-1)L^{-\alpha-1}x^{\alpha}}{1-\left(\frac{L}{H}\right)^{-\alpha-1}} & \text{if } L \leq x \leq H \\ 0 & \text{otherwise.} \end{cases}$$

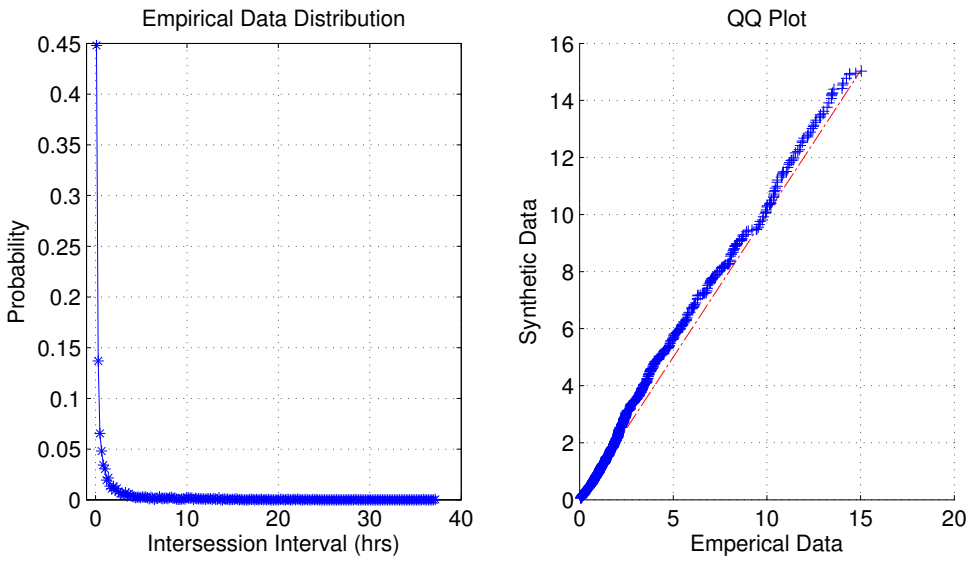


Figure 3.4: The distribution of the inter-session times for Facebook in Livelab dataset (left), and the QQ plot of this distribution versus a Pareto law distribution (right).

After fitting the data, more than 97% of the app-usage models are found to have α between -1 and -1.5. According to Vagna *et al.* [94], Pareto law fits different human activity models with α between -1 and -2.

3.6 Privacy Model

Here we model the privacy threats caused by mobile apps'/libraries' access of the user's location.

3.6.1 Preliminaries

Below we describe the models of user mobility, app-usage, and adversaries that we will use throughout the chapter.

User Mobility Model: We assume there is a region (e.g., city) of interest which includes set of places that the user can visit. So, a domain of interest is represented by the set Pl of all the places available in that domain: $Pl = \{pl_1, pl_2, pl_3 \ldots\}$. Under this model, the user visits a set of places, $U_{Pl} \subseteq Pl$, as part of his/her daily life, spends time at pl_i running

some apps and then moves to another place pl_j. We alternatively refer to these places as the user's *Points of Interest* (PoIs).

We associate every place pl_i with a visit probability of p_i, reflecting the portion of time the user spends at pl_i. The user's *mobility profiles* are defined as the set, U_{pl}, of places s/he visited and the probability, p_i, of visit to each place. The mobility profile is unique to each user since a different user visits a different set of places with a different probability distribution [28].

App-Usage Model: In Section 3.5, we showed that each app session is equivalent to an observation of the user's visit to a place. The app accumulates observations of the set of places that the user visits. The app will eventually observe that a user visited a certain place pl_i for c_{pl_i} times. So, we view the app as a random process that samples the user's entire location trace and outputs a histogram of places of dimension $|U_{Pl}|$. Each bin in the histogram is the number of times, c_{pl_i}, the app observes the user at that specific place. The total number of visits is represented as $N = \sum_{i=1}^{|U_{Pl}|} c_{pl_i}$.

The histogram represents the app's view of the user's mobility. Most apps do not continuously monitor user's mobility as they do not access location in the background. As such, they cannot track users; the most these apps can get from a user is the histogram of the places s/he visited, which constitutes the source of location-privacy threats in this case.

Adversary Model: The adversary in our model is not necessarily a malicious entity seeking to steal the user's private information. It is rather a curious entity with possession of the user's location trace. The adversary will process and analyze these traces to infer more information about the user that allows for a more personalized service. This is referred to as *authentic apps* [95]. The objective of our analysis is to study the effect of the ordinary apps collecting location on the user's privacy.

Apps accessing location in the foreground cannot track the user (Section 3.8). So, the adversary seeks to profile the user based on locations s/he visited. We use the term *profiling* to represent the process of inferring more information about the user through

Table 3.3: The metrics used for evaluating the location privacy threats.

Metric	Description
PoI_{total}	Fraction of the user's PoIs
PoI_{part}	Fraction of the user's infrequently visited PoIs
$Prof_{cont}$	Distance between the user's histogram and mobility profile
$Prof_{bin}$	χ^2 test of the user's histogram fitting the mobility profile

the collected location data. The profiling can take place at multiple levels, ranging from identifying preferences all the way to revealing the user's identity. Instead of modeling the adversary's profiling methods/attacks, we quantify the amount of information that location data provides the adversary with. The intuition behind our analysis of the profiling threat is that the more descriptive the app's histogram of the actual user's mobility pattern, the higher the threat is.

3.6.2 Privacy Metrics

Table 3.3 summarizes the set of metrics that we utilize to quantify the privacy threats that each app poses from its location access. The simplest metric is the fraction of the users' PoIs the app can identify [84]. We evaluate this metric by looking at the apps' actual collected location traces, rather than a downsampled location trace. We will henceforth refer to this metric as PoI_{total}.

We also consider a variant of the metric (referred to as PoI_{part}) as the portion of the sensitive PoIs that the apps might identify. We define the sensitive PoIs as those that have a very low probability of being visited. These PoIs will exhibit abnormalities in the user's behavior. Research results in psychology [96, 97] indicated that people regard deviant (abnormal) behavior as being more private and sensitive. Places that an individual might visit that are not part of his/her regular patterns might leak a lot of information and are thus more sensitive in nature.

The histogram, as we mentioned before, is a sample of the user's mobility pattern. The

second aspect of the profiling is quantifying how descriptive of the user's mobility pattern (original distribution) the app's histogram (sample) is.

For the purpose of our analysis and the privacy-preserving tool we propose later, we need two types of metrics. The first is a *continuous* metric, $Prof_{cont}$, that quantifies the profiling threat as the distance between the original distribution (mobility profile) and the sample (app's histogram). The second is a *binary* metric, $Prof_{bin}$, that indicates whether a threat exists or not.

For $Prof_{cont}$, we use the KL-divergence [98] as a measure of the difference (in bits) between the histogram (H) and the user's mobility pattern. The K-L divergence is given by $D_{\mathrm{KL}}(H\|p) = \sum_{i=1}^{|U_{Pl}|} H(i)\ \ln \frac{H(i)}{p_i}$, where $H(i)$ is the probability of the user visiting place pl_i based on the histogram, while p_i is the probability of the user visiting that place based on his/her mobility profile. The lower (higher) the value of $Prof_{cont}$, the higher (lower) the threat will be since the distance between the histogram and mobility pattern will be smaller (larger).

$Prof_{cont}$ is not useful in identifying histograms that pose privacy threats. There is no intuitive way by which a threshold can separate values that pose threats and those not posing any threat. So, we need a criterion indicating whether or not a threat exists based on the app's histogram. We use Pearson's Chi-square goodness of fit test to meet this need. This test indicates if the observed sample differs from the original (theoretical) distribution. Specifically, it checks if the null hypothesis of the sample originating from an original distribution can be accepted or not.

The test statistic, in our context, is $\chi^2 = \sum_{i=1}^{|U_{Pl}|} \frac{(c_{pl_i} - E_i)^2}{E_i}$ where $E_i = N.p_i$ is the expected number of visits to the place pl_i. The statistic converges to a Chi-squared distribution with $|U_{Pl}| - 1$ degrees of freedom when the null hypothesis holds. The test yields a p-value which if smaller than the significance level (α) then the null hypothesis can be rejected ($Prof_{bin} = 0$—no threat), else $Prof_{bin} = 1$, where null hypothesis cannot be rejected, indicating the existence of a threat. In Sections 3.7 and 3.8, we employ the widely-used

value of 0.05 as the significance level.

Here, we have to highlight the same issue of the second chapter of this thesis. Our modeling of the user mobility drops timing information which could leak additional information to the adversary. Nevertheless, the metrics adopted in this chapter can easily accommodate tagging each bin of the histogram with time information (time of the day or day of the week). Still, they (the metrics) cannot consider the order of user visits to different locations. Modeling privacy threats, while taking the ordering information into account, is studied in the next chapter of this thesis.

A&A libraries: can aggregate location information from the different apps in which they are packed and allowed to access location. We can thus view the histogram pertaining to an A&A library as the aggregate of the histograms of the apps in which the library is packed. We evaluate the same metrics for the aggregated histogram.

For the case of PoI_{total} and PoI_{part} metric, the aggregate histogram will be representative of the threat posed by the libraries. As for $Prof_{cont}$ and $Prof_{bin}$, we consider the aggregate histogram as well as the individual apps' histograms. The threat per library is the highest of that of the aggregate and individual histograms. The privacy threat posed by the library is at least as bad as that of any app that packs it in.

3.7 Anatomy

We now present the major findings from our measurement campaign. We analyze the location trace of each app and user, and hence, every data point in the subsequent plots belongs to an app–user combination. We constructed each app's histogram by overlaying its location-access pattern on its usage data for every user.

Privacy Threat Distribution: Fig. 3.5 shows the distributions of PoI_{total}, PoI_{part}, and $Prof_{cont}$ for both the apps and A&A libraries. As to PoI_{total}, most of the apps can identify at least 10% of the user's PoIs; while for 20% of the app–user combinations, apps were able to identify most of the user's PoIs. Apps cannot identify all of the user's PoIs for two

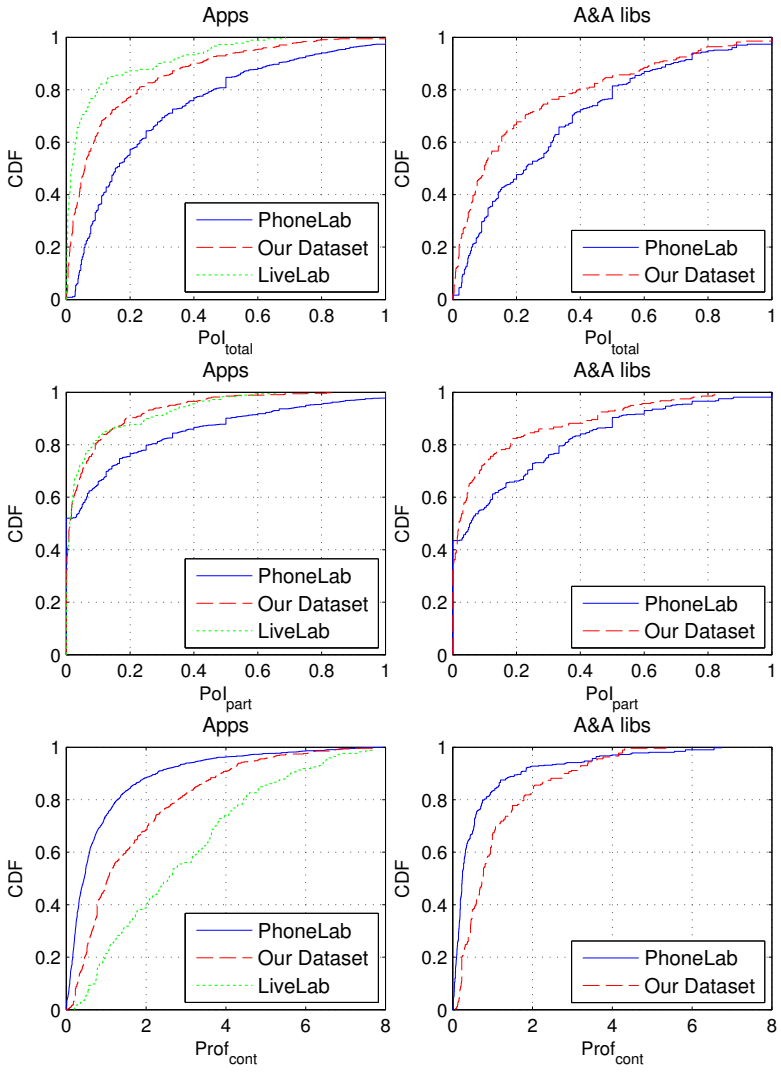


Figure 3.5: The distributions of PoI_{total} (top), PoI_{part} (middle), and $Prof_{cont}$ (bottom) for the apps (left) and A&A libraries (right) from our datasets.

reasons: (1) not all apps access the user's location every time, as highlighted in Section 3.4, and (2) users do not run their apps from every place they visit. On the other hand, A&A libraries can identify more of the user's PoIs, with most of the libraries identifying at least 20% of the user's PoIs. Moreover, as the middle plots of Fig. 3.5 indicate, around 30% of the apps were able to identify some of the user's sensitive (less frequently visited) PoIs. More importantly, A&A libraries were able to identify more of the user's sensitive PoIs, indicating the level of privacy threats they pose.

The two bottom plots of Fig. 3.5 show the distributions of the profiling metric $Prof_{cont}$

for the foreground apps in the three datasets. The lower the value of the metric, the higher the privacy threat is. There are two takeaways from these two plots. First, apps do pose significant privacy threats; the distance between the apps' histogram and the user's mobility pattern is less than 1 bit in 40% of the app–user combinations for the three datasets. The second observation has to do with the threat posed by A&A libraries. It is clear from the comparison of the left and right plots that these libraries pose considerably higher threats. In more than 80% of user–library combinations, the distance between the observed histograms and the user's mobility profile is less than 1 bit.

Apps tend to even pose higher identification threats. As evident from Fig. 3.5, some apps can identify a relatively minor portion of the user's mobility which might not be sufficient to fully profile the user. Nevertheless, the portion of PoIs tend to be those users frequently visit (e.g., home and work) which may suffice to identify them [28, 46, 20]. This might not be a serious issue for those apps, such as Facebook, that can learn the user's home and work from other methods. Other apps and libraries (e.g., Angry Birds), however, might infer the user's identity even when s/he anonymously uses them (without providing an identity or login information).

Fig. 3.5 also confirms our intuition in studying the location traces from the apps' perspective. If apps were to uniformly sample the user's mobility as has been assumed in literature, $Prof_{cont}$ should be mostly close to 0 (indicating no difference between the histogram and the mobility pattern), which is not the case.

Privacy Threats and App-Usage: We also evaluated the posed privacy threats vs. the app-usage rate as shown in Fig. 3.6. As evident from the plots, there is little correlation between the amount of posed threats and the app-usage rate. Apps that are used more frequently, do not necessarily pose higher threats, as user mobility, the app's location-access pattern, and the user's app-usage pattern affect the privacy threat.

With lower usage rates, both PoI_{total} and $Prof_{cont}$ vary significantly. Users with little diversity in their mobility pattern are likely to visit the same places more frequently. Even

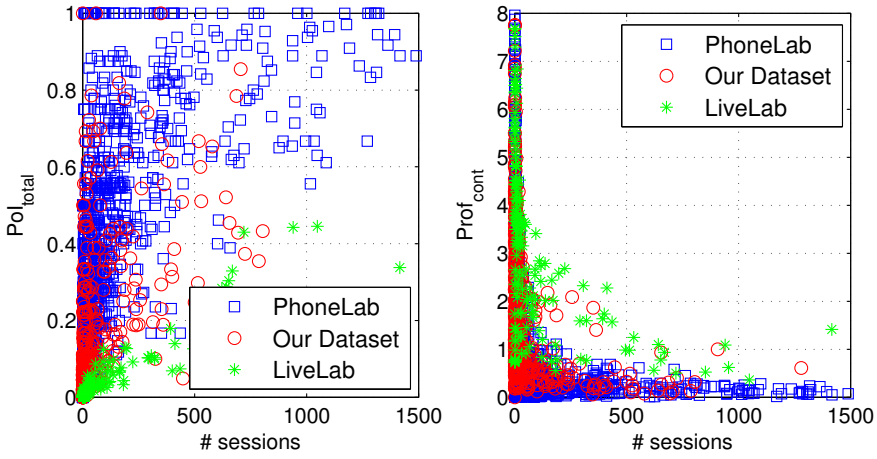


Figure 3.6: The distribution of PoI_{total} (left) and $Prof_{cont}$ (right) vs. the number of app sessions.

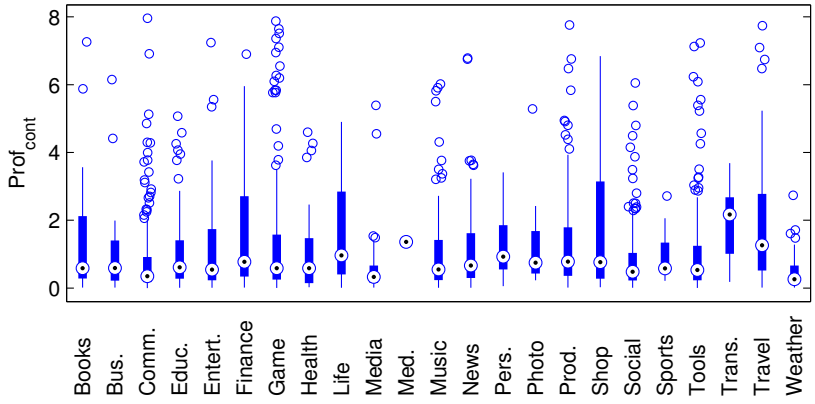


Figure 3.7: The distribution of $Prof_{cont}$ vs. app categories.

the same user could invoke apps differently; s/he uses some apps mostly at unfamiliar places (navigation apps), while using other apps more ubiquitously (gaming apps), thus enabling the apps to identify more of his/her PoIs.

Finally, we studied the distribution of the threat in relation to app categories. Fig. 3.7 shows that the threat level is fairly distributed across different app categories and the same category. This confirms, again, that privacy threats result from multiple sources and are a function of both apps and users. Some app categories, however, pose lower threats on average. For example, transportation apps (including navigation apps) pose lower threats as users tend to use from unfamiliar places.

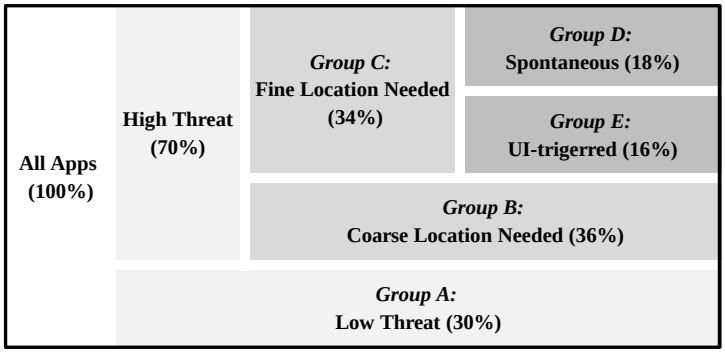


Figure 3.8: App categorization according to threat levels, location requirements, and location-access patterns.

Threat Layout: Given the three datasets, we were able to analyze the profiling threats as posed by 425 location-aware apps (Fig. 3.8). For this part, we use $Prof_{bin}$ metric to decide which apps pose privacy threats and those which do not. As apps pose different threats depending on the users, we counted an app as posing a threat if it poses a privacy threat to at least one user. Only a minority of the apps (30%) pose negligible threats.

The rest of the apps pose a varying degree of profiling threat. We analyzed their functionality: 52% of such apps do not require location with high granularity to provide location-based functionality. For these apps, a zipcode- or city-level granularity would be more than enough (weather apps, games). This leaves us with 34% of the apps that require block-level or higher location granularity to provide usable functionality. These apps either spontaneously access location (18%) or in response to a UI event (16%).

3.8 OS Controls

Having presented an anatomy for the location-privacy threats posed by mobile apps, we are now ready to evaluate the effectiveness of existing OSes' location access controls in thwarting these threats.

Global Location Permissions: Android's location permissions attempt to serve two purposes: notification and control. They notify the user that the app s/he is about to install

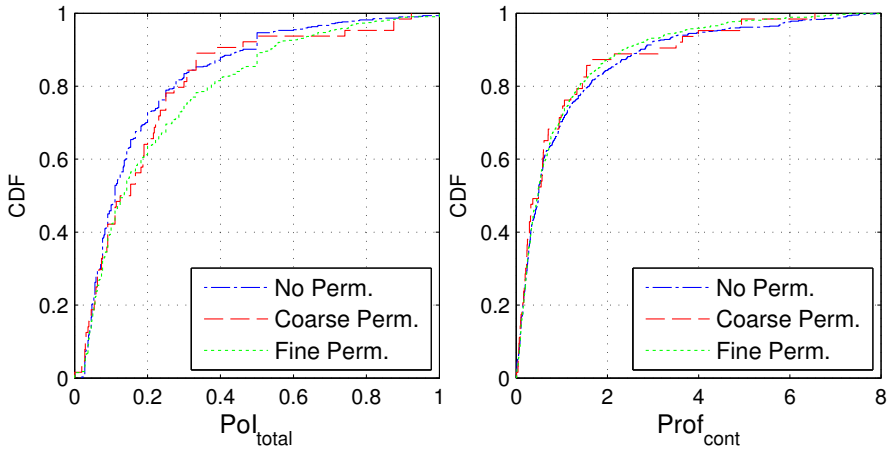


Figure 3.9: The distribution of PoI_{total} (left) and $Prof_{cont}$ (right) for PhoneLab apps with different permissions.

can access his/her location. Also, permissions aim to control the granularity by which apps access location. Apps with coarse-grained location permission can only access location with both low granularity and frequency.

Fig. 3.9 compares the profiling threats (PoI_{total} and $Prof_{cont}$) posed by apps with fine location permissions and those with coarse location permissions. It also plots the distribution of the privacy metrics for apps without location permissions assuming that they accessed location when running. While this might seems oblivious at a first glance, we aim to compare the location-based usage of apps with different location permissions. This allows us to study if the location permissions are effective as a notification mechanism so that users use apps from different places depending on the location permissions.

The apps with fine-grained location permissions exhibited very similar usage pattern to those apps without location access. The users ran the app from the same places regardless of whether they have location permissions or not. We conclude that this notification mechanism does little to alert users on potential privacy threats and has no effect on the app-usage behavior. Similar observations have also been made by others [61].

Almost a half of the apps (Table 3.1) that request fine-grained location permissions are found to be able to achieve the location-based functionality with coarser-granularity

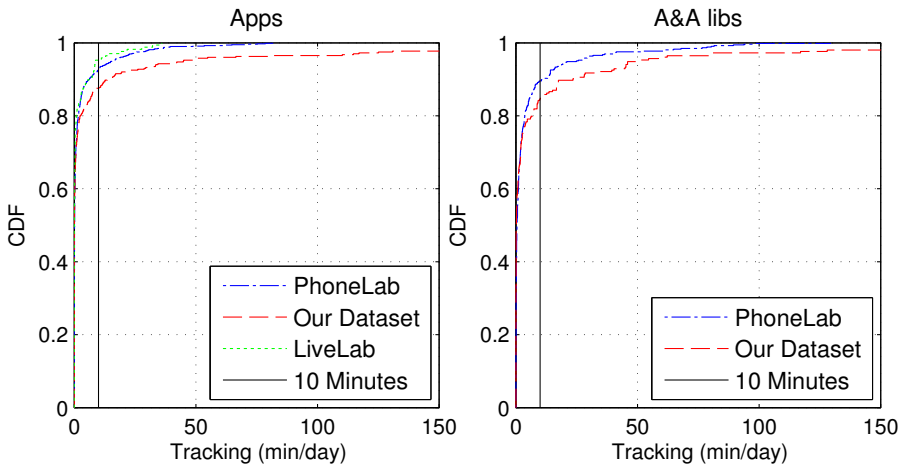


Figure 3.10: The distribution of the tracking threat posed by the foreground apps (left) and A&A libraries (right).

location. This suggests that apps abuse location permissions. If used appropriately, permissions can be effective in thwarting the threats resulting from apps' abuse of location access (∼40% of the apps — Group B — according to Fig. 3.8).

Background Location Access: Background location access is critical when it comes to tracking individuals. It enables comprehensive access to the user's mobility information including PoIs and frequent routes. Recently, iOS 8 introduced a new location permission that allows users to authorize location access in the background for apps on their devices.

This permission strikes a balance between privacy and QoS. We showed in Section 3.4 that apps rarely access location in the background. Thus, this option affects a very low portion of the user's apps, but is effective in terms of privacy protection, especially in thwarting tracking threats. We evaluated the tracking threat in terms of tracking time per day [82, 60] for the three datasets for foreground location access.

Fig. 3.10 (left) shows that in 90% of the app–user combinations, blocking background location access will limit the location exposure to less than 10 minutes a day (from foreground location access). The third-party libraries tend to pose slightly higher tracking threats than apps (Fig. 3.10 – right).

Per-app Location Permissions: To improve over static permissions, iOS enables the

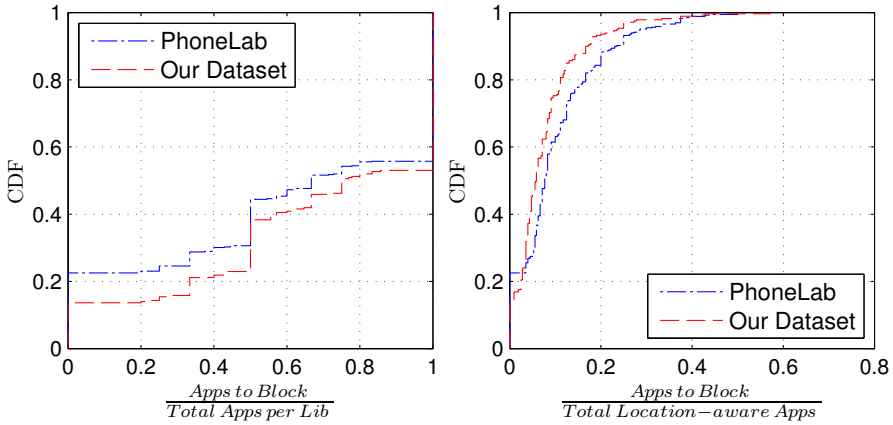


Figure 3.11: The fraction of the user's apps that must be blocked from accessing location to protect against privacy threats posed by A&A libraries.

user to allow/disallow location access on a per-app basis. The users gain two advantages from this model: (i) location access can be blocked for a subset, but not all, of the apps, and (ii) the apps retain some functionality even when the location access is blocked.

Even if the user trusts an app with location access, the app can still profile him/her through the places s/he visited (Groups D and E in Fig. 3.8). To combat these threats, the user has to either allow location access to fully exploit the app and lose privacy, or gain his/her privacy while losing the location-based app functionality. Currently, mobile platforms offer no middle ground to balance privacy and QoS requirements.

In Section 3.7, we showed that A&A libraries pose significant threats that users are completely unaware of as they access location from more than one app. The user cannot identify which apps s/he must disallow to access location in order to mitigate threats from third-party libraries. Fig. 3.11 shows the portion of the user's apps that must be disbarred from accessing location to thwart threats from packed A&A libraries. It turns out (left plot of Fig. 3.11) that in order to protect the user from privacy threats posed by a single library, at least 50–70% of the apps carrying the library must be disbarred from accessing location. This amounts to blocking location for more than 10% of the apps installed on the device.

In conclusion, a static permission model suffers serious limitations, blocking location access in the background is effective in mitigating the tracking threat but not the profiling

one, and per-app controls exhibit an unbalanced tradeoff between privacy and QoS. Also, they are ineffective against the threats caused by A&A libraries. Thus, a finer-grained location access control is required, allowing control for each app session depending on the context. Per-session location access control allows users to leverage better and more space in the privacy–QoS spectrum.

3.9 LP-Doctor

Users cannot utilize the existing controls to achieve per-session location-access controls for two reasons. First, these controls are coarse-grained (providing only per-app controls at best). For finer-level controls, the user has to manually modify the location settings before launching each app, which is quite cumbersome and annoying. Second, even if the user can easily change these settings, making an informed decision is a different story. Therefore, we propose `LP-Doctor` that helps users utilize the existing OS controls to provide location-access control on a per-session basis.

3.9.1 Design

`LP-Doctor` trusts the underlying OS and its associated apps; it targets user-level apps accessing location while running in the foreground, as we found that most apps do not access location in the background. `LP-Doctor` focuses on the apps with fine location permissions as they could pose higher threats. `LP-Doctor` automatically coarsens location for apps requesting coarse location permissions to ensure a commensurate privacy-protection level.

The main operation of `LP-Doctor` consists of two parts. The first involves the user-transparent operations described below, while the second includes the interactions with the user described in Section 3.9.2.

We bundled `LP-Doctor` with CyanogenMod's app launcher.[2] It runs as a background

[2]Source code: `https://github.com/kmfawaz/LP-Doctor`.

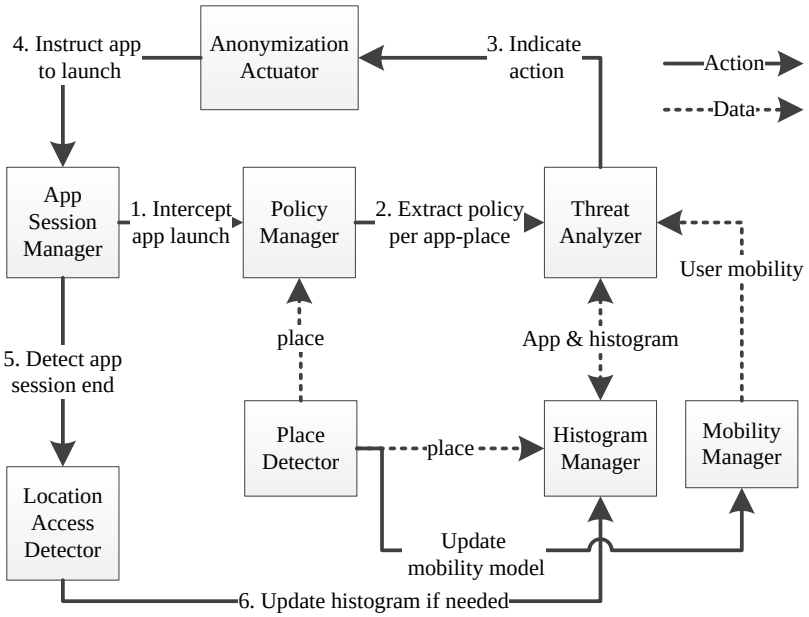


Figure 3.12: The execution flow of LP−Doctor when a location-aware app launches.

service, intercepts app-launch events, decides on the appropriate actions, performs these actions, and then instructs the app to launch. Fig. 3.12 shows the high-level execution flow of LP−Doctor. Next, we elaborate on LP−Doctor's components and their interactions.

App Session Manager: is responsible for monitoring app launch and exit events. LP−Doctor needs to intercept app-launch events to anonymize location.

Fortunately, Android (recently iOS as well) allow for developing custom app launchers. Users can download and install these launchers from the app store which will, in turn, be responsible for listening to the user's events and executing the apps. We instrumented CyanogenMod's app launcher (available as open source and under Apache 2 license) to intercept app launch events.

Particularly, before the app launcher instructs the app to execute, we stop the execution, save the state, and send an intent to LP−Doctor's background service (step 1 in Fig. 3.12). LP−Doctor takes a set of actions and sends an intent to the app launcher, signaling the app can launch (steps 2 and 3 in Fig. 3.12). The app launcher then restores the saved state and proceeds with execution of the app (step 4 in Fig. 3.12). In Section 3.9.4, we will report

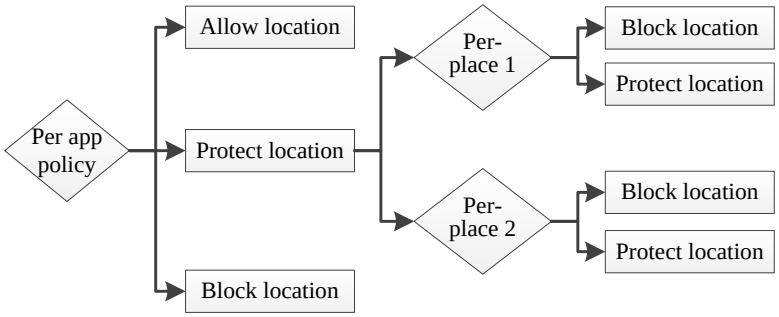


Figure 3.13: The policy hierarchy of `LP-Doctor`.

the additional delay incurred by this operation.

In the background, `LP-Doctor` frequently polls (once every 10s) the current foreground app to detect if the app is still running. For this purpose, it uses `getRecentTasks` on older versions of Android and `AppUsageStats` class for Android L. When an app is no longer running in the foreground, `LP-Doctor` executes a set of maintenance operations to be described later (steps 5 and 6 in Fig. 3.12).

Policy Manager: fetches the privacy policy for the currently visited place and the launched app as shown in Fig. 3.13.

At installation time, the user specifies a privacy policy to be applied for the app. We call this the *per-app policy* which specifies three possible actions: *block*, *allow*, and *protect*. If the per-app policy indicates privacy protection, `LP-Doctor` asks the user to specify a per-place policy for the app. The per-place policy indicates the policy that `LP-Doctor` must follow when the app launches from a particular place. The policy manager passes the app's policy and the current place to the threat analyzer.

Place Detector & Mobility Manager: The place detector monitors the user's actual location, and applies online clustering to extract the spatio-temporal clusters which represent places that the users visit. Whenever the user changes the place s/he is visiting, the place detector module instructs the mobility manager to update the mobility profile of the user as defined in Section 3.6.

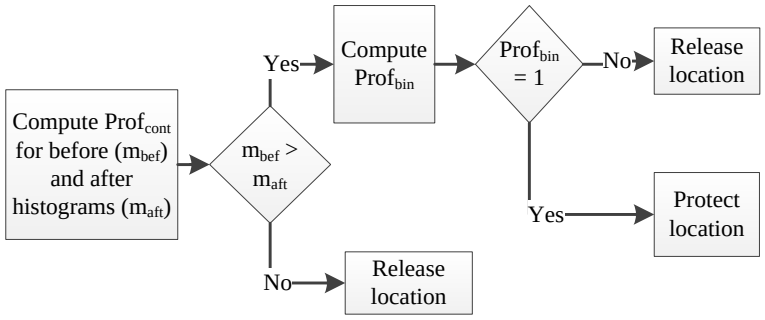


Figure 3.14: The threat analyzer's decision diagram.

Histogram Manager: maintains the histogram of the places visited as observed by each app. It stores the histograms in an SQLite table that contains the mapping of each app–place combination to a number of observations. The threat analyzer module consults the histogram manager to obtain two histograms whenever an app is about to launch. The first is the current histogram of the app (based on previous app events) which we refer to as the "before" histogram. While the second one is the potential histogram if the app were to access location from the currently visiting place; we call this histogram as the "after" one.

Threat Analyzer: decides on the course of action regarding apps associated with a *protect* policy. It basically performs the decision diagram depicted in Fig. 3.14 to decide whether to release the location or add noise.

The threat analyzer determines whether the "after" histogram leaks more information than the old one through computing $Prof_{cont}$ for each histogram. If $Prof_{cont}$ increases `LP-Doctor` decides to release the location to the app. On the other hand, if $Prof_{cont}$ decreases, `LP-Doctor` uses $Prof_{bin}$ to decide if the "after" histogram fits the user's mobility pattern and whether to release or anonymize location.

$Prof_{bin}$ depends on the significance level, α, as we specified in Section 3.6. In `LP-Doctor`, α is a function of the privacy level chosen by the user. `LP-Doctor` recognizes three privacy levels: low, medium, and high. Low privacy corresponds to $\alpha = 0.1$; medium privacy corresponds to $\alpha = 0.05$; and high privacy protection corresponds to the most conservative

$\alpha = 0.01$.

The procedure depicted in Fig. 3.14 will not hide places that the user seldom visits but are sensitive to him/her. The per-place policies allow the user to set a privacy policy for each visited place, effectively allowing him/her to control the places s/he wants revealed to the service providers. Also, `LP-Doctor` can be extended to support other privacy criteria that try to achieve optimal privacy by perturbing location data [99, 100].

Anonymization Actuator: receives an action to perform from the threat analyzer. If the action is to protect the current location, the actuator computes a fake location by adding Laplacian noise [66] to ensure location indistinguishability. The privacy level determines the amount of noise to be added on top of the current location. One the other hand, if the action is to block, the actuator computes the fake location of $< 0, 0 >$.

As specified by Andrés *et al.* [66], repetitive engagement of Laplacian noise mechanism at the same location leaks information about the location. To counter this threat, `LP-Doctor` computes the anonymized location once per location and protection-level combination, and saves it. When the user visits the same location again, `LP-Doctor` employs the same anonymized location that was previously computed to prevent `LP-Doctor` from recomputing a fake location for the same place.

After computing/fetching the fake location, the actuator module will engage the mock location provider. The mock location provider is an Android developer feature to modify the location provided to the app from Android. It requires no change in the OS or the app. The actuator then displays a non-intrusive notification to the user, and signals the session manager to start the app.

End-of-Session Maintenance: When the app finishes execution, the actuator disengages the mock location provider, if engaged. The location-access detector will then detect if the app accessed location to update the app's histogram accordingly. The location access detector performs a "dumpsys location" to exactly detect if the app accessed location or not while running. If it did access location, the location-access detector module updates

the app's histogram (increment the number of visits from the current location). It is worth noting that LP−Doctor treats sessions of the same app within 1 min as the same app session.

3.9.2 User Interactions

LP−Doctor interacts with the user to communicate privacy-protection status. It also enables him/her to populate the privacy profiles for different apps and places. As will be evident below, the main philosophy guiding LP−Doctor's design is to minimize the user interactions, especially intrusive ones. We satisfy two design principles proposed by Felt *et al.* [101] that should guide the design of a permission granting UI. The first principle is to conserve user attention by not issuing excessively repetitive prompts. The second is to avoid interrupting the user's primary tasks.

Bootstrapping Menu: The first communication instance with LP−Doctor takes place upon its installation. LP−Doctor will ask the user to set general configuration options. These options include (1) alerting the user when visiting a new location to set the per-place policies and (2) invoking protection for A&A libraries. The menu will also instruct the user to enable the mock location provider and grant the app "DUMP" permissions through ADB. This interaction takes place only once per LP−Doctor's lifetime.

Installation Menu: LP−Doctor prompts the user when a new (non-system and location-aware) app is installed. The menu enables the user to set the per-app profiles. Fig. 3.15 shows the displayed menu when an app ("uber" in this case) has finished installation. The user can choose one of three options which populates three app sets: app_{allow}, app_{block}, and $app_{protect}$.

Logically, this menu resembles the per-app location settings for iOS, except that it provides users with an additional option of privacy protection. The protection option acts as a middle-ground between completely allowing and blocking location access to the app. The user will interact with this menu; only once per app, and only for non-system apps that

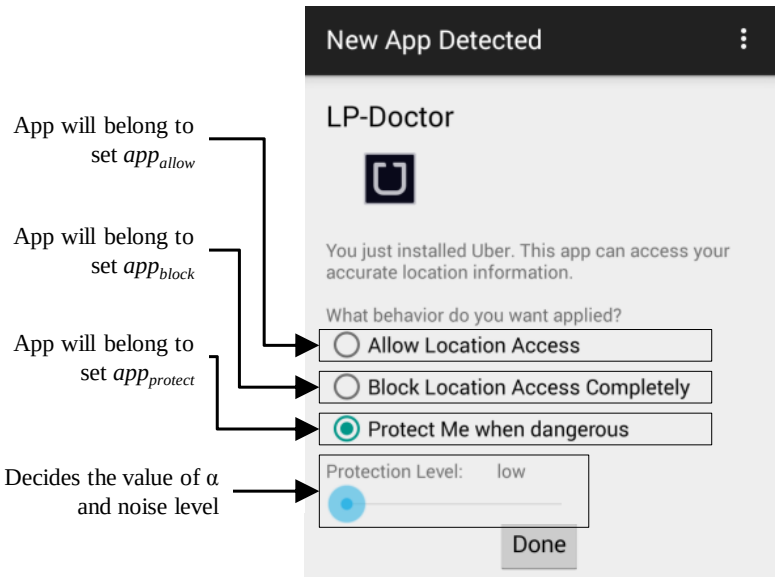


Figure 3.15: The installation menu.

requests the fine location permission. Based on our PhoneLab dataset, we estimate that the user will be issued this menu on average for one app s/he installs per five installed apps on the device.

Per-Place Prompts: `LP-Doctor` relies on the user to decide its actions in different visited places, if s/he agrees to get prompted when visiting new places. Specifically, whenever the user visits a new place, `LP-Doctor` prompts him/her to decide on actions to perform when running apps that the user chose to protect. We call these *per-place policies* (Fig. 3.13).

The per-place policies apply for apps belonging to the set $app_{protect}$. The user has the option to specify whether to block location access completely, or apply protection. Applying protection will proceed to execute the operations of the threat analyzer as defined in Fig. 3.14. `LP-Doctor` allows the user to modify the policies for each app–place combination.

`LP-Doctor` issues this prompt only when the user launched an app of the set $app_{protect}$ from a new location. From our PhoneLab dataset, we estimate that such a prompt will be issued to the user at most once a week.

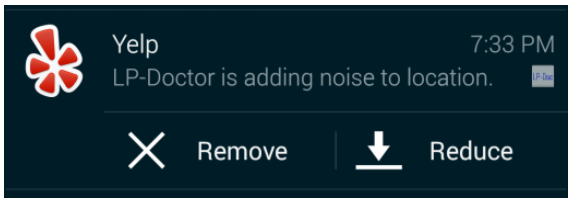


Figure 3.16: LP−Doctor's notification when adding noise.

Notifications: As specified earlier, the threat actuator displays a non-intrusive notification (Fig. 3.16) to the user to inform him/her about the action being taken.

If the action is to allow location access (because the policy dictates so or there is no threat), LP−Doctor notifies the user that there is no action being taken. The user has the option to invoke privacy protection for the current app session. If the user instructs LP−Doctor to add noise for a single app over two consecutive sessions from the same place, LP−Doctor will create a per-place policy for the app and move it to the $app_{protect}$ set if it were part of app_{allow}.

On the other hand, if LP−Doctor decides to add noise to location or block it, it will notify the user of it (Fig. 3.16). The notification includes two actions that the user can make: remove or reduce noise. If the user overrides LP−Doctor's actions for two consecutive sessions of an app from the same place, LP−Doctor remembers the decision for future reuse.

LP−Doctor leverages the user's behavior to learn the protection level that achieves a favorable privacy–utility tradeoff. Since the mapping between the chosen privacy and noise levels is independent of the running app, the functionality of certain apps might be affected. LP−Doctor allows the user to fine-tune this noise level and then remembers his/her preference for future reuse.

Reducing the noise level will involve recomputing the fake location with a lower noise value (if no such location has been computed before). One could show that leak of information (from lowering noise level successively) will be capped by that corresponding to

the fake location with the lowest noise level released to the service provider.

Using our own and PhoneLab's datasets, we estimate LP-Doctor's need to issue such non-intrusive notification (indicating protection taking place) for only 12% of the sessions on average for each app.

3.9.3 Limitations

The user-level nature of LP-Doctor introduces some limitations related to certain classes of apps. First, LP-Doctor, like other mechanisms, is inapplicable to apps that require accurate location access such as navigation apps for elongated period of times.

Second, LP-Doctor cannot protect the user against apps utilizing unofficial location sources such as "WeChat." Such apps might scan for nearby WiFi access points and then use scan results to compute location. LP-Doctor cannot anonymize location fed to such apps, though it can warn the user of the privacy threat incurred if the user is to invoke the location-based functionality. Also, it can offer the user the option to turn off the WiFi on the device to prevent accurate localization by the app when running.

Finally, LP-Doctor doesn't apply privacy protection to the apps continuously accessing location while running in the background. Constantly invoking the mock location provider affects the usability of apps that require fresh and accurate location when running. Fortunately, we found that the majority of the apps do not access location in the background (Section 3.4). Nevertheless, this still highlights the need for OS support to control apps' location access in the background (like the one that iOS currently provides).

It is worth noting that LP-Guardian does not suffer from the last two limitations, because of its implementation. This design choice highlights the trade-off between deployability and privacy protection. LP-Guardian goes a bit deeper into the smartphone to provide more comprehensive privacy protection at the cost of ease of installation. On the other hand, LP-Doctor is easy to install but cannot cover all the location-access scenarios.

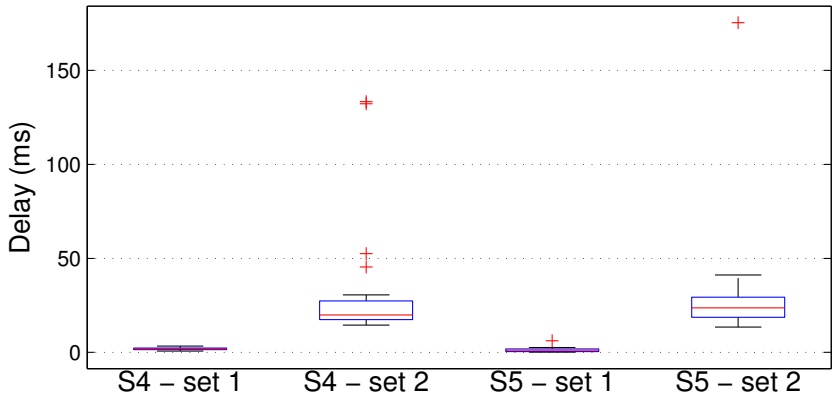


Figure 3.17: The app launch delay caused by `LP-Doctor`.

3.9.4 Evaluation

We now evaluate and report `LP-Doctor`'s overhead on performance, Quality of Service (QoS), and usability.

3.9.4.1 Performance

`LP-Doctor` performs a set of operations which delay the app launching. We evaluate this delay on two devices: Samsung Galaxy S4 running Android 4.2.2, and Samsung Galaxy S5 running Android 4.4.4. We recorded the delay in launching a set of apps while running `LP-Doctor`. We partitioned those apps into two sets. The first (set 1) includes the apps which `LP-Doctor` doesn't target, while the second (set 2) includes non-system apps that request fine location permissions.

Fig. 3.17 plots the delay distribution for both devices and for the two app sets. Clearly, apps that belong to the first set experience very minimal delay, varying between 1 and 3ms. The second set of apps experience longer delays without exceeding 50ms for both devices. We also tested `LP-Doctor`'s impact on the battery by recording the battery depletion time when `LP-Doctor` was running in the background and when it was not. We found that `LP-Doctor` has less than 10% energy overhead (measured as the difference in battery depletion time). Besides, `LP-Doctor` runs the same logic as our PhoneLab survey app in the background which 95 users ran over 4 months and reported no performance or battery

issues.

3.9.4.2 User Study

To evaluate the usability of `LP-Doctor` and its effect on QoS, we conducted a user study over Amazon Mechanical Turk. We designed two Human Intelligence Tasks (HITs), each evaluating a different representative testing scenario of `LP-Doctor`.

Apps that provide location-based services (LBSes) fall into several categories. On one dimension, an app can *pull* information to the user based on the current location, or it can *push* the user's current location to other users. On another dimension, the app can access the user's location *continuously* or *sporadically* to provide the LBS. One can then categorize apps as: *pull-sporadic* (e.g., weather, Yelp, etc.), *pull-continuous* (e.g., Google Now), *push-sporadic* (e.g., geo-tagging, Facebook check-in, etc.), or *push-continuous* (e.g., Google Latitude). As `LP-Doctor` isn't effective against apps continuously accessing the user's location (which are a minority to start with), we focus on studying the user's experience of `LP-Doctor` while using **Yelp**, as a representative example of *pull-sporadic* apps, and **Facebook**, as representative example of *push-sporadic* apps.

We recruited 120 participants for the Yelp HIT and another 122 for the Facebook HIT[3]; we had 227 unique participants in total. On average, each participant completed the HIT in 20min and was compensated $3 for his/her response. We didn't ask the users for any personal information and nor did `LP-Doctor`. We limited the study to Android users.

Of the participants: 28% were females vs. 72% males; 32% had high school education, 47% with BS degree or equivalent; and 37% are older than 30 years. Also, 52% of the participants reported that they have taken steps to mitigate privacy threats. Interestingly, 93% of the participants didn't have mock locations enabled on their devices indicating the participants are not tech-savvy.

We constructed the study with a set of connected tasks. In every task, the online form

[3] `https://kabru.eecs.umich.edu/wordpress/wp-content/uploads/lp-doctor-survey-fb.pdf`

displays a set of instructions/questions that the participant user must follow/answer. After successfully completing the task, LP−Doctor displays a special code that the participant must input to proceed to the next task. In what follows, we describe the various tasks that we asked users to perform and how they responded.

Installing and configuring LP−Doctor: The participants' first task was to download LP−Doctor from Google Play and enable mock locations. We asked the users to rate how difficult it was to enable mock locations on the scale of 1 (easy) to 5 (difficult). 83% of the participants answered with a value of 1 or 2 implying that LP−Doctor is easy to install.

Installation menu: In their second task, the participants interacted with the installation menu (Fig. 3.15). The users had to install (re-install if already installed) either Yelp or Facebook. Just when either app completes installation, LP−Doctor presents the user with the menu to input the privacy options. The participants reported a positive experience with this menu; 83% reported it was easy to use (rated 1 or 2 on a scale of 1 (easy) to 5 (hard)); 86% said it was informative; 83% thought it provides them with more control than Android's permission; 79% answered it is useful (rated 1 or 2 on a scale of 1 (useful) to 5 (useless)); and 74% would like to have such menu appearing whenever they install a location-aware app (12% answered with not sure).

Impact on QoS: The survey version of LP−Doctor adds noise on top of the user's location regardless of his/her previous choice. This allowed us to test the impact of adding noise (Laplacian with 1000m radius) to the location accessed by either Yelp or Facebook. We didn't ask the participants to assess the effect of location anonymization on the QoS directly. Rather, we asked the Yelp respondents to report their satisfaction with the list of restaurants returned by the app. While we asked the Facebook respondents to indicate whether the list of places to check-in from is relevant to them. The participants in the first HIT indicated that Yelp ran normally (82%), the restaurant search results were relevant (73%), the user experience didn't change (76%), and Yelp need not access the user's accurate location (67%).

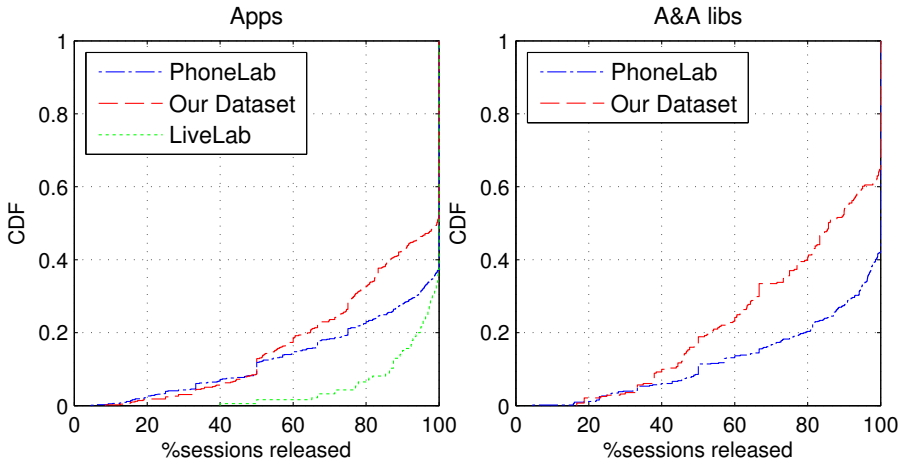


Figure 3.18: The distribution of percentage of sessions where apps maintain QoS for apps (left) and A&A libraries (right).

The Facebook HIT participants exhibited similar results: Facebook ran normally (80%), the list of places to check-in was relevant (60%), user experience didn't change (80%), and Facebook need not access the user's accurate location (80%).

Fig. 3.18 shows the percentage of sessions (for all app–user combinations) that will not experience any noise addition according to our datasets. It is obvious that the percentage of sessions with potential loss in QoS (when `LP-Doctor` adds noise) is minimal (less than 20%, a bit higher if the user opts for A&A libraries protection). Our user study shows that more than 70% of the users will not experience loss in QoS in these sessions. For those users who do face loss in QoS, `LP-Doctor` provides them with the option of adjusting the noise level at runtime through the notifications.

Notifications: In the final task, we asked the participants to test the noise reduction feature that allows for a personalized privacy–utility trade-off. After they reduced the noise level, they would invoke the location-based feature in both Yelp and Facebook and check if the results were improved. Indeed, most of the participants who reported loss in QoS reported the Yelp's search results (64%) and Facebook's check-in places (70%) improved after reducing the noise.

The participants also indicated the the noise reduction feature is easy to use (75%). 86%

of the participants will not mind having this feature whenever they launch a location-aware app.

Post-study questions: As we couldn't control the per-place prompts given our study design, we asked the participants for their opinion about being prompted when visiting new places (per-place prompts). Only 54% answered they would prefer prompted, 37% answered negatively, and the rest answered "I am not sure." These responses are consistent with our design decision; the user has to approve per-place prompts when initially configuring `LP-Doctor` as they are not automatically enabled.

Also, 82% of the participants felt comfortable that Facebook (80%) and Yelp (85%) didn't access their accurate location. Finally, 77% of the participants answered "Yes" when asked about installing `LP-Doctor` or other tool to protect their location privacy. Only 11% answered "No" and the rest answered with "I am not sure." This result comes at an improvement over the 52% who initially said they took steps in the past to address location-privacy threat.

In summary, we conducted one of the few studies (e.g., [102]) that evaluate a location-privacy protection mechanism in the wild. We showed that location-privacy protection is feasible in practice where a balance between QoS, usability, and privacy could be achieved.

3.10 Conclusion

In this chapter, we posed a question about the effectiveness of OS-based location-access controls and whether they can be improved. To answer this question, we conducted a location-collection campaign that considers location-privacy threats from the perspective of mobile apps. From this campaign, we observed, modeled, and categorized profiling as being the prominent privacy threat from location access for both apps and A&A libraries. We concluded that controlling location access per session is needed to balance between loss in QoS and privacy protection. As existing OS controls do not readily provide such

functionality, we proposed `LP-Doctor`, a user-level tool that helps the user better utilize existing OS-based location-access controls. `LP-Doctor` is shown to be able to mitigate privacy threats from both apps and A&A libraries with little effect on usability and QoS. In future, we would like to test `LP-Doctor` in the wild and use it to explore the dynamics that affect users' decisions to install a location-privacy protection mechanism.

CHAPTER IV

PR-LBS

4.1 Introduction

Localization technologies [10], tailored for indoor spaces such as retail stores, malls, airports, museums, and hospitals, are gaining popularity. An *indoor service provider* (SP) (e.g., retail store owner) utilizes the customers' indoor location information to study their behaviors and infer their preferences and interests. In the best case, this should be a win-win for both customers and SPs; the SPs collect location data and deliver better service to the customers which leads to enhanced customer satisfaction and eventually increased revenues.

Unfortunately, indoor localization has not been realized to its full potential. Customer resistance is forcing SPs to either sideline the technology (e.g., Nordstrom ceased customer tracking after public outrage [103]) or rely solely on anonymous data collection [104]. Analyzing customers' location data anonymously prevents the service provider from offering them personalized services that would result in revenue-generation/increase .

To gain a better understanding of the users' perspectives towards indoor localization, we surveyed 200 shoppers in two major retailers: Walmart and Nordstrom. The survey shows that customers have both privacy and utility concerns.

A. Privacy Concerns: Users cited privacy concerns for not accepting this technology (consistent with other surveys [105]). An SP, tracking users' mobility, has the potential

to infer personality traits and/or habits that could be private to them. For example, a retailer can infer from the frequently-visited aisles the shopper's gender (men's vs. women's clothing), ethnicity (ethnic food aisles), socioeconomic status (expensive vs. inexpensive clothing and accessories), health condition (pharmacy aisles), sensitive interests (sporting goods, adult magazines and films), religious beliefs (clothing, specific food aisles), etc.

Unlike the outdoor case, the indoor SP is directly involved in the user's localization through the deployed infrastructure such as Wi-Fi and Bluetooth. Unless the users turn off their devices, the SP does not provide an opt-out mechanism by which users can exert control over how much of their mobility is being tracked. According to our survey, users are not comfortable with the SP storing their mobility information even when it is processed anonymously.

B. Utility Concerns: Users expressed interest in receiving rewards for sharing some of their mobility information. This is referred to in the literature as a *fair transaction* [106]; a user shares some data proportionately with the received rewards. In the indoor case, the users might find it challenging to engage in a fair transaction with the SP. First, it is hard for them to associate their mobility with a privacy cost in the typically public indoor space (in the outdoor case there is some notion of a private location such as home). Second, although the location information might help the user indirectly, through improving store layout, product placement, or waiting time in checkout lines, but these are not personalized and tangible services that would make users feel satisfied for revealing their mobility.

In this chapter, we first pose a question: *can the seemingly conflicting requirements of the users and SPs be effectively resolved?* To answer this question, we propose `PR-LBS` (Privacy vs. Reward in Location Based Service); a novel framework that addresses the user's privacy concerns and enables them to receive the right reward from their location sharing on one side. On the other side, it provides the indoor SPs with enough information to perform aggregate and more personalized analysis of the customers.

`PR-LBS` puts the users in control, allowing them to specify a privacy setting that trans-

lates into a provable privacy guarantee. PR−LBS packs in an online private location release mechanism that achieves differential privacy guarantees in indoor environments. Additionally, PR−LBS enables the users to set high-level policies that provide their utility definition as a function of the privacy "cost" of sharing location and "benefit" received from the SP. To estimate the cost of sharing the user's indoor mobility, we introduce a new privacy criterion, which is based on information disclosure.

PR−LBS improves on the current approaches of take-it or leave-it; it ensures a fair transaction of the user's location information with the SP by abstracting the interactions between the user and the SP as a repeated play model [107]. PR−LBS employs the strategic experts algorithm [108] to choose, at run-time, the action (hide, reveal, or anonymize location) sequence that maximizes the user's utility.

PR−LBS is a generic framework that supports various practical deployment scenarios. A user can simply download and install it to the device, which we call *device mode*, if localization is device-based, such as iBeacons [109]. Also, a localization provider can employ PR−LBS as a broker between the user and the SP, which we call *infrastructure mode*, in case localization is infrastructure-based such as CUPID [10]. PR−LBS could act as a privacy guarantee/seal in this case [110], which will make users more comfortable to share their location. In this chapter, we design and evaluate PR−LBS in both modes and present a full real-life implementation on Android for the device mode.

PR−LBS has a low energy footprint when running on the user's device and is easy to use as our user study (100 respondents) shows. Our survey also indicates that users are more comfortable with location tracking technology with PR−LBS being deployed. Further, our evaluations of PR−LBS in 8 different scenarios show that PR−LBS strikes a balance between the user and the SP. It controls the release of location information to protect the users' privacy, rewards them with commensurate service, and maintains data utility for the SP.

The chapter is organized as follows. Section 4.3 reviews the related work. Section 4.4

presents our survey. Sections 4.5 and 4.6 present our system and privacy models, respectively. Section 4.7 details the design of PR−LBS. Section 4.8 describes our implementation and evaluation of PR−LBS. Section 4.9 lists some limitations of PR−LBS. Section 4.10 comments on the applicability of PR−LBS in outdoor scenarios. Finally, Section 4.11 concludes the chapter.

4.2 Specifics of the Indoor Environments

Although one could find the privacy threats similar between the indoor and outdoor cases, the privacy protection mechanisms of Chapters II and III do not apply to the indoor case. The indoor environment presents with a set of unique challenges that make models and treatments available for other outdoor scenarios inapplicable.

Monitoring location in an outdoor environment has the potential of profiling the users, as we indicated earlier. Nevertheless, an indoor location service provider does not necessarily have access to the user's outdoor mobility. It is, from the user's perspective, an additional entity that has the potential of posing profiling threats. Existing models and techniques for the location privacy protection in the outdoor case, such as LP−Guardian and LP−Doctor can do little in addressing the indoor location privacy threats for the following reasons:

- Hiding the user's private locations (home, work, etc.) prevents an outdoor location service provider from inferring sensitive information (e.g., identity). The indoor space, on the other hand, is typically a public space where there is no notion of a private location.

- The outdoor mobility tends to be more relaxed in time; transitions between different areas take place in the order of minutes. There is ample time to incorporate user's feedback to inform the privacy valuation. In contrary, the user's mobility is more

time-constrained in the indoor case, and users cannot reasonably provide timely feedback to a privacy protection service.

- Adding to the difference between both environments is the service vs. location access ecosystem. In the outdoor case, the service providers do not engage in continuous tracking. The user sporadically reveals his/her location and receives some service in return. It is not the case in the indoor case where user's location can be continuously monitored with no tangible return.

- In the outdoor case, the user's device typically estimates the current position independently of the service provider. On the other hand, the service provider is involved in computing the user's location mainly through the deployed infrastructure (e.g., WiFi, iBeacons).

4.3 Related Work

There have been numerous efforts to mitigate the privacy risks in indoor environments. Retailers provide customers with opt-out options and claim to analyze their data in aggregate [104]. Our survey showed that users are likely to opt-out if provided with the option. Also, aggregate processing does little to protect the users' privacy. The SP still stores mobility data that is tagged with a MAC address (or a hashed form thereof). The hashed MAC can link the user to his/her traces [111] and can be reverse-mapped to the original MAC [112]. Alternatively, `PR-LBS` provides provable privacy guarantees by limiting the additional knowledge the SP attains from observing the user's mobility.

Other approaches rely on complete prevention of localization [57, 55, 56, 54]. `PR-LBS` capitalizes on these mechanisms by acting as a control knob to opportunistically decide when to activate/deactivate them. `PR-LBS` exercises fine-grained location control to protect users' privacy while allowing them to interact with the SP. Recently, there have been mobile apps (such as that by Placed and Shopkick) that allow the users to receive rewards

for location check-ins. PR−LBS automates this process; it acts on the user's behalf to decide when it is beneficial to share location or not. The user only specifies a privacy setting and high-level policy while PR−LBS takes care of deciding the privacy cost, service level, and sharing/hiding/anonymizing the location.

Researchers have proposed mechanisms to appraise private data before sharing it so that the user is properly rewarded (e.g., the architectures of Riederer *et al.* [113] and Ghosh and Roth [114]). These mechanisms typically require cooperation from both users and SPs. PR−LBS does not change the communication between the user and the SP and targets the case of a non-cooperative SP. Finally, Shokri [115] proposes a theoretical framework that optimizes user-side obfuscation to maintain both differential and distortion privacy with the impact on utility not being greater than the case of optimizing for a single privacy criterion. PR−LBS takes a different approach; it provides concrete mechanisms that achieve differential privacy. Moreover, it maximizes the user's rewards by adapting actions to the SP's services. PR−LBS is also a practical system that users can run in real-world environments.

4.4 Survey

We designed two surveys to study individuals' behavior and privacy preferences when shopping in Walmart (104 respondents) and Nordstrom (100 respondents) using Amazon Mechanical Turk. We compensated each participant with $3 and the average time for survey completion was 24 minutes. To protect the privacy of the participants, we did not collect any personal information and processed the data in aggregate. We also introduced a set of questions to weed out inconsistent responses.

The participants are diverse; they are uniformly distributed among genders, 42% are between 15 and 29 years old, while 43% fall between 30 and 44 years of age, half with a university/college degree, 29% with a high school degree, and 75% visit a brick-and-mortar shop at least once a week (97% at least once a month).

In the first section of the survey, we presented the respondents with the disclaimer

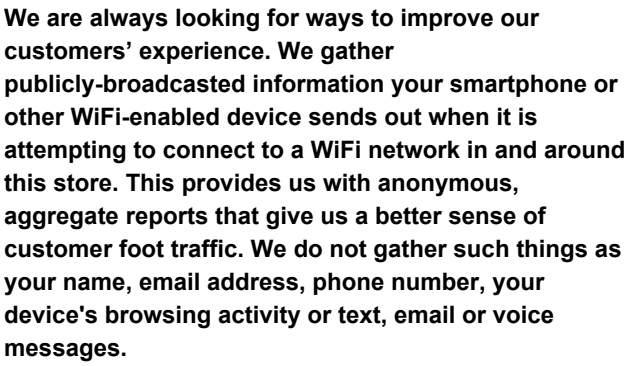


Figure 4.1: The disclaimer presented at the start of the survey.

(Fig. 4.1) that Nordstrom displayed to the shoppers in 2013. We then asked them, after reading the disclaimer, whether they would hypothetically consent to either Nordstrom or Walmart gathering Wi-Fi information assisting in their localization. This was the first mentioning of indoor localization-related terminology in the survey; we used the same language of a retailer to avoid any bias.

Interestingly, the participants responded with a preference to prevent the store from gathering information assisting in their localization (70%), 18% indicated that they would consent to the store gathering parts of their Wi-Fi information. Only 10% of the participants consented to full gathering of Wi-Fi information broadcasted by their devices. We then posed the same question differently by indicating that the disclaimer effectively asks for the user's consent for indoor location tracking. The response distribution shifted a bit; 61% chose to prevent location tracking entirely, 24% chose to allow the store to gather part of their mobility, while the rest (15%) consented to full location tracking.

In the rest of this chapter, we refer to the first set of individuals (rejecting tracking) as *privacy-oriented*, the second set (consenting to only part of tracking) as *neutral*, while we refer to the third set of individuals as *service-oriented*. These categories are akin to Westin's [116] categorization of privacy orientations of individuals as fundamentalists (privacy-oriented), pragmatists (neutral), and unconcerned (service-oriented).

Privacy-oriented Participants: These participants cited a set of privacy-related reasons

as to why they reject location tracking. The most recurring reason was that they do not trust the store with mobility data (49%), the second being they do not feel comfortable with their mobility information being gathered (43%), and the third was that the store provides nothing in return for gathered mobility data (41%).

We also asked these respondents about their perception of the difference between smartphone-based and other tracking technologies such as monitoring purchase history (through credit card or rewards program) and using CCTV cameras. Regarding purchase history, only 10% indicated that purchase history reveals the same information as their mobility. The rest of the participants indicated that they do not want the store to know what items they are interested in but did not end up buying or that they use cash for their purchases. We observed a similar trend with CCTV cameras; only 20% of the participants indicated that cameras and location tracking reveal the same information while the others felt that it is harder to track them using CCTV cameras.

Neutral Participants: The second set of participants cited similar reasons as to why they want some part of their mobility to be hidden. As for which parts of their mobility they want to be hidden, they responded with those areas that they deem private (65%), areas of the store that include items they browse but do not buy (44%), and areas where they receive nothing in return from the service provider (25%).

Service-oriented Participants: Individuals belonging to this set of survey respondents do not feel threatened by the store owner tracking their mobility. When asked whether they would change this perspective if the store owner would treat them differently based on mobility data, 30% indicated that would choose to prevent tracking, 37% still consented to full tracking, and the rest (33%) answered by not being sure. The participants' perspective further shifted when we indicated that the store owner might share their mobility with a third-party entity (an advertisement agency for example). 50% indicated that they do not consent to location tracking anymore and only 26% responded that they have no problem with their location being tracked.

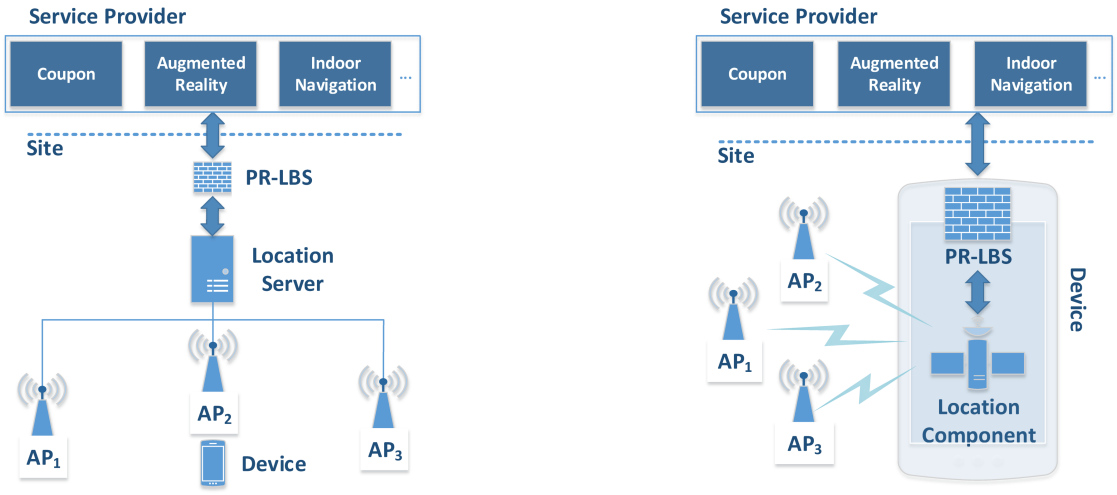
In the second part of the survey, we asked participants to trace their path, on a map of the store, the last time they went shopping at either Walmart or Nordstrom if they remember it well. We then asked them to indicate the parts of the path they would hide from either store. Interestingly, even privacy-oriented respondents did not choose to hide all zones of their paths. In the last part of the survey, we asked participants to input their satisfaction level in different situations. More than 40% of the privacy-oriented users indicated that they would be satisfied if they were to share some of their mobility and receive very good service in return.

4.5 System Model

`PR-LBS` addresses the case of a user's location tracking **exclusively** in constrained public (including indoor) spaces, such as retail stores, malls, museums, theme parks, etc., where a localization system is installed. We consider the following main entities involved in the ecosystem:

- **User**: the individual moving around in the space of interest while carrying a mobile device.
- **Service Provider (SP)**: the entity owning the space in which the user moves. It manages a set of application servers that analyze the user's location and push service in the form of coupons, directions, deals, promos, etc.
- **Localization Provider (LP)**: an entity contracted by the SP to localize the users. It relies on the deployed Wi-Fi access points or Bluetooth beacons to track users via the devices they carry. The LP can reside either on the device side or on the infrastructure side.

The SP deploys a mobile app (e.g., Shopkick) that acts as its communication channel with the user. The SP uses a consistent identifier such as the MAC address to map the



(a) Infrastructure mode

(b) Device mode

Figure 4.2: PR-LBS deployment options

location updates to the user running the app. The SP then pushes the tailored location-based content to the user through the app.

Logically, PR-LBS runs between the LP and the SP. It is a trusted module that controls the release of the location information to the SP in a privacy-aware manner. It is very important to note that the SP only views that mobility of the user that has been released by PR-LBS. PR-LBS runs in *device* or *infrastructure* mode:

Infrastructure mode (Fig. 4.2a): fits infrastructure-based localization where the LP has to install and run PR-LBS as the device can not control location sharing. This, however, could only happen if the SP has enough incentives to do so. Given users' privacy concerns, the SP has an incentive to deploy a solution that mitigates these concerns. For example, European companies have to apply for a privacy certification before collecting users' data (including indoor location) [110].

Device mode (Fig. 4.2b): fits device-based localization that computes the location on the device and then shares it with the SP. The user installs and runs PR-LBS that controls location release from the device.

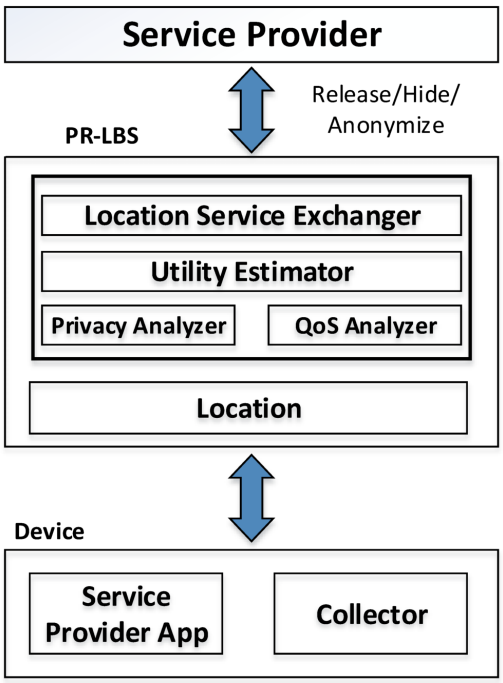


Figure 4.3: The high-level operations of PR−LBS.

4.5.1 High-Level Description

Fig. 4.3 shows the high-level operations that PR−LBS performs when deployed in infrastructure mode. In device-mode deployment, PR−LBS performs the same operations while running on the device.

In the view of PR−LBS, the area of interest is partitioned into a set of zones: $Z = \{z_k\}$. The zones are semantic sub-areas within the area of interest which the SPs are typically interested in mapping the user's location to. For example, a zone could refer to an aisle in a supermarket, a department in a store, or an entertainment station in a theme park.

As the user moves from a zone z_i to another zone z_j, PR−LBS decides whether to *release*, *hide*, or *anonymize* z_i from the SP (Section 4.7.1). This action will result in a potential privacy cost to the user, $leak(z_i)$, as estimated by the *privacy analyzer* as described in Section 4.6.3. While spending time at z_i and then moving to z_j, the SP will be pushing a service to the user's device. When the user visits z_j, the *QoS analyzer* (Section 4.7.3)

estimates the value of the service, $serv_i$, the user received as a result of hiding, releasing, or anonymizing z_i to the SP.

A *transaction* between the user and the SP takes place during the time period spanning the user reaching z_i and just before arriving at z_j. The *utility estimator* (Section 4.7.2) module of PR−LBS computes the utility (user-defined) the user gained after incurring a cost=$leak(z_i)$ and receiving a benefit= $serv_i$. The exchange module employs a set of privacy preserving mechanisms (Section 4.6.2) as "experts" that dictate the action to perform. This module utilizes the history of the user–SP transactions to decide the best expert (maximizing users' utility) to follow when reaching z_j.

Finally, PR−LBS has a *collector* module that runs on the device and collects the privacy preferences of the user. While PR−LBS is running, the collector module gathers information to assist the QoS analyzer in computing the service that the user receives.

4.6 Privacy Model

From the users' perspective, any entity collecting their location has the potential of posing privacy threats. We make a natural choice to trust the user's device as no solution is feasible without such a trust. We also choose to trust the LP only if it deploys PR−LBS, as it will indicate a willingness to provide privacy protection to the user. Thus, we assume that the privacy threats originate from the SP's analysis of the collected location data and the resulting treatment of the user.

The SP is an honest-but-curious entity that passively profiles the user through location information. These SPs will not collude with the LP (infrastructure-based case) if they choose to deploy PR−LBS. While the privacy threats are evident for an infrastructure-based LP, device-based localization might be perceived as less threatening. Proximity beacons can not track the user's mobility, but smartphone apps scanning for beacons pose tracking threats. We found that several shopping apps, including Shopkick, scan for nearby beacons and upload them along with consistent identifiers allowing them to track the user's mobility.

In this chapter, we only consider the threats to the user's privacy from location tracking that originate from the user's smartphone. An SP might utilize other channels to localize a user, e.g., CCTV cameras, which the user and PR−LBS, unfortunately, can not control. We also view security challenges as orthogonal to this work.

Intuitively, any privacy loss that the user suffers from the SP accessing his/her location takes place through processing and analyzing this collected mobility information. Privacy loss is then a function of the information disclosed from observing the user's mobility. In what follows, we define the mobility model, the privacy mechanisms of PR−LBS, and the cost function which enables the location–service exchange of PR−LBS as will be evident later.

4.6.1 Mobility Model

Topology: PR−LBS views the topology of the area as an unweighted and undirected graph $G = (Z, E)$, where Z, the set of zones represents the nodes and E is the set of edges of the graph representing the transitions between neighboring zones. Each edge, e, is associated with the time, $t(e)$, the user takes to travel along it. In a typical public space, all zones are reachable from the entrance, making the graph connected. We can then define a path as the sequence of visited zones (of interest to the user) in the graph as: $p_l = \langle z_k \rangle_{z_k \in Z, z_k \neq z_{k+1}}$, where l is the path length (number of zones) and z_k is the k^{th} zone of the path. To count a zone as part of the path, the user must *visit* the zone and stay there for at least 30 seconds, not just passing through.

We define two functions in the graph G: the distance between two zones in the graph $d_G(z_i, z_j)$ is the length (number of edges) of the shortest path between z_i and z_j; and the time between two zones $t_G(z_i, z_j)$ as the shortest time it takes the user to travel between two zones (taken as the shortest path when the weights in the graph are considered as $t(e)$ instead of 1).

PR−LBS only releases the path defined above (p_l) or a variant thereof. Therefore, it

Table 4.1: The symbols table.

Symbol	Meaning
Z	set of zones in the area of interest
N	number of sessions
p	path traversed in a session
$l[p]$	length of traversed path in a session
$n[p]$	number of observations of path p
$P(p) = n[p]/N$	probability of traversing a path p
p_l	a single path p of length l
P_l	the set of paths of length l (all paths p_l)
$d_G(z_i, z_j)$	distance between two zones: z_i and z_j
$t_G(z_i, z_j)$	shortest time to travel between two zones

hides the intra-zone as well as low-level mobility and only releases significant changes in the user's location (when visiting a new zone). As such, consecutive zones in the path need not be geographical neighbors; the path, as we define it, is not equivalent to the actually traversed path, but rather a part of it. For example, the user might have traveled along the zones A-B-C-D, but only spent time at A and C. PR−LBS releases the path (or a variant of) A-C.

Sessions: The user's mobility is broken down into *sessions*. In each session, the user enters the area, traverses a path, and then leaves; i.e., a session maps to one traversed path. We focus on the path as it embodies all the information about the user's mobility including the zones of interest, their priority and importance to the user, and other tracking information. The path starts at the beginning of a session and ends at the end of the session; subpaths do not count as independent paths. Therefore, each session will be associated with one path p of length $l[p]$.

We model the user's mobility as the probability distribution of a set of paths s/he traverses. PR−LBS populates this mobility model empirically based on the SP's observations (zones that PR−LBS revealed to the SP). After N sessions, the SP observes the user traversing a path p for $n[p]$ times. Each path in the mobility model is a distinct event; the probability of the user traversing each path (as observed by the SP), $P(p)$, is simply the

count of the path divided by the number, N, of the user's sessions, $P(p) = \frac{n[p]}{N}$.

The set of paths of equal lengths ($P_l = \{p|l[p] = l\}$) forms a probability distribution: $\forall p \in P_l, P(p) = \frac{n[p]}{N}$. The probability of each path is the probability of the user following a path of the same length in a session. As some sessions will not have a path of length l, the probability distribution will include the event of the user not following a path of such length denoted by $P(\langle\phi_l\rangle) = 1 - \sum_{p \in P_l} P(p)$.

4.6.2 Private Location Release Mechanisms

PR−LBS protects the privacy of the user's mobility by anonymizing the traversed paths at runtime. PR−LBS has to guarantee an entire path's privacy while sequentially releasing zones along the said path, i.e., before it knows what the path is going to be. This is very different from most of the existing approaches that consider offline private publishing of mobility traces including the works by Rastogi and Nath [29], Abul *et al.* [30], Terrovitis and Mamoulis [31], and Chen *et al.* [32]. Moreover, adding noise, drawn from a distribution [66, 117], on the user's visited location does not apply in the indoor case. In most cases, the user's location is defined in terms of a zone, such as a UUID of an iBeacon, rather than a geographical location ($< x, y >$) so that noise drawn from some planar Laplacian distribution can not be added to a UUID value of an iBeacon.

The main privacy protection of PR−LBS comes from anonymizing the user's path, i.e., releasing a path, $path_{obs}$, instead of the actual traversed path. In particular, PR−LBS aims to provide (ϵ, d_m) differential privacy [115, 118, 119, 120, 121] such that:

$$P(path_{obs}|path) \leq e^{\epsilon} P(path_{obs}|path'), \tag{4.1}$$

where $d(path, path') \leq d_m$ and $P(path_{obs}|path_{tr})$ is the probability of observing $path_{obs}$ given the user traversed $path_{tr}$.

The criterion of Eq. (4.1) states that the privacy preserving mechanism releases a path, $path_{obs}$, (observed by the SP), such that the probability of this path being the result of

applying the privacy mechanism on the actually traversed path is indistinguishable (to an exponential factor) from that of applying the same mechanism on another path at most a distance d_m from the actual path. In other words, the SP, after observing $path_{obs}$, can not identify the user's actual path. The user's actual path is indistinguishable among the set of paths within a distance d_m from the user's actual path – we refer to this set as $path_{d_m}$.

The main challenge here is that PR−LBS can not treat the user's path as a series of zones devoid of any geographical significance. Blindly attempting to satisfy Eq.(4.1) will help the SP narrow down the search space by eliminating some implausible paths from $path_{d_m}$, given the released path. For example, a user enters a tunnel that can only be traversed in one direction: A-B-C-D. If PR−LBS releases any path of length 4, then the SP will directly infer the user's original path as A-B-C-D (the only plausible path in $path_{d_m}$ regardless of the value of d_m in this case).

This challenge arises from the fact that for a certain path $pa \in path_{d_m}$, $P(pa|path_{obs}) = 0$ (implausible given the observation) so that by Bayes' rule $P(path_{obs}|pa) = 0$, which violates the promised differential privacy guarantees. PR−LBS ensures that for $\forall pa \in path_{d_m}$, the probability $P(pa|path_{obs}) > 0$ so that the SP can not reduce the size of the search space after observing $path_{obs}$. PR−LBS's privacy preserving mechanism need not ensure the observed path to be plausible per-se, but any path in $path_{d_m}$ must plausibly be the actual path, given the released $path_{obs}$.

Recalling that a path is a sequence of zones associated with time, a plausible path is one which the time separating each two consecutive zones allows for a person to travel between them. While the SP has access to the area's map, PR−LBS relies on previously recorded user mobility to populate the graph G describing the layout. Each time PR−LBS observes a new transition, it adds the newly observed edge to the graph along with the travel time. Eventually, PR−LBS populates the graph and uses it to compute the travel time between any two zones in the graph.

4.6.2.1 Differential Privacy (D.P.) Mechanism

PR−LBS chooses to release the user's visited zone with a probability q_0 and chooses a zone at a distance of i with a probability $q_i = \alpha^i.q_0$ such that $\sum_i \alpha^i.q_o = 1$ and $i < d_m$, where d_m is the indistinguishability threshold. We define the distance between two equal-length paths ($d(path, path')$) as the edit distance with a non-negative weight. The only operation we consider in the edit distance is substitution so that the weight/cost of each substitution is the distance ($d_G(z_1, z_2)$) between the two substituted zones. For example, the distance between two paths $A - B - C - D$ and $A - E - C - F$ is: 0 (cost of sub A with A) + $d_G(B, E)$ (cost of sub B with E) + 0 (cost of sub C with C)+ $d_G(D, F)$(cost of sub D with F). Since the weight between two zones is symmetric ($d_G(z_1, z_2) = d_G(z_2, z_1)$), the distance between two paths satisfies the axioms of a metric.

So, this mechanism achieves differential privacy such that (proof in Appendix A):

$$q_0 \leq \frac{1}{\sum \alpha^i} \; s.t. \; \alpha \geq \frac{|Z|_{d_m}}{e^{\epsilon/m}}, i < d_m, \tag{4.2}$$

where $|Z|_{d_m}$ is the number of zones within a distance d_m of the user's visited zones.

When the user moves from a zone za_l to za_{l+1} (with a travel time ta), s/he would have traversed a path pa of length $l + 1$ so far. The D.P. mechanism releases a zone z instead of za_{l+1} according to probability distribution described above. At the same time, PR−LBS keeps track of $path_{d_m}$, the set of paths of equal length of pa and of a distance less than d_m from pa. For each path p_r (comprised of zones zr_i) in $path_{d_m}$, PR−LBS estimates the travel time of the transition from zr_l to zr_{l+1} as $t_r = t_G(zr_l, zr_{l+1})$. If there is at least one path of $path_{d_m}$ where $t_r >> ta$, then PR−LBS hides za_{l+1} completely (and does not release any anonymized zone). Therefore, PR−LBS avoids releasing a path to the SP that violates the indistinguishability criterion. At the end of Section 4.8, we will show that PR−LBS can effectively distort the distribution of the zone visit time for privacy-oriented users. This prevents the SP from utilizing timing information to infer more probable paths from the set

$path_{d_m}$.

The value of d_m controls the trade-off between privacy and utility. A lower value of d_m will allow the privacy criterion to be more relaxed (a higher value of q) so that the observed path will be closer to the actual path, thus becoming indistinguishable among a smaller set of paths.

Learning: While PR−LBS is populating the graph, it can not apply the above mechanism as it will not have a full view of the area's topology (a list of zones without transitions). In such a case, it applies a variant of the D.P. mechanism. In this variant, PR−LBS releases the user's visited zone, z_v, with a probability q or any other zone $z \in Z \setminus z_v$ with a probability $1 - q$. We define the distance, $d(path, path')$, between two paths $path$ and $path'$ as the edit distance with weight 1 (=the number of different zones between two paths). This mechanism achieves (ϵ, d_m) differential privacy (proof in Appendix A), where

$$q \leq \frac{e^{\epsilon/d_m}}{|Z| - 1 + e^{\epsilon/d_m}}. \tag{4.3}$$

This mechanism is not very efficient in terms of the privacy–utility tradeoff; it treats all the other zones as being equidistant to the current zone. Once the full topology of the graph is known, PR−LBS applies the full D.P. mechanism which exhibits a finer-grained privacy–utility tradeoff by providing indistinguishability over topologically close paths.

4.6.2.2 Anonymity Set (A.S.) Mechanism

In some scenarios, PR−LBS runs on a device with no capability of feeding the SP app with a fake zone instead of the user's visited zone; it can control whether to release or hide the currently visited zone. Therefore, PR−LBS can not apply the D.P. mechanism described above.

Instead, PR−LBS resorts to the A.S. mechanism that releases the user's visited zone with a probability q and hides it with a probability $1 - q$. This mechanism can not provide differential privacy guarantees since there will always be a path such that the observation

probability will be 0. For example, if PR−LBS releases the path $p_{obs} =< a, b >$, then the probability of observing this path given the user traversed $< c, a, d >$ is 0. The expression of Eq. (4.1) can not be satisfied. Therefore, we focus on another privacy indicator which is the size of the anonymity set. The anonymity set is defined as the set of paths from which the released path could have possibly resulted, i.e., $\forall p | P(path_{obs}|p) > 0$. In appendix A, we derive the expected size of the anonymity set for a traversed path of length m as:

$$E(S) = \sum_{k=0}^{m} \sum_{r=0}^{k} \binom{k}{r}^2 q^r (1-q)^{k-r} \frac{(|Z|-r)!}{(|Z|-k)!} \tag{4.4}$$

As evident from the expression of Eq. (4.4), the value of q controls the uncertainty at the SP side. For instance, when $q = 0$, the value of $E(S)$ assumes the maximum value since the adversary will not observe any of the user's mobility. The mobility inside a zone and between two released zones (z_i and z_j in this case) is always hidden. During this gap, the user could have spent time at z_i or in one or more zones in between (on path p_l). The SP can not definitely decide whether the user actually visited a zone in between (on path p_l) or spent the entire time between at z_i.

4.6.2.3 Path Diversity

Both D.P. and A.S. mechanisms rely on hiding the user's visited path within an anonymity set of other paths. The size of this set provides the privacy guarantees to the user and is mainly controlled by the number of zones in the area and the possible transitions between these zones as recorded by previous user mobility. It is very important to note here that while mobility can be restricted in an indoor case, our definition of a zone visit relaxes this restriction. In particular, in an indoor space, the graph depicting the area's topology is connected so that every zone is reachable from any other zone.

Since consecutive zones in a path are not geographically adjacent, the number of zones reachable from the currently visited zone is not restricted to neighboring zones, but by their feasible transitions. PR-LBS copes with the issue of limited feasible transitions, which

takes place at the bootstrapping stage, by maintaining the size of the anonymity set for both mechanisms. When the size of the anonymity set is small, PR-LBS hides the currently visited zone.

4.6.3 Information Disclosure

In what follows, we analyze the (privacy) cost incurred from PR−LBS's release of a path to the SP (even if it is anonymized). Any observation (a zone visit) will necessarily change the probability distribution spanning the mobility model. The amount of change introduced to the mobility model is what we attempt to quantify. Even when PR−LBS releases an anonymized path, this path will still carry information of the actual mobility.

We first quantify the information disclosure (alternatively leak) for the entire user's path after observing a new zone visit and then state our information leak model for the specific zone visit.

4.6.3.1 Per-path Criterion

To quantify the information leak for observing a path, we follow the lead of Miklau and Suciu [122] by considering a metric of positive information disclosure. We focus on the improvement of the probability of the user traversing a certain path as being indicative of the amount of information released:

$$lk(s, v) = \sup_{s} \frac{P(S = s|v) - P(S = s)}{P(S = s)}. \tag{4.5}$$

In Eq. (4.5), $P(S = s)$ is the *prior* probability distribution of a secret s that the adversary attempts to identify, v is the observation, and $P(S = s|v)$ is the probability distribution of S after observing v.

In our setting, the secret that the adversary wants to unravel is the user's probability distribution of the paths traversed. When PR−LBS is about to release a new zone z_l to the SP, after releasing a path p_{l-1}, it estimates the information leak of the total observed

path being $p_l = \langle p_{l-1}, z_l \rangle$. The information disclosure considers the improvement of the SP's observation probability of the user traversing a path as being indicative of the amount of information released. If the observation of a path does not improve the adversary's knowledge, it leaks little/no information about the user to the SP (the SP already expects the user to traverse such a path), and vice versa.

If the user has visited the area N times (number of sessions), out of which s/he traversed the path p_l for $n[p_l]$ times, then the per-path information leak is (see the derivation in Appendix A):

$$lk(p_l, z) = \frac{1-a}{a(N+1)},\ a = \frac{n[p_l]}{N}. \tag{4.6}$$

4.6.3.2 Per-zone Criterion

We can rewrite the leak function of Eq. (4.6) to to represent the information leak in bits as: $leak(p_l, z) = \log_2 (lk(p_l, z) + 1) = \log_2 \left(\frac{N(n[p_l]+1)}{n[p_l](N+1)} \right)$.

It is worth noting that $leak(p_l, z)$ represents the improvement of the observer's knowledge of traversing a path, p_l, directly after the observation of z. Having visited N sessions, of which a path p_l has been traversed $n[p_l]$ times, the probability of visiting path p_l is originally $P(p_l) = n[p_l]/N$. For the $(N+1)^{th}$ session, if p_l is traversed, it will be the only path (of the user's mobility model) experiencing a positive information disclosure as $P(p_l|z)$ will be $(n[p_l]+1)/(N+1)$, which represents $P(S = s|v)$ of Eq. (4.5). It is straightforward to show that for $N > n[p_l]$ (which is always the case since N is the total number of sessions), $\frac{n[p_l]+1}{N+1} > \frac{n[p_l]}{N}$ so that the information disclosure will always be positive. Therefore, applying the logarithm to compute $leak(p_l, z)$ is always feasible.

For $N > 0$, $n[p_l] > 0$ and $n[p_l] \leq N$, the value of $leak(pv_n, z)$ varies between 0 (minimum leak) and 1 (maximum leak). In the initial case where no mobility has been observed about the user ($N = 0$ or $n[p_l] = 0$), we associate $leak(p_l, z)$ with the maximum value of 1. We define the information leak per observed zone as the difference between the

leaks resulting from the paths p_l and p_{l-1}:

$$\begin{aligned} leak(z) &= leak(p_l, z) - leak(p_{l-1}, z) \\ &= \log_2 \left(\frac{n[p_{l-1}](n[p_l]+1)}{n[p_l](n[p_{l-1}]+1)} \right). \end{aligned} \tag{4.7}$$

A closer look at Eq. (4.7) reveals that the leak per zone is the leak defined by the conditional probability distribution $P(z_l|p_{l-1})$ which is $\frac{n[p_l]}{n[p_{l-1}]}$. Since PR−LBS considers the user mobility one zone at a time, by the time the user reached z_l, the entire path p_{l-1} must have been observed by the SP. Hence, $leak(z_l)$ focuses on the additional leak incurred from releasing z_l given that the SP has already observed the user's path p_{l-1}. Appendix B provides an example of how PR−LBS computes the privacy cost of a traversed path.

This information leak is especially crucial for the case of the A.S. mechanism which will release raw location data. We rely on the information disclosure as a cost metric to cap the additional knowledge leaked about the user to the service provider.

Finally, the information leak as defined in Eq. 4.6 offers a nice property. If the value of N is large enough then the information leak from observing path p_l can be approximated by $\frac{1}{n[p_l]}$. This implies that the first observations of a path will have higher leaks compared to future observations. On the other hand, when the value of N is low, there is always going to be a leak of information, as the probabilities of following paths will be changing more abruptly. For a fixed N, a lower probability $P(p_l)$ of traversing a path will always lead to a higher information leak. Individuals tend to perceive behaviors with low probability as being more private because they indicate unexpected (thus conspicuous) behaviors [96, 123].

4.7 PR-LBS

In what follows, we describe PR−LBS, its different components, and their interactions.

4.7.1 The Location-Service Exchange

We model the interactions between the user and the SP as a repeated play model [107] composed of the user (player) and the SP (opponent). The user, with PR−LBS acting on his/her behalf, chooses one of three actions: *hiding*, *releasing*, or *anonymizing* the location. In return, the SP pushes a service with varying value.

We rely on the SEA algorithm [108] to decide on which action to take at each phase (when visiting a new zone). In this model, the player has access to a set of experts each offering an advice for the action to take at each interaction. The algorithm is akin to reinforcement learning and is based on a combination of exploration and exploitation. The exploration phase will enable the player to learn the opponent's response to the different actions, while the exploitation phase enables the player to follow the expert's advice who has accumulated the highest average utility.

SEA has some nice properties that make it suitable for our context. First, it assumes a non-oblivious opponent whose actions might (or might not) depend on the player's actions (as in our case) as opposed to minimum regret algorithms [107]. SEA, also, avoids being short-sighted in its objective and instead focuses on the asymptotic behavior of the player. Finally, it can achieve a stricter bound if the opponent is assumed flexible. A flexible opponent is one who forgets the player's actions after a while. Analytics and advertisement servers constitute a relevant example, their user recommendations are usually based on recently observed behavior.

The user–SP interaction takes place when PR−LBS detects the user has visited a new zone. PR−LBS has a set of three experts with each recommending an action to follow as defined in Table 4.2.

The mechanism invoked by the second expert of the exchange module will depend on the capabilities available to PR−LBS. If PR−LBS can change the zone reported to the SP's app, then it can rely on the D.P. mechanism (while running in device-based mode on a rooted device or in infrastructure-based mode). Otherwise, the second expert uses the

Table 4.2: The experts utilized by PR−LBS.

Expert	Advice
First	Hides location all time
Second	Release location according to privacy mechanism
Third	Release location all time

A.S. mechanism when PR−LBS can only enable/disable location release (while running in device-based mode on an unrooted device).

The player can designate an interaction as either an exploration or exploitation stage according to a biased probability distribution. Specifically, PR−LBS designates the i^th interaction as an exploration stage with a probability $1/i$ and as an exploitation stage with probability equal to $1 - 1/i$. In the exploration stage, the player chooses one expert at random. This ensures that every expert is sampled infinitely many times. In the exploitation stage, the player simply chooses to follow the advice of the expert with the highest accumulated average utility. At the earlier interactions, where the value of i is small, PR−LBS chooses exploration with a higher probability as to learn the utilities of the different experts. With more interactions, the behavior of PR−LBS stabilizes and it chooses exploitation with a higher probability as it would have accumulated enough average utility for each expert.

After PR−LBS chooses an expert, it follows its advice for the coming interaction between the user and SP. Every interaction involves deciding on an action (based on the expert's advice), computing the privacy cost of the performed action, estimating the service value of the whole interaction, and computing the utility resulting from the interaction (a function of the cost and reward). After the interaction ends, PR−LBS updates the average utility for the chosen expert. PR−LBS, then, chooses another expert for the next interaction.

Table 4.3: The privacy profile of a privacy-oriented user.

	No Service	**Some Service**	**Full Service**
No Privacy	0	0	1
Some Privacy	0	0	1
Full Privacy	1	1	2

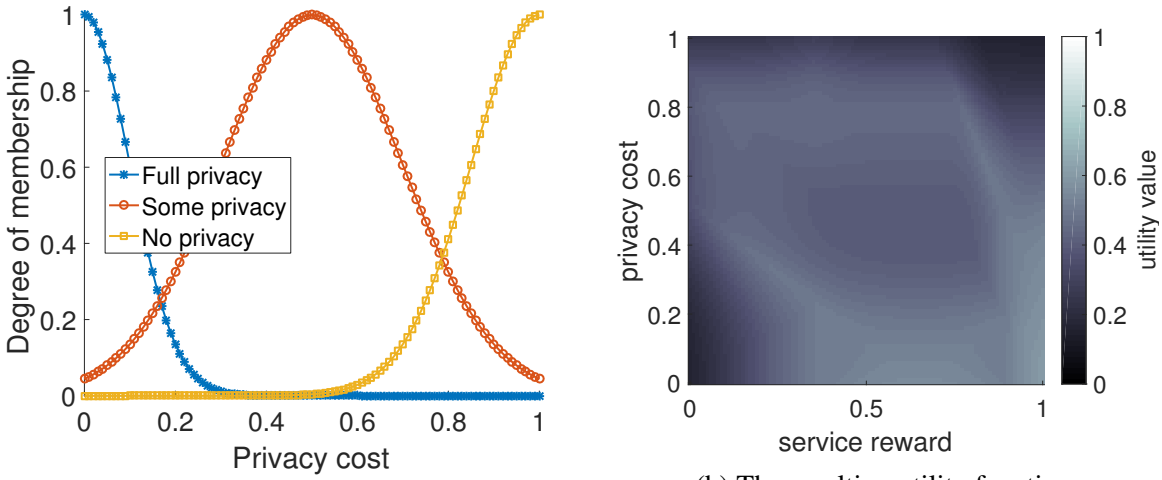
4.7.2 Utility Estimator

Deriving a utility estimate from the privacy and QoS estimates is not straightforward; it is like comparing apples to oranges. Besides, the utility function is subjective as it depends on the user's perception. A privacy-oriented user will suffer lower utility if more location samples are released, while a service-oriented user will suffer lower utility if s/he does not receive services.

In PR−LBS, the user defines a privacy profile which indicates his/her tradeoff between the cost of releasing location and the benefit from the received service. The utility estimator module converts this "high-level" profile to a utility function that maps the privacy cost and service quality pair to a value between 0 (no utility) and 1 (full utility). Table 4.3 shows an example of the privacy profile of a privacy-oriented user (from our survey). In the survey, we asked respondents to fill in their privacy profile through a table similar to Table 4.3. Each entry specifies the respondent's satisfaction (0 – not satisfied, 1 – somehow satisfied, and 2 – fully satisfied) value for each of the privacy and service combination. It is clear how this respondent favors curbing location sharing.

Given a privacy profile (one that looks like Table 4.3), the utility estimator module converts it to a numerical function. The resulting function takes two inputs: privacy cost ($leak(z_i)$) and estimated service quality values ($serv_i$); it returns an output which is the utility value such that $utility = f(leak(z_i), serv_i)$. As the concepts of the privacy profiles (privacy, QoS, utility) are defined in qualitative (or humanistic) terms, a fuzzy inference system (FIS) [124] is thus suitable to derive the utility function out of these values.

We rely on a Mamdani-type fuzzy inference system; such a system has two main com-



(a) Privacy membership function

(b) The resulting utility function.

Figure 4.4: Utility estimator illustration.

ponents: rules (defining relationship between inputs and outputs) and membership functions. In our context, the rules are defined as per the privacy profile (similar to Table 4.3). For example, the top left entry of the table defines this rule: *No Privacy* ***AND*** *No Service* $\implies$ *Not satisfied*.

The membership function defines the value's relevance to the property it is claiming over a domain. For example, instead of defining a hard threshold between what is a low threat path and a high threat one, one can define softer thresholds, as evident from Fig. 4.4a. A path definitely poses a low threat (full privacy) when $leak(p) = 0$, as the value of 0 has a membership of 1 in the low function. The membership value decreases gradually as the privacy value increases until it hits 20%. Similar membership functions can be defined for medium, high threat. A similar logic applies for the service and utility values.

The second step in the fuzzy inference system is mapping the inputs to the output; this is achieved through the fuzzy operators, which results in a fuzzy utility value. The final stage is the conversion of a fuzzy utility value to a crisp output to determine the actual utility to feed the SEA algorithm through the defuzzification step. Ultimately, the FIS will result in a continuous and smooth 2D function mapping both the privacy and service values to a utility value. Fig. 4.4b shows the final utility function for the privacy profile of Table 4.3.

The utility takes its maximum value at maximum privacy and reward levels. The utility, smoothly, decreases as the privacy level or reward decrease.

4.7.3 QoS Analyzer

Measuring the quality of service the user received from interactions with the SP's app is essential to the operation of PR−LBS. Market research literature has a wealth of studies that analyze the user's retail app usage and its effect on user satisfaction and purchases [125, 126, 127, 128, 129, 130, 131, 132, 133]. This literature leads to the following conclusions regarding retail apps:

1. Retail apps rely heavily on push notifications to communicate retail services to users, which we confirmed from our analysis of multiple retail apps. In 2014, more than 80% of the notifications pushed by the retail apps were consumed by users [130, 134].

2. Continued app usage and interaction inside the store (during shopping) directly relates with user's satisfaction during the shopping experience [125, 126, 127, 128, 129].

3. Higher retail app usage rate (on-screen time) during shopping is correlated with more brick-and-mortar store visits, longer shopping visits, and increased purchase rates [131, 132, 133].

Consistent with market research literature, we utilize user interaction with the SP's app as an indicator of the user's satisfaction with the services provided by the app. An interaction with a typical retail app takes place in three stages: (1) the app pushes a service to the user through a notification, (2) the user consumes the notification (by checking and reading it), and (3) the user opens and interacts with the SP's app. To measure user interaction with the SP's app, PR−LBS observes if the user consumed a push notification from the app and measures the time s/he interacted with the app.

PR−LBS monitors the level of user–app interactions through the collector module to compute the QoS metric: $serv_i$. It observes the time the user spent interacting with the SP's app, denoted by $time_{foreground}$, that was preceded by a consumed push notification from the SP's app. Particularly, if the user consumed a push notification by opening the SP's app, then PR−LBS measures the fraction of time the user spent actively interacting with the app as:

$$serv_i = \frac{time_{foreground}}{time_{z_i - z_j}}, \tag{4.8}$$

where $time_{z_i - z_j}$ denotes the time spent in the last zone(z_i), before moving to the new zone(z_j).

To further assess whether $serv_i$ is a good indicator of user satisfaction with retail apps, we analyzed retail app usage data from two datasets: the LiveLab dataset of Rice University [81] and our dataset of participants whom we recruited from PhoneLab [80]. The Livelab dataset contains the app usage data, along with other data, for 34 iPhone users over a 12–18 month period. Our dataset has app usage data for 95 Android users over 4 months. We identified retail apps from their app category in either Google Play or iTunes. For every retail app and user combination, we calculated $serv_i$ as the app total usage time (in seconds) normalized by the installation time (in days). We then associated every app session with its average rating on the app store. We categorized apps into two categories, in terms of rating: low (rating ≤ 2.5) and high (rating ≥ 4).

Fig. 4.5 plots the distribution of the $serv_i$ for each of the two categories (low and high) for both datasets. There is a large discrepancy between the high and low distributions suggesting that highly rated apps tend to enjoy higher usage rates. For the apps with a low rating, 80% of the apps are used at most 0.1 sec/day, while 80% of the highly used apps have more usage rate than that. The discrepancies of values between both datasets relates to the duration of each dataset (Livelab is much longer than PhoneLab). It is clear that $serv_i$ correlates with the average user rating of the app at the app store (iTunes/Google Play) –

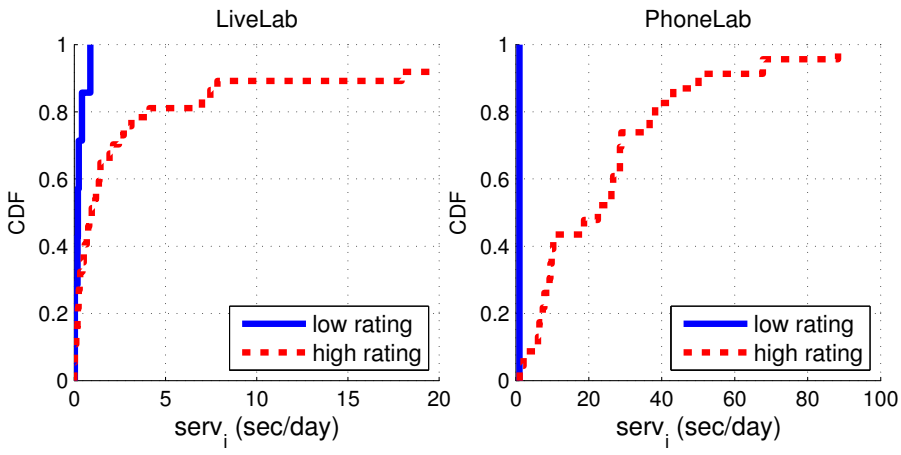


Figure 4.5: The distribution of QoS metric for the LiveLab (left) and PhoneLab (right) datasets.

app user rating acts as an indicator of user satisfaction of its service [135].

The proposed reward metric offers several advantages in terms of practicality and usability. **First**, it limits interactions with the user. Existing methods that rely on surveys to measure user satisfaction do not apply in our context. PR−LBS needs to measure user feedback at a "micro-scale" as the user is moving from one zone to the other, not after the visit is completed. It is impractical to continuously ask the user for feedback about the received service as s/he moves from one zone to the other. **Second**, the reward metric is app-agnostic; it does not require specific knowledge of the semantics of the service provider's app. Otherwise, PR−LBS has to tailor its estimation methodology of the service rewards to every service provider app, which is impractical. **Finally**, $serv_i$ is practical to measure as it can be extracted from user-level information on the user's device. It requires neither intercepting network traffic nor instrumenting the user's mobile operating system.

4.8 Implementation and Evaluation

We now present the implementation of PR−LBS in device mode and the evaluation of its effectiveness.

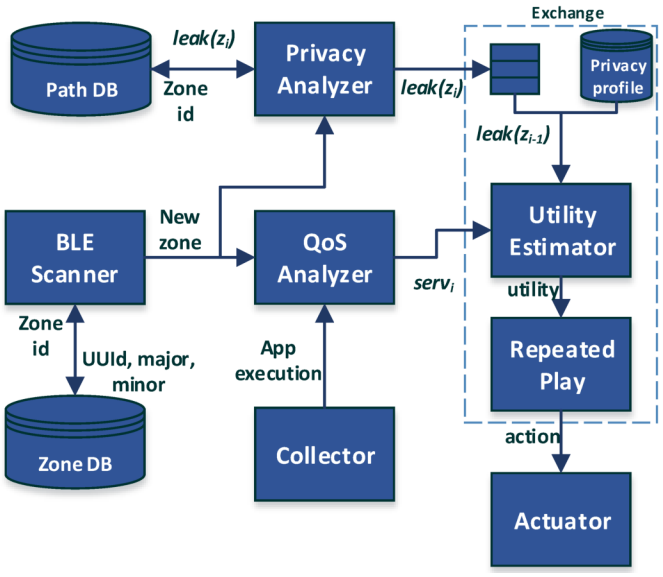


Figure 4.6: The architecture of PR−LBS in device mode.

4.8.1 Implementation

We implement the device mode of PR−LBS as a standalone Android (4.4) app, which is compatible with beacon-based localization. Fig. 4.6 shows the architecture of the PR−LBS app. PR−LBS runs as a background service that detects if the user is visiting a place where localization is deployed. When the user starts the visit, PR−LBS prompts him/her to set the privacy preferences and executes the logic of Fig. 4.6. The privacy preferences include setting the privacy level and the privacy profile of Fig 4.8. The privacy level ($priv_{lvl}$) is a slider between no privacy $priv_{lvl} = 0$ and full privacy ($priv_{lvl} = 1$) that sets the parameters of the privacy mechanisms as such:

D.P. Mechanism: if the maximum distance (length of shortest path) between any to zones in the area is d, then $d_m = priv_{lvl}.m.d$. When its learning variant is running, then d_m is set as: $d_m = priv_{lvl}.m$, where m is the average path length.

A.S Mechanism: the probability controlling the release of location is simply set as $q = 1 - priv_{lvl}$.

BLE Scanner: PR−LBS utilizes Android's Bluetooth Low Energy (BLE) APIs to scan regularly for BLE beacons. During the scan duration, PR−LBS receives advertisements

from multiple beacons. It decides on the beacon with the lowest power attenuation as the closest to the user. It extracts the identifying fields from the beacon advertisement to maps a zone *id*. If the new zone is different from the last detected zone, the scanner alerts the QoS and Privacy analyzers. `PR-LBS` uses the scanned zones to populate the topology graph.

Collector: This module records app execution events and keeps track of the time the user spent for interacting with the service provider's app. As Android does not provide a public API for this purpose, the collector module frequently polls (once a second) the running tasks to find which apps the user is currently running in the foreground. It also runs an an `Accessibility Service` to intercept the SP app's notifications and the resulting user actions. Whenever a new zone is detected, the collector passes this information to the QoS analyzer that calculates the QoS metric.

Actuator: The actuator is responsible for performing the action decided by the exchange module. The action could be *hide*, *release*, or *anonymize* the visited zone. While running on an unrooted device and with the absence of any other support, the only actions available are to release or hide the currently visited zone (A.S. mechanism). In such case, `PR-LBS` uses the Android's Bluetooth Admin permission to globally control Bluetooth scanning on the device. This will prevent the service provider's app (and potentially other apps) from tracking the users. On the other hand, when running on a rooted device, `PR-LBS` will be able to apply the *anonymize* action. We instrument the Bluetooth scanning function in Android's framework to so that `PR-LBS` changes the Bluetooth Low Energy scan results that the SP's app will receive. We are currently exploring using the user's smartwatch to advertise dummy beacons to anonymize the currently visited zone which will not require rooting the user's smartphone.

4.8.2 App-Based Evaluation

We evaluate the energy overhead of `PR-LBS` when it runs on Samsung Galaxy S5 (Android 4.4.4). We deployed a set of iBeacons in a lab environment with one iBeacon

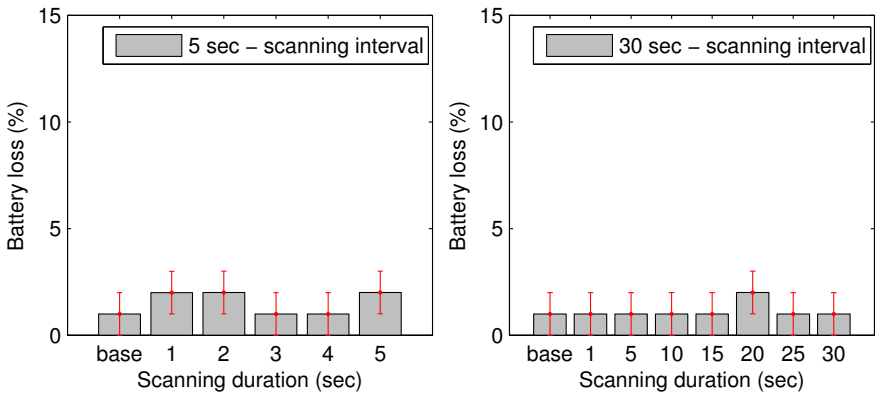


Figure 4.7: The energy consumption by PR−LBS.

(closest to the device) continuously changing its identifiers, making PR−LBS believe a new zone was detected upon each BLE scan. Each new zone event triggers PR−LBS to execute its components (Fig. 4.6). Hence, the frequency of detecting new zones along with the scanning interval (frequency) and duration (length of the scan) determine PR−LBS's energy consumption. We report on the battery energy consumption (which includes the full operation of PR−LBS with all of its components) in Fig. 4.7 when PR−LBS runs under two scanning intervals: 5s and 30s. For each scanning interval, we vary the scan duration to study its effect as shown in Fig. 4.7. We also compare the battery loss with the base case, with PR−LBS turned off. We run all the experiments for 10 minutes with the screen fully lit while turning off background apps, Wi-Fi, and data connections. It is clear from Fig. 4.7 that PR−LBS incurs limited energy overhead since PR−LBS is a lightweight app that incurs little processing overhead.

We also test the usability of the privacy profile input interface (Fig. 4.8). We deployed PR−LBS on Google Play and asked 100 participants (recruited over Amazon Mechanical Turk) to test the app and answer a set of questions. We paid each participant $1 and the average time for tasks completion was 7 minutes. When the participant completed interaction with the app, it displayed a special code to input in the survey to ensure completion of the required task. Furthermore, we provided the participants with the following instructions on how to interact with UI:

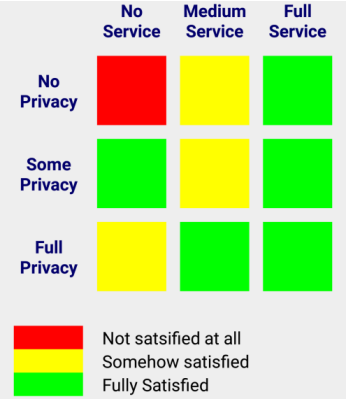


Figure 4.8: The UI to input the privacy profile in PR−LBS.

Columns constitute the service level you might receive from the store, while rows denote the privacy level you are enjoying. Please tap on each of the nine squares to indicate how much are you satisfied in different scenarios (privacy - service configuration). Every time you tap a square, it changes its color.

93% of the participants considered that the interface is easy to use and 85% of them indicated that it is clear.

4.8.3 Trace-Based Evaluation

Datasets: We utilize 6 datasets from the CRAWDAD data repository to evaluate PR−LBS. These datasets represent the mobility of individuals in public constrained spaces that PR−LBS operates in. We also utilize two other datasets that belong to respondents from our survey. In both surveys (Walmart and Nordstrom), we displayed a map of the store with labeled zones. We asked each participant to trace the path s/he traversed during the last visit. Table 4.4 shows a list of the datasets.

To evaluate PR−LBS, we transform each dataset into a stream of location samples. PR−LBS processes every location sample regardless of whether it came from the real world or a location trace, which indicates that our evaluation is representative of PR−LBS's operation in the real world. We further partitioned every stream into sessions or paths. The last two datasets, Walmart and Nordstrom, had one path per user and no time information

Table 4.4: The eight datasets used for the evaluation.

Dataset	Location	Duration (days)	# zones	# users
HOPE 2008 [136]	Hotel	3	21	1273
HOPE 2010 [137]	Hotel	3	26	1119
State Fair [138]	Fair	5	32	19
Orlando [138]	Theme park	61	44	41
NCSU [138]	Campus	80	49	35
KAIST [138]	Campus	78	44	92
Walmart	Retailer	-	29	93
Nordstrom	Retailer	-	34	76

associated with the location trace.

Scenarios: We simulated four classes of SPs that reward the user for sharing location information differently, while not rewarding for hiding location. In our model, the SP will offer the user a service with a reward value ($serv_i$) between 0 and 1. This abstracts both the SP and the QoS analyzer module. The SP classes are:

1. *None*: No reward for the user.
2. *Low*: Low reward (below 0.3) for sharing.
3. *Med*: Medium reward ($\geq$ 0.3 and $\leq$0.8) for sharing.
4. *High*: High reward (higher than 0.8) for sharing.

We consider the three privacy profiles defined in Section 4.4: service-oriented, neutral, and privacy-oriented, which we constructed based on our survey results. Each respondent filled a table exactly like Table 4.3 where each response corresponds to the privacy profile of the respondent. To generate the three privacy profiles, we average the profiles of each (as defined in Section 4.4) respondent (in the three profiles) and then round the values to the nearest integer.

We execute `PR-LBS` for each user in each dataset for each scenario (service class and privacy profile combination). At the end of each run, `PR-LBS` would have released some of the user's mobility. Our evaluation is based on comparing, for each user and in each scenario, the released paths with their original counterparts.

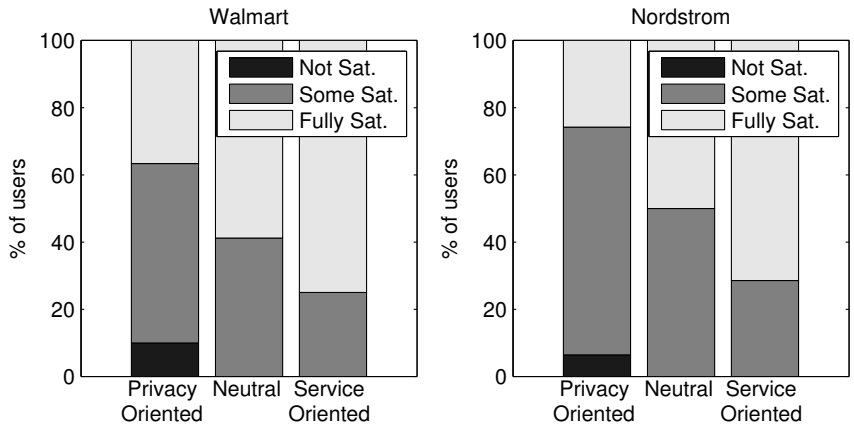


Figure 4.9: User satisfaction distribution with high service.

PR-LBS Feasibility: In Section 4.6.2.3, we indicate that even in a constrained indoor area, the number of feasible transitions (not necessarily from geographically neighboring nodes) is what matters for PR−LBS's operation. For the eight mobility datasets of Table 4.4, the number of geographical transitions is indeed limited by the area's topography. Nevertheless, the size of the set of feasible transitions (measured from user mobility) is near perfect for all the datasets. In particular, the "NCSU" dataset has 1912 feasible transitions out of 2352 possible ones, "State Fair" has 992 out of 992, "Orlando" has 1724 out of 1892, "KAIST" has 1892 out of 1892, "Walmart" has 812 out of 812, "Nordstrom" has 1122 out of 1122, "Hope 2008" has 420 out of 420 and "HOPE 2010" has 650 out of 650. The number of possible transitions in an area is simply $|Z|.(|Z|-1)$, where $|Z|$ is the total number of zones.

Since PR−LBS considers only the visited zones as composing a path, it is able to overcome constraints in an area's topology. It exploits the larger set of feasible transitions to provide a larger anonymity set for both the D.P. and A.S. mechanisms. PR−LBS need not hide zones because of infeasible transitions, which maintains utility for the user and service providers.

User satisfaction: To study whether PR−LBS matches user expectations, we execute PR−LBS over each path of both Walmart and Nordstrom datasets with the different SP

models (low, medium, and high). At the end of each run, we capture the path that PR−LBS releases and we compute the privacy metric ($leak(path)$) according to Section 4.6. We also compute the average service value received per zone. We end up, for each user, with the service value that the user received and privacy loss experienced.

We follow the same procedure as Section 4.7.2 to build a utility function for each user in both the Walmart and Nordstrom datasets from the table containing their privacy profile. Recall that these profiles are nothing else than a mapping between a privacy – service pair to a utility metric (reflecting satisfaction). We then use the privacy and service values of each path as inputs to each user's profile to estimate his/her satisfaction. We round the output to the nearest integer reflecting three satisfaction levels: "not satisfied at all", "somehow satisfied", and "fully satisfied".

Fig. 4.9 shows the user satisfaction distribution for each privacy profile and for the high service level. The percentage of unsatisfied users is close to 0 in all the situations. Also, service-oriented users (and neutral users to a lesser degree) tend to be more satisfied than other users because they accommodate more location sharing. Although not shown due to space limitation, service levels correlate with user satisfaction for all three profiles; the higher the service is the more satisfied the users are.

Privacy Protection: We study PR−LBS's effect on protecting users' privacy through $norm\_length$: the number of zones that PR−LBS released and the user actually visited divided by the total length of the original path. This metric indicates how much of the user's actual mobility information has been released.

Fig. 4.10 shows the distribution of $norm\_length$ for users with a privacy-oriented profile for the "NCSU" and "Nordstrom" datasets (other datasets show similar results, but are omitted due to space limitation). It is evident that PR−LBS curbs location sharing for privacy-oriented users with 60% of the paths hidden completely regardless of the service level and even when $p = 1$. PR−LBS shares some non-private location data (according to the cost metric) as part of its exploration stage.

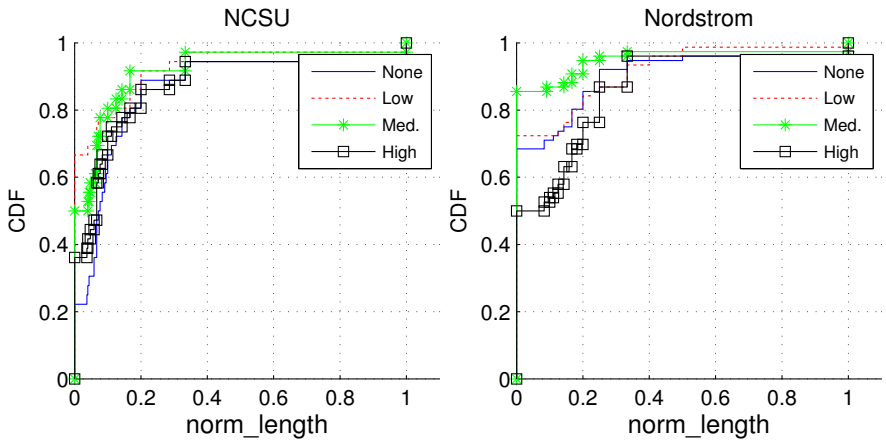


Figure 4.10: $norm\_length$ for a privacy-oriented profile.

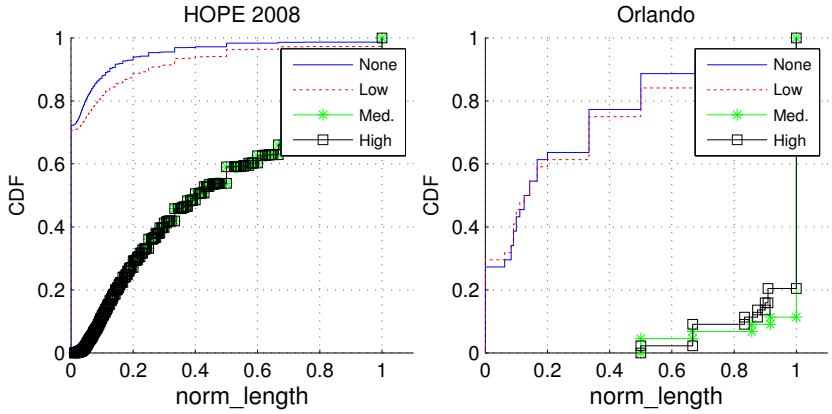


Figure 4.11: $norm\_length$ for a a service-oriented profile.

Fig. 4.11 shows the distribution of $norm\_length$ for service-oriented users in the "HOPE 2008" and "Orlando" datasets. The amount of sharing is higher than that for privacy-aware users and roughly corresponds to the SP's service level. There is one caveat: sharing is mobility-dependent. `PR-LBS` hides more of the user's location for the HOPE 2010 dataset than the State Fair dataset. In the State Fair dataset, the environment is more constrained. Individuals exhibited less diverse paths and the portion of paths leaking information according to the metric of Section 4.6 were negligible. The mobility in "HOPE 2010" dataset is diverse with most of the paths exhibiting a high privacy threat. In such a case, even if the rewards provided by the SP are high, `PR-LBS` reduces sharing to protect the user's privacy.

Service Provider Perspective: Currently, SPs focus solely on aggregate analysis in

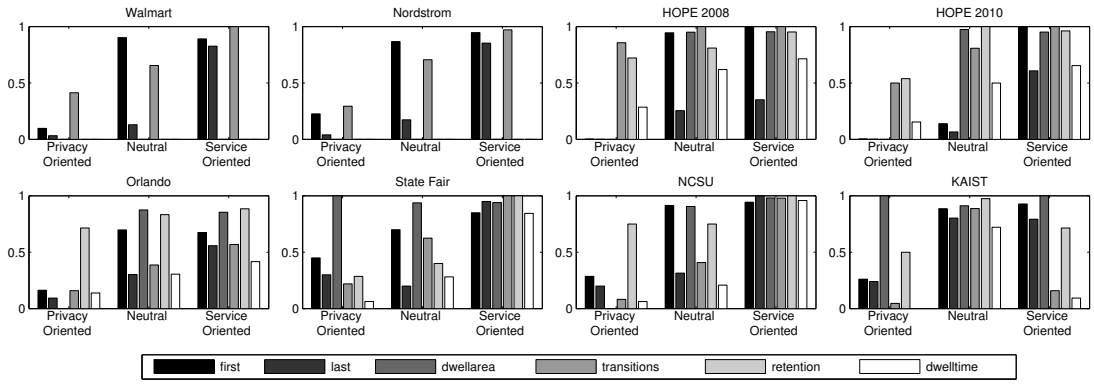


Figure 4.12: Utility metrics for realistic service levels. Some metrics in Walmart and Nordstrom datasets are not available.

Table 4.5: The utility metrics description.

Metric	**Description**
first	Portion of paths with correctly identified start zones.
last	Portion of paths with correctly identified end zones.
dwellarea	Portion of zones with accurate estimate of percentage time spent over all the users.
transitions	Portion of correctly identified zone transitions.
retention	Portion of zones with accurate estimate of retention (number of customers in a zone at a one time).
dwelltime	The portion of the zones in the area with accurate estimate of the dwell time. The dwell time of a zone is the average time spent at the zone.

an effort to comfort users. PR−LBS improves on the status-quo by enabling SPs with the capability to perform personalized analysis. In what follows, we evaluate the SP's utility using seven metrics [139, 140] as defined in Table 4.5. The closer these metrics are to 1.0, the higher is the utility that the SP enjoys. To model the SP's service level, we relied on the service value estimates of Fig. 4.5 from our Phonelab dataset, instead of using the synthetic values. We normalized the service values to fall between 0 and 1. Fig. 4.12 shows the metrics for each dataset and user privacy profile for these realistic service values.

First, the performance of PR−LBS is consistent among the different datasets which shows it can adapt to different environments and settings. **Second**, PR−LBS ensures a decent utility level even a significant portion of the users' mobility is not shared. Fig. 4.11 (left) shows that PR−LBS hides a significant portion of users' paths to protect their privacy

for HOPE 2010 dataset (the results are similar to HOPE 2008). Still, the utility metrics are fairly accurate as they represent an aggregate of users' mobility. **Third**, as shown in Figs 4.15 and 4.14 (in Appendix C for space considerations), PR−LBS adapts sharing to the service level; when the server is more rewarding, PR−LBS will react by sharing more of the user's mobility, and vice versa. PR−LBS can incentivize the SP to reward the user with better service as it reflects with improved utility. The SP's utility decreases when it provides lower service to the users.

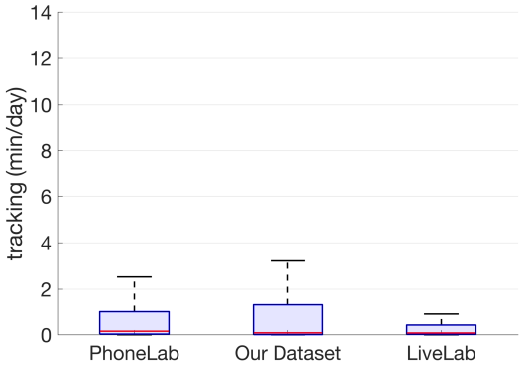
Finally, PR−LBS effectively protects the privacy of the privacy-oriented users by releasing fewer data about them. More importantly, the SP's accuracy in deciding the user's dwell time is always less than 10%. This indicates that the SP can not use the dwell time distribution for these users to identify possibly hidden zones from the timing information in the observed path.

4.9 Limitations

Lack of User Feedback: To achieve a usable and practical user experience, we made a conscious decision not to require user feedback regarding privacy decisions and rewards estimation. While PR−LBS attempts to estimate the privacy cost objectively and provide privacy guarantees, it does not capture the user's stance towards hiding or revealing every visited zone. Similarly, the reward metric of Section 4.7.3 might not be very accurate in describing the user's satisfaction with the provided service. In the future, we will investigate mechanisms to incorporate user feedback, in a usable manner, to improve the privacy and service estimations.

Evaluation Methodology: Our evaluation methodology suffers an inherent limitation. It assumes that user's privacy preferences are stationary, while they are prone to change if the user is presented with information about the SP's access to his/her location. Although our survey (see Section 4.4) results indicate the respondents' privacy preferences did not change before and after we informed them about retailers accessing their location, we be-

(a) Tracking time (min/day).



(b) Portion of identified PoIs.

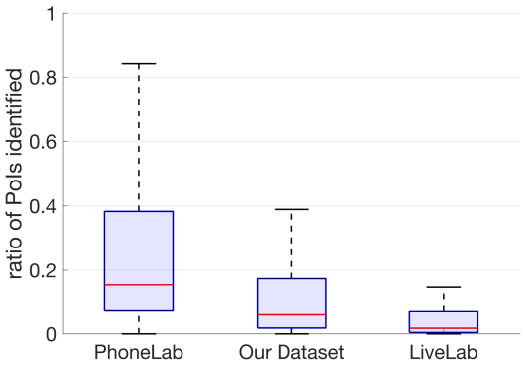


Figure 4.13: Location tracking metrics of smartphone apps running in the foreground.

lieve this part warrants additional investigation in the future.

4.10 Relation to Outdoor Location Privacy

Before we conclude this chapter, we address the relevance of the mobility model of Section 4.6.1 to the outdoor scenarios as manifested by smartphone apps. In this chapter, we model an individual's movement, in an indoor environment, as a series of traversed (with minimum staying time) zones during a single visit. This model applies to modeling the user's mobility in an outdoor scenario as a series of visited PoIs during a single day.

We opt not to utilize this model for the outdoor case; instead, we focus on modeling the user's mobility in terms of frequency of visits to the different PoIs. The main reason for this choice is the nature of the adversary we are considering. In the indoor scenario, the service provider is monitoring the entire mobility of an individual in a constraint environment comprising less than 50 zones (Table 4.4). The outdoor case (of smartphone apps), on the other hand, exhibits a scenario of different nature. A single smartphone app, running in the foreground, accesses location only in a sporadic manner as we indicated earlier in Chapters II and III.

To put this in context, we show, in Fig. 4.13 the amount of information an app collects

about users over extended periods. In particular, Fig. 4.13a shows the time per day during which the app gathers the user's location. It depicts the distribution of time tracked per day over user-app combinations in the three datasets. Each data point (or a user-app combination) corresponds to a single user utilizing a single app for the data collection period. As evident from the figure, the 90th percentile of the tracking time falls below 10 minutes per day which is comparable to a single app session (Section 3.5). A single app can identify only a small snapshot of the user's mobility during a single day. In Fig. 4.13b, we also show the portion of the total observed user's PoI a single app can identify over the course of the data collection period (a few weeks to few months). It is evident a single app cannot accumulate more than 40% of the user's total number of PoIs even over an extended period. A service provider cannot trace even a part of the user's daily path using a location-aware foreground app. This implies that the model mobility of this chapter is inapplicable for foreground apps accessing the user's location sporadically. As we indicated earlier, our models of Chapters II and III ignore the order information between the visits and their timings. One could explore amending the ordering and timing information to a location-access model outdoor case in future work. Opposed to earlier research, such a model must consider tracking threats as well as profiling and identification threats.

We still have to handle the issue of the apps accessing user's location while running in the background. In Chapter II, we address this issue by passing them coarsened location samples. In Chapter III, we claim that those apps constitute a tiny minority of the user's installed apps. Recently, there has been a rise of a newer class of AI-based digital assistants that continuously run in the background to provide location-based service; Google Now is one example of such apps. These apps require accurate location access to provide users with services based on the locations they visit. As such, they resemble the case of an indoor service provider with comprehensive access to an individual's mobility inside an area. We can then apply the mobility model of this chapter to model the location access (and the privacy threats) of apps running in the background; this could be a venue for future

research.

4.11 Conclusion

In this chapter, we have designed, implemented, and evaluated PR−LBS, a system that balances between privacy and rewards in an indoor environment. It creates a win–win situation for both the users and service providers. PR−LBS packs in two mechanisms for private location release in indoor environments as well as a novel privacy criterion to estimate the cost of sharing location. It subjects the user's mobility to a location–service exchange that is based on the repeated play model to ensure the users are rewarded for sharing some aspects of their mobility. Our evaluations show that PR−LBS is easy to deploy, has low energy overhead, is usable, effectively remedies the user's concerns, and does not affect the SP's utility. In future, we would like to conduct a field study of PR−LBS's device-based prototype. We also want to explore options to providing APIs for the SPs to access aggregate information privately without releasing any raw location data.

Appendix A

In the following, the user traverses a path of a length m in an area containing $|Z|$ zones. PR−LBS releases a path, $path_{obs}$, to the service provider.

D.P. Mechanism Variant Privacy Guarantees

For the mechanism to achieve (ϵ, d_m) differential privacy, it must satisfy:

$$P(path_{obs}|path) \leq e^{\epsilon} P(path_{obs}|path') \tag{4.9}$$

such that $d(path, path') \leq d_m$ and $P(path_{obs}|path_{tr})$ is the probability of observing

$path_{obs}$ given the user traversed $path_{tr}$. $P(path_{obs}|path)$ is given by:

$$P(path_{obs}|path) = q^{m-d}(1-q)^d \frac{1}{(|Z|-1)^d} \tag{4.10}$$

such that $d = d(path_{obs}|path)$ is the edit distance between the two paths.

Then, we have:

$$\frac{P(path_{obs}|path)}{P(path_{obs}|path')} \leq e^{\epsilon} \tag{4.11}$$

$$\frac{q^{m-d}(1-q)^d \frac{1}{(|Z|-1)^d}}{q^{m-d'}(1-q)^{d'} \frac{1}{(|Z|-1)^{d'}}} \leq e^{\epsilon} \tag{4.12}$$

$$\left(\frac{1-q}{q(|Z|-1)}\right)^{d-d'} \leq e^{\epsilon} \tag{4.13}$$

When $d(path, path') \leq d_m$, then $d - d' \leq d_m$, because $d > 0$ and $d' > 0$ then $d(path_{obs}, path) - (path_{obs}, path') < |d(path_{obs}, path) - (path_{obs}, path')| < d(path, path')$ by using the reverse triangle inequality for the metric spaces (the edit distance is a metric). Then we have:

$$\left(\frac{1-q}{q(|Z|-1)}\right)^{d-d'} \leq \left(\frac{1-q}{q(|Z|-1)}\right)^{d_m} \leq e^{\epsilon} \tag{4.14}$$

Finally, this mechanism will achieve (ϵ, d_m) differential privacy for:

$$q \leq \frac{e^{\epsilon/d_m}}{|Z| - 1 + e^{\epsilon/d_m}} \tag{4.15}$$

D.P. Mechanism Privacy Guarantees

After releasing m zones, the D.P. mechanism satisfies the expression in Eq. (4.9) for any value of d_m. The probability of observing a path given some other traversed path is given by:

$$P(path_{obs}|path) = \prod (q_i/|Z|_i)^{n_i} \tag{4.16}$$

Where $|Z|_i$ represents the number of zones at a distance i from user's zones and $q_i/|Z|_i$ the probability to choose a zone from those at a distance i from the visited zone. n_i represents the number of released zones that are a distance i from the actual zones. For two paths at a distance d_m from each other, we need to satisfy the following:

$$\frac{\prod (q_i/|Z|_i)^{n_i}}{\prod (q_j/|Z|_j)^{n_j}} \leq e^{\epsilon};\ i,j \leq d_m \tag{4.17}$$

The expression of Eq. (4.17) will assume its maximum value when the observed path is the user's actual path and $path'$ is a path at a distance d_m. In such a case we have (for a path of length m):

$$\frac{q_0^{d_m}}{(\alpha^{d_m} q_0/|Z|_{d_m})^{d_m}} \leq e^{\epsilon} \tag{4.18}$$

Where $|Z|_{d_m}$ is the maximum number of zones at a distance of d_m from any of the zones of the released path.

We can then derive an lower bound for α and upper bound for q_0 as:

$$\alpha \geq \frac{|Z|_{d_m}}{e^{\epsilon/m}};\ q_0 \leq \frac{1}{\sum \alpha^i}$$

A.S. Mechanism Anonymity Set

The size of the anonymity set is a random variable, S, depends on the length of the released path. Let R be a random variable that represents the length of the released path and k represent the possible value of the traversed path length such that $k \leq m$. Our objective is to compute the expected size of the anonymity set $E(S)$.

$$E(S) = \sum_{k=0}^{m} \sum_{r=0}^{k} E(S|R=r).P(R=r)$$

$$= \sum_{k=0}^{m} \sum_{r=0}^{k} \binom{k}{r} \binom{|Z|-r}{k-r} (k-r)! \binom{k}{r} q^r (1-q)^{k-r}$$

$$= \sum_{k=0}^{m} \sum_{r=0}^{k} \binom{k}{r}^2 q^r (1-q)^{k-r} \frac{(|Z|-r)!}{(|Z|-k)!}$$

Per-Path Information Disclosure

When the user moves to a new zone z_l, PR−LBS estimates the information leak if the SP is to observe that the user visited z_l, with total observed path being $pa_l = \langle pa_{l-1}, z_l \rangle$.

Let $visit(z)$ be the event that the SP observed user visited the zone z. $P(pa_l|visit(z_l))$, then, refers to the probability distribution of the user visiting the current path pa_l of length l after observing the visit to zone z_l. By definition, a new observation will necessarily change $P(p_l)$ (the SP's existing belief about the user's mobility); the amount of change in this PDF is what we refer to as the information leak. The amount of leaked information can be defined as:

$$lk(p_l, z) = \sup_{p_l \in P_l} \frac{P(p_l|visit(z)) - P(p_l)}{P(p_l)}. \tag{4.19}$$

Since we focus on the positive information disclosure, the only path that will experience a positive improvement in the amount of information is pa_l, the path the user is currently visiting. This reduces Eq. (4.19) to:

$$lk(pa_l, z) = \frac{P(pa_l|visit(z)) - P(pa_l)}{P(pa_l)}. \tag{4.20}$$

If the user has visited the area N times (number of sessions), out of which s/he traversed the path pa_l for $n[pa_l]$ times, then $P(pa_l) = \frac{n[pa_l]}{N}$. When observing a new visit, the probability will be $P(pa_l|visit(z)) = \frac{n[pa_l]+1}{N+1}$. The information leak will then be:

$$lk(pa_l, z) = \frac{1-a}{a(N+1)},\ a = \frac{n[pa_l]}{N}.$$

Appendix B

Suppose the user visits an area comprised of four zones: "A", "B", "C", and "D". After 10 sessions, the user's mobility model is as follows:

1. **Paths of length=4:** P($A-B-C-D$)= $2/10$, P($A-C-B-D$)=$3/10$, P($A-D-B-A$)=$1/10$, P($B-D-C-B$)=$1/10$, P($B-D-C-A$)=$1/10$, and P(ϕ)=$2/10$.
2. **Paths of length=3:** P($A-B-C$)= $2/10$, P($A-C-B$)=$3/10$, P($A-D-B$)=$1/10$, P($B-D-C$)=$2/10$, and P(ϕ)=$2/10$.
3. **Paths of length=3:** P($A-B$)= $2/10$, P($A-C$)=$3/10$, P($A-D$)=$1/10$, P($B-D$)=$2/10$, and P(ϕ)=$2/10$.
4. **Paths of length=3:** P(A)= $6/10$, P(B)=$2/10$, and P(ϕ)=$2/10$.

During the 11^{th} session, the user traverses the path $B-D-C-A$.

1. First visited zone is B; the path will only comprise B at this point. At the beginning of the visit, the SP expects the user (based on previous mobility) to visit either A or B. Since the user visited B, there is an information leakage.
2. Second visited zone is D; the new path will be $B-D$. This path leaks some information because there are multiple expected paths of length 2. But the visited zone leaks no information according to our criterion. The only expected visited zone after B is D. The user conformed to the SP's expectations and leaked no information. It is worth noting that the information leak of the path thus far is equal to information leak from the first visited zone, which is B. The same applies for the third visited zone C.

Table 4.6: Privacy cost of visited path.

Zone (z)	**Path** (p_l)	$P(p_l)$	$P(p_l\|z)$	$lk(p_l, z)$	$leak(z)$
B	B	$2/10$	$3/11$	$log_2(\frac{15}{11})$	$log_2(\frac{15}{11})$
D	$B-D$	$2/10$	$3/11$	$log_2(\frac{15}{11})$	0
C	$B-D-C$	$2/10$	$3/11$	$log_2(\frac{15}{11})$	0
A	$B-D-C-A$	$1/10$	$2/11$	$log_2(\frac{20}{11})$	$log_2(\frac{4}{3})$

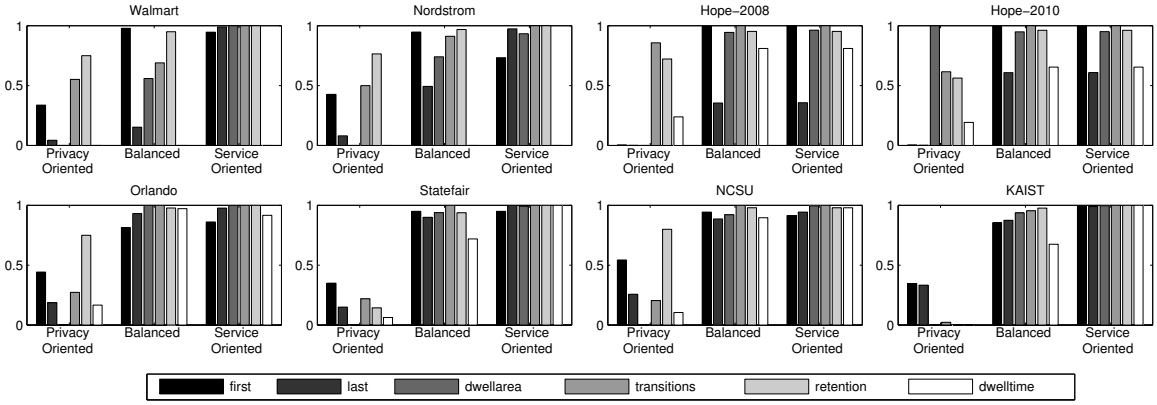


Figure 4.14: Utility metrics for high service level. Some metrics in Walmart and Nordstrom datasets are not available.

3. Last visited zone is A'; the path comprises $B-D-C-A$. The newly visited zone leaks some information since there are two possibilities after traversing $B-D-C$, either A or B. By visiting D, the user offered the SP new information that resulted in a shift of its belief about the user mobility.

Appendix C

Figs 4.14 and 4.15 show that PR−LBS adapts sharing to the service level. When the server is more rewarding, as in Fig. 4.14, PR−LBS shares more of the user's mobility. All the utility metrics for the neutral and service-oriented users are close to 1. While those for the privacy oriented users are lower so that PR−LBS protects their privacy. On the other hand, Fig. 4.15 shows the utility metrics for a low-rewarding server. It is evident that the utility metrics drop considerably when compared to the high-rewarding service provider.

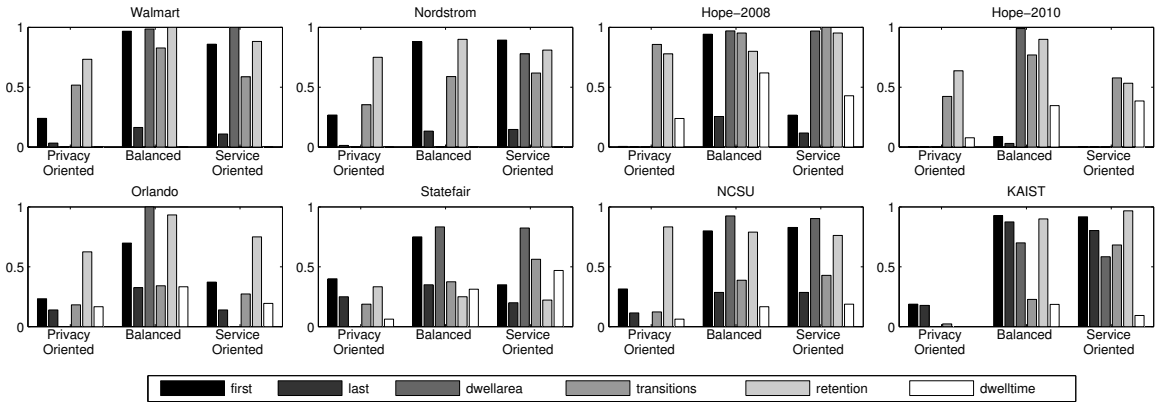


Figure 4.15: Utility metrics for low service level. Some metrics in Walmart and Nordstrom datasets are not available.

CHAPTER V

BLE-Guardian

5.1 Introduction

In this chapter, we consider the second source for location privacy threats as defined in Section 1.2.1. We particularly focus on the BLE protocol as the representative wireless technology for wearable devices.

Bluetooth Low Energy (BLE) [11] has emerged as the *de facto* communication protocol in the new computing paradigm of the Internet of Things (IoTs) [12, 13, 14, 15, 16, 17]. In 2013, over 1.2 billion BLE products were shipped [16], with this number expected to hit 2.7 billion in 2020 [141]. BLE-equipped products are embedded and used in every aspect of our lives; they sense nearby objects, track our fitness, control smart appliances and toys provide physical security, etc. The BLE protocol owes this proliferation to its low energy and small processing footprint as well as its support by most end-user devices [142], such as PCs, gateways, smartphones, and tablets.

A BLE-equipped device advertises its presence to let interested nearby devices initiate connections and glean relevant information. These advertisements, however, are a double-edged sword. An unauthorized, potentially malicious, party can use these advertisements to learn more about the BLE-equipped devices of a certain user or in a specific environment [21]. Enumerating the user's employed devices is generally referred to in literature as the *inventory attack* [22]. Revealing the device's presence is the stepping stone toward

more serious privacy and security attacks with grave consequences in the case of medical devices for example, especially for high-value targets [143].

The BLE specification contains some privacy provisions to minimize the effects of inventory attacks and ensuing threats, namely *address randomization* and *whitelisting*. A BLE device is supposed to randomize its address to prevent others from tracking it over time. Moreover, only devices with a pre-existing trust relationship (whitelisted devices) are supposed to access the BLE-equipped device.

In this chapter, we first analyze how existing BLE's privacy measures fare in the real-world deployments through our own data-collection campaign. To the best of our knowledge, this is the first study that systematically analyzes threats to the BLE-equipped devices in the wild. We recruited participants from our institution and the PhoneLab testbed [80] to collect the BLE advertisements in their vicinity. We have collected and analyzed the advertisements from 214 different types of BLE-equipped devices. Analyzing our dataset has led to a surprising discovery: BLE advertisements, due to poor design and/or implementation, leak an alarming amount of information that allows the tracking, profiling, and fingerprinting of the users. Furthermore, some devices allow external connections without an existing trust relationship. Unauthorized entities can access unsecured data on the BLE-equipped devices that might leak sensitive information and potentially inflict physical harm to the bearer.

Almost all of the existing approaches addressing some of the above threats rely on mechanisms that necessarily include changes to the protocol itself or to the way the BLE-equipped devices function [41, 42]. Changing the operation of such devices, post-production, requires their patching by securely pushing a firmware update. With thousands of manufacturers and developers around the world, it is very challenging, sometimes impossible, to guarantee firmware patches to the millions of already deployed devices [43]. Even a security-aware user might lack the ability to update the firmware of a BLE-equipped device. Patch management is, therefore, the leading security challenge in the emerging

IoTs [144, 145] (including BLE-equipped devices) for many reasons:

- Manufacturers might lack the ability to apply OTA updates [146] for some deployed BLE-equipped devices because they (such as a BLE-equipped pregnancy test) are neither programmable nor equipped with an Internet connection.
- Customers might neither receive news about the update nor be able to apply an update even if available. For example, a month after the 2013 "Foscam" webcams hacking incident, 40,000 of 46,000 vulnerable cameras were not updated although a firmware update was available [147].
- Companies do not have enough financial incentives or resources to maintain the devices post deployment [148]. For example, Samsung discontinued two lines of smart refrigerators after 2012 so that customers cannot receive updates for their purchased refrigerators [149].

There is, therefore, a need for a new class of practical approaches to mitigate the privacy threats to BLE-equipped devices. In this chapter, we seek to answer the following related question: *can we effectively fend off the threats to BLE-equipped devices: (1) in a device-agnostic manner, (2) using COTS (Commercial-Off-The-Shelf) hardware only, and (3) with as little user intervention as possible?*

We present `BLE-Guardian` as an answer to the above question. It is a practical system that protects the user's BLE-equipped devices so that *only* user-authorized entities can discover, scan, or connect to them. `BLE-Guardian` relies on an external and off-the-shelf Bluetooth radio as well as an accompanying application. Therefore, a user can easily install (and control) `BLE-Guardian` to any BLE gateway, be it a smartphone, tablet, PC, Raspberry PI, Artik-10, etc. The external radio achieves the physical protection, while the application, running on the gateway, enables the user to interact with `BLE-Guardian`.

`BLE-Guardian` provides privacy and security protection by targeting the root of the threats, namely the advertisements. In particular, `BLE-Guardian` opportunistically in-

vokes reactive jamming to determine the entities that can observe the device existence through the advertisements (*device hiding* module), and those that can issue connection requests in response to advertisements (*access control* module). In a typical BLE environment, however, achieving `BLE-Guardian`'s objective is rather challenging. Many BLE-equipped devices, including the ones to be protected, advertise on the same channel; while at the same time other devices, in response to advertisements, issue scan and connection requests. The timing is of an essence for `BLE-Guardian`; it invokes jamming at the right time for the right duration. Therefore, `BLE-Guardian` does not inadvertently harm other devices, preserves the ability of authorized entities to connect the BLE-equipped device, and always hides the BLE-equipped device when needed.

More than one device might be authorized to connect to the BLE-equipped device. `BLE-Guardian` differentiates the scan and connection requests originating from authorized devices versus those that are fraudulent. This is particularly challenging as the BLE advertisement channel lacks any authentication mechanism for the advertisements and connections. `BLE-Guardian` utilizes Bluetooth classic as an out-of-band (OOB) channel to authorize a device after obtaining the user's permission. It uses the OOB channel to instruct the connecting device to issue ordinary connection requests with (varying) special parameters that other unauthorized devices cannot predict. It also alerts the user when unauthorized parties attempt connection to the user's BLE devices.

`BLE-Guardian` achieves its objectives with minimum requirements from the external radio. Effectively, `BLE-Guardian` operates with a radio that offers only the basic capabilities of reception and transmission on the BLE channels. As a result, `BLE-Guardian` avoids employing sophisticated and customized (thus impractical) radios and signal processing approaches.

We implement `BLE-Guardian` using the commercially available Ubertooth One[1] USB dongle so that `BLE-Guardian` can be easily installed on any BLE gateway. We also

[1] https://greatscottgadgets.com/ubertoothone/

implement accompanying apps for different BLE gateways, such as Android and Raspberry PI. We evaluate `BLE-Guardian` using several BLE devices for different real-world scenarios, where we assess its effectiveness in combating privacy threats, its low overhead on the channel and devices, and little disruption to the operation of legitimate BLE devices. In particular, `BLE-Guardian` is able to protect up to 10 class-2 target BLE-equipped devices within a 5m range with less than 16% energy overhead on the gateway.

The rest chapter is organized as follows. Section 5.2 discusses the related work. Section 5.3 provides the necessary BLE background. Section 5.4 states the privacy threats arising from BLE advertisements through our data-collection campaign. Section 5.5 details the design of `BLE-Guardian`. Section 5.6 presents the implementation of `BLE-Guardian` and evaluates its effectiveness. Finally, the chapter concludes with Section 5.7.

5.2 Related Work

There have been limited efforts related to BLE devices that target the security and privacy threats resulting from the devices revealing their presence. The only exception is the work by Wang [42], where a privacy enhancement is proposed for BLE advertisements to ensure confidentiality and prevent replay attacks as well as tracking. This enhancement is based on providing an additional 3-way handshake between the peripheral and the gateway. Unarguably, this enhancement changes both the protocol and the peripheral which is highly impractical as we argued before.

Another related field of research includes wearable and body-area networks. The work by Leonard [41] uses a honeypot to lure in adversaries that attempt to attack the user's wearable devices. The honeypot uses a set of helper nodes to expose fake services with known weaknesses so that the attacker connects to them. This work, however, doesn't handle the privacy threat arising from BLE advertisements. A determined attacker will be able to distinguish fake traffic from legitimate one based on RF signatures from the devices and issue connections to the user's real devices.

Other relevant work includes approaches to protecting medical devices. Mare *et al.* [150] propose a mechanism that protects health sensors when communicating with a gateway. The proposed system, albeit relevant, doesn't apply for the BLE ecosystem. It also mandates changing the medical devices. Gollakota *et al.* [151] propose an external device, called *Shield*, that the user wears to control access to his/her embedded medical device. Shield implements friendly jamming so that only an authorized programmer can communicate with the medical device.

`BLE-Guardian` takes an entirely different approach by targeting the control plane of the BLE protocol instead of the data plane. `BLE-Guardian` does not need to continually protect an ongoing authorized connection and more importantly need not invoke jamming signal cancellation that requires accurate estimation of channel condition in a dynamic mobile indoor environment as well as a full duplex radio. `BLE-Guardian` constitutes a reference design that can function with any radio that has reception and transmission capabilities on the 2.4 GHz band. `BLE-Guardian`, also, considers far less restrictive scenarios than Shield. It does not have to be within centimeters of the device-to-be-protected as the case with Shield. Moreover, `BLE-Guardian`'s practical design allows scaling up protection for multiple devices (multiple connectors and protected devices) simultaneously, which is not the case for Shield that considers a two-device scenario only [152].

Finally, researchers have explored ways to reduce information leaks from sensors in a smart home environment [153, 154]. Srinivasan *et al.* [153], Park *et al.* [154], and Schurgot *et al.* [155] propose a set of privacy enhancements that include perturbing the timing of broadcasted sensory data along with padding the real sensory data with fake data to confuse the adversary. These protocols fail to address the threats resulting from BLE advertisements and have the shortcoming of requiring changes to the sensors as well.

Table 5.1: The four types of BLE advertisements.

Type	Advertising Interval
ADV_IND	20ms − 10.24s
ADV_DIRECT_IND	3.75ms
ADV_NONCONN_IND	100ms − 10.24s
ADV_SCAN_IND	100ms − 10.24s

5.3 BLE Primer

The BLE (Bluetooth 4.0 and newer) protocol has been developed by the Bluetooth SIG to support low power devices such as sensors, fitness trackers, health monitors, etc. Currently, more than 75,000 devices in the market support this protocol along with most of more capable devices such as smartphones, tablet, PCs, and recently access points [18].

5.3.1 BLE States

A BLE device assumes either a central or peripheral role. A peripheral device is typically the one with lower capabilities and with the information to advertise. The central device, typically an AP, PC, or smartphone, scans for advertisements and initiates connections.

The BLE specification places a higher burden on the central device. It is responsible for initiating the connection and thus has to keep scanning until it receives an advertisement. Conversely, the peripheral (prior to its connection) sleeps for most of the time and only wakes up to advertise, which helps save its limited energy.

5.3.2 Advertisements

BLE advertisements are instrumental to the operation of the protocol, and constitute the only means by which a device can be discovered. The specification defines 4 advertisement message types as shown in Table 5.1, and 3 advertisement channels: 37 (2402MHz), 38 (2426MHz), and 39 (2480MHz).

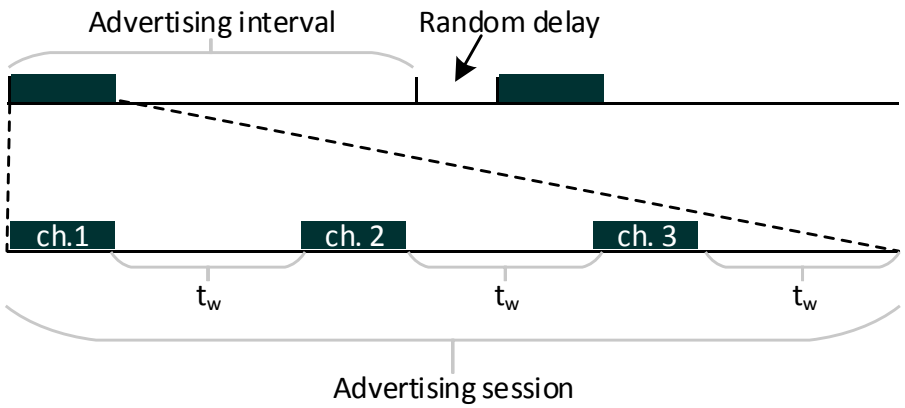


Figure 5.1: The advertisement pattern in BLE.

ADV_DIRECT_IND (introduced in Bluetooth 4.2) is a special advertisement; it enables fast reconnection between the central and the peripheral devices. The peripheral, when turned on, will broadcast advertisements at a fast rate (once every 3.75ms, for 1.28 seconds) that are directed to the client (with a pre-existing trust relationship) before assuming the central role. The advertisement message only contains the message type, source, and destination addresses.

The other three advertisements are similar to each other in that they are periodic. The advertisement interval determines the frequency with which each device advertises. This interval has to be chosen, at configuration time, between 20ms and 10.24 seconds (at increments of 0.625ms) for the ADV_IND advertisement and between 300ms and 10.24 seconds for the other two advertisements. To prevent advertisements of different devices from colliding with each other, each device waits for a random amount of time between 0 and 10ms (in addition to the advertisement interval) before it advertises (Fig. 5.1).

The advertisement session constitutes the period when the device is actually advertising. During each advertisement session, the device advertises on the three advertisement channels given a pre-configured channel sequence. Before switching to the next channel, the device has to wait for a preset period of time (less than 10ms) for scan and connection requests (t_w in Fig. 5.1). We will henceforth refer to the advertisement interval, the channel sequence, and the waiting time as the *advertisement pattern*.

Each advertisement message contains the message type, source address, along with some of the services offered by the device and their respective values. The specification defines a set of services that have unique UUIDs, such as device name. The message is limited in length, and hence, to get more information about the device, an interested device can either issue a scan request to which the advertising device responds with a scan response or connect to the advertising device.

5.3.3 Connections

Not all BLE devices accept connections; devices that use ADV_NONCONN_IND advertisement messages run in transmit mode only and they do not accept any scan or connection requests such as iBeacons.

Also, devices advertising with ADV_SCAN_IND messages do not accept connections but accept scan requests. Particularly, when the device broadcasts an advertisement message on some channel, it listens on the same channel for some time (less than 10ms) before switching to the next channel. It waits for scan requests from clients wanting to learn more information to which it responds with a scan response.

Devices that advertise using ADV_IND messages are scannable and connectable; they respond to scan messages and connection requests. After sending an advertisement message, the device listens for connection requests. The connection request contains the source and destination addresses along with other connection parameters. These parameters contain the connection interval, the timeout, and the slave interval. When connected, the device starts frequency hopping according to a schedule negotiated with the central. If the device (now peripheral) doesn't receive any communication from the central over the period defined by the "timeout interval", it drops the connection.

While connected, the device must not broadcast connectable advertisement messages (the first two types of Table 5.1). It can, however, still broadcast non-connectable advertising messages to share information (the last two types of Table 5.1) with other clients

in its transmission range which still leaks information about the device's name, type, and address.

5.3.4 Privacy and Security Provisions

The BLE specification borrows some security provisions from classical Bluetooth to establish trust relationships between devices, a process known as *pairing*. When the device boots for the first time, it will advertise using ADV_IND with its public Bluetooth address. The user can then pair a smartphone (or other BLE-equipped device) so that the two devices exchange a secret key that will enable future secure communication.

Once a BLE-equipped device is paired with another device, it can invoke more privacy and security provisions. The first provision is whitelisting, and the device will only accept connections from devices it has been paired with before, i.e., those that are whitelisted. Also, the device might accept connections from any client but might require higher security levels for some of the services it exposes so that only authorized users access sensitive content.

Finally, the BLE specification defines a privacy provision based on address randomization to prevent device tracking. When two devices are paired, they exchange an additional key called the *Identity Resolution Key* (IRK). The device uses this key to generate a random address according to a timer value set by the manufacturer, which it will use to advertise. This random address will be resolved by the paired device using the same key. As this random address is supposed to change regularly, curious parties shouldn't be able to track a BLE-equipped device. Devices that do not utilize address randomization can resort to direct advertising (ADV_DIRECT_IND) to enable fast and private reconnections.

These privacy provisions are akin to those proposed earlier in the context of WiFi networks. Researchers have long identified privacy leaks from the consistent identifiers broadcasted by wireless devices. They proposed privacy enhancements that include randomizing or frequent disposing of MAC addresses [156, 56] and hiding them through encrypting the

entire WiFi packets [157]. These enhancements require introducing changes for the client devices.

5.4 Threats from BLE Devices

While, in theory, BLE's privacy provisions might be effective to thwart threats to the user's privacy, whether or not various manufacturers and developers properly implement them is an entirely different story. In what follows, we investigate how the BLE privacy provisions fare in the wild using a dataset collected in our institution and using the PhoneLab testbed [80] of SUNY Buffalo.

We developed an Android app that collects the content of the advertisement messages. We recruited users from our institution and from the PhoneLab testbed. We didn't collect any personal information about the users and thus obtained non-regulated status from the IRB of our institution. One could view this study as crowdsourcing the analysis of BLE devices; instead of purchasing a limited set of devices and analyzing them, we monitored the behavior of a broad range of devices in the wild. Analyzing the advertisements we collected from 214 different types of devices (sample of these devices shown in Tables 5.2 and 5.3), we observed:

1. Two advertisement types (ADV_NONCONN_IND and ADV_SCAN_IND) require a fixed address which would enable tracking of a mobile target.
2. Devices that are bonded to the users advertise using ADV_IND messages instead of the more private ADV_DIRECT_IND.
3. Some devices, albeit not expected to do so, use their public Bluetooth addresses in advertisements. This also enables tracking as well as identifying of the device manufacturer.
4. Other devices implement poor address randomization by flipping some bits of the address rendering them ineffective against tracking. This has also been identified in

Table 5.2: A sample of devices with revealing names.

Name	Type
LG LAS751M(27:5D)	music streaming
JS00002074	digital pen
ihere	key finder
spacestation	battery/storage extension
Jabra PULSE Smart	smartbulb
DEXCOMRX	Glucose monitor
Clover Printer 0467	printer
Frances's Band ea:9d LE	smartband
Gear Fit (60ED)	activity tracker
Lyve Home-00228	photo storage
Matthias-FUSE	headset
Richelle's Band b2:6a LE	smartband
vivosmart #3891203273	activity tracker
KFDNX	key fob
OTbeat	heart rate monitor
Thermos-4653	Smart Thermos
POWERDRIVER-L10C3	smart power inverter

other studies of BLE devices [21].

5. A large number of devices advertise their names (Table 5.2), revealing sensitive information about them, the user, and the environment. Also, some of the device names contain personal information or unique identifiers that may enable user tracking.

6. Some devices use a consistent Bluetooth address for long periods of time which renders address randomization ineffective (Table 5.3). Examples include various types of wristbands (Fitbit Flex, Forerunner 920, etc.), headsets, smartwatches (Apple Watch or Samsung Gear), etc. This has also been identified by Das *et al.* [158], where they analyzed the advertisements of BLE-equipped fitness trackers. Das *et al.* found the fitness trackers constantly advertising with consistent (non-private) BLE addresses. In our experiments, we observed that Flex and One kept the same address for 37 days, so did One and Charge for 30 days.

7. Some devices accept connections from non-bonded devices. This allows access to the services on the device including the unique manufacturer ID, for instance, which allows for user tracking regardless of the device's address. For example, we were

Table 5.3: A sample of devices with consistent addresses for more than a day.

Name	Type	Days observed
One	activity tracker	37
Flex	activity tracker	37
Zip	activity tracker	37
Surge	activity tracker	36
Charge	activity tracker	36
Forerunner 920	smartwatch	36
Basis Peak	sleep tracker	25
MB Chronowing	smartwatch	15
dotti	pixel light	7
UP MOVE	fitness tracker	2
GKChain	laptop security	2
Gear S2 (0412)	smartwatch	2
Crazyflie	quadropter	1
Dropcam	camera	1

able to connect to various devices and access data from them without any existing trust relationship, such as various Fitbit devices (One, Zip Flex, Charge), Garmin Vivosmart, digital pens, etc. It is worth noting that we connected to these devices under controlled experimental settings, not in the wild. As a result, an external access control mechanism is necessary to protect such devices.

The above observations indicate that the address randomization, part of the BLE specifications, fails to provide the promised privacy protection. Various developers and manufacturers do not implement it properly; they rely on public Bluetooth addresses, apply weak randomization, or keep a consistent address for a long time. On the other hand, even if address randomization is properly implemented, other information in the advertisement or in the device might contain unique information (device name or id) that allows for its tracking.

Moreover, data accessed from an advertisement or the device (once connected) might reveal sensitive information about the user or the environment. Through our data collection campaign, we were able to detect different types of glucose monitors, wristbands, smart watches, fitness trackers, sleep monitors, laptops, smartphones, laptop security locks, security cameras, key trackers, headsets, etc. Knowing which type of glucose monitor the user

(a) Mobile

(b) Vehicle

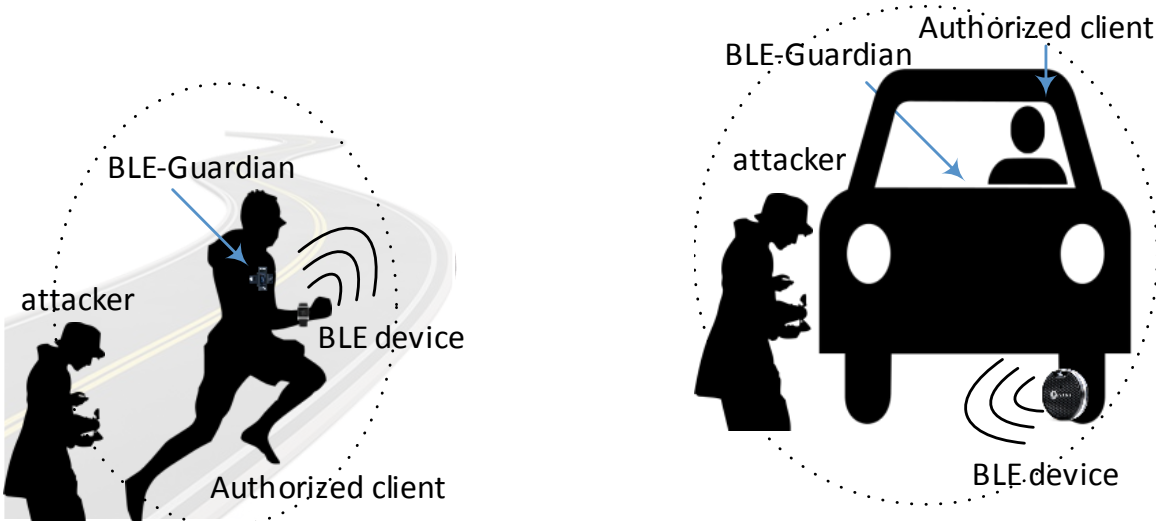


Figure 5.2: Example deployments of BLE-Guardian.

is carrying or the type of physical security system s/he has installed could lead to serious harm to the user. Finally, an adversary might use such advertisement data as side information to infer more about the user's behavior. For example, a temperature sensor constantly reading $60\,^{\circ}\mathrm{F}$ in winter would indicate a vacant property [159] which may invite in a thief.

5.5 BLE-Guardian

BLE-Guardian addresses the aforementioned privacy threats by allowing only *authorized clients* to discover, scan, and connect to the user's BLE-equipped device. Before delving into the inner workings of BLE-Guardian, we first describe the system and threat models.

5.5.1 System and Threat Models

5.5.1.1 System Model

A typical BLE scenario involves two interacting entities: the client and the BLE-equipped device. The BLE device broadcasts advertisements to make other nearby clients aware of its presence along with the services/information it is offering. A client can then connect to the device to access more services/information and control some of its attributes, in which case it will be the BLE-device's gateway to the outside world.

The user's mobile device (e.g., smartphone or tablet) acts a gateway where BLE devices are wearable (e.g., fitness trackers), or mHealth devices (e.g., Glucose monitor) (Fig. 5.2a). In a home environment, the user's access point, PC, or even mobile device, serves as a gateway for BLE devices that include smart appliances, webcams, physical security systems, etc. Last but not least, a smart vehicle or the driver's mobile device operate as gateways (Fig. 5.2b) for the different BLE-equipped sensors in the vehicle, such as tire pressure.[2] An interested reader could consult the work of Rouf *et al.* [160] for a discussion on the security and privacy risks of a wireless tire pressure sensor.

`BLE-Guardian` leverages the existence of gateways near the BLE-equipped devices to fend off unauthorized clients scanning and connecting to them. It comprises both hardware and software components. The hardware component is an external Bluetooth radio that connects physically to the gateway, while the software component is an accompanying application that runs on the gateway. `BLE-Guardian` requires another software component to run on the clients willing to discover and connect to the user's BLE devices. The user, be it an owner of the BLE-equipped device or a client, interacts with `BLE-Guardian` through its software components.

5.5.1.2 Threat Model

`BLE-Guardian` only trusts the gateway on which it is running. Otherwise, the entire

[2]https://my-fobo.com/Product/FOBOTIRE

operation of BLE-Guardian will be compromised and will fail to provide the promised privacy provisions. BLE-Guardian achieves privacy protection at the device level, so that if it authorizes a client to access the BLE device, all applications running on that device will have same access privileges. This security/privacy dimension is orthogonal to BLE-Guardian and has been addressed elsewhere [161]. It also requires the user's BLE device — the one to be protected — to comply with the BLE specifications.

BLE-Guardian protects a target BLE-equipped device against an adversary or an unauthorized/unwanted device that sniffs the device's advertisements, issues scan requests and attempts to connect to the device. The adversary aims to achieve three objectives based on the BLE devices that the user is deploying:

1. **Profiling:** The adversary aims to obtain an inventory of the user's devices. Based on this inventory, the adversary might learn the user's health condition, preferences, habits, etc.

2. **Tracking:** The adversary aims to monitor the user's devices to track him/her over time, especially by exploiting the consistent identifiers that the devices leak as we observed in Section 5.4.

3. **Harming:** The adversary aims to access the user's BLE device to learn more sensitive information or even to control it. Both will have severe consequences for the user, especially if a certain device is known to have some vulnerabilities that allow remote unauthorized access [162].

This adversary can have varying passive and active capabilities, from curious individuals scanning nearby devices (e.g., using a mobile app), to those with moderate technical knowledge employing commercial sniffers, all the way to sophisticated adversaries with software-defined radios.

A *passive* attacker is capable of sniffing all the communications over advertisement channels including those that fail the CRC check. This includes all commercial Blue-

tooth devices and existing Bluetooth sniffers in the market, such as the Texas Instruments CC2540 chip. The adversary might possess further capabilities by employing MIMO receiver that could recover the original signal from the jammed signal [163], especially in stationary scenarios. We refer to this adversary as the *strong passive* attacker.

Furthermore, the adversary is capable of injecting traffic into any Bluetooth channel at any given point of time, but will "play" within the bounds of the BLE specifications when attempting communication with the BLE device. This is a reasonable assumption, as the device will not otherwise respond to any communication. We refer to such an adversary as the *active* attacker. On the other hand, the attacker might be able to transmit with higher power than allowed by regulatory bodies, in which case we refer to as the *strong active* attacker.

Thus, we have four classes of attackers referring to the combinations of their passive and active capabilities as shown in the first column of Table 5.4.

Attacks, including jamming the channel completely, masquerading as fake devices to trick the users to connect to them, or attacking the bonding process are orthogonal to our work. Such attacks have been addressed by O'Connor and Reeves [164] and Ryan [165]. Finally, once `BLE-Guardian` enables the authorized client to connect to the BLE device, it will not have any control over what follows later.

5.5.2 High-Level Overview

`BLE-Guardian` is a system the user can use out of the box; it only requires installing a hardware component (an external Bluetooth radio) to the gateway and running an app on the gateway to control and interface with the Bluetooth radio. Conceptually, `BLE-Guardian` consists of *device hiding* and *access control* modules. The device hiding module ensures that the BLE device is invisible to scanners in the area, while the access control module ensures that only authorized clients are allowed to discover, scan, and connect to the BLE device.

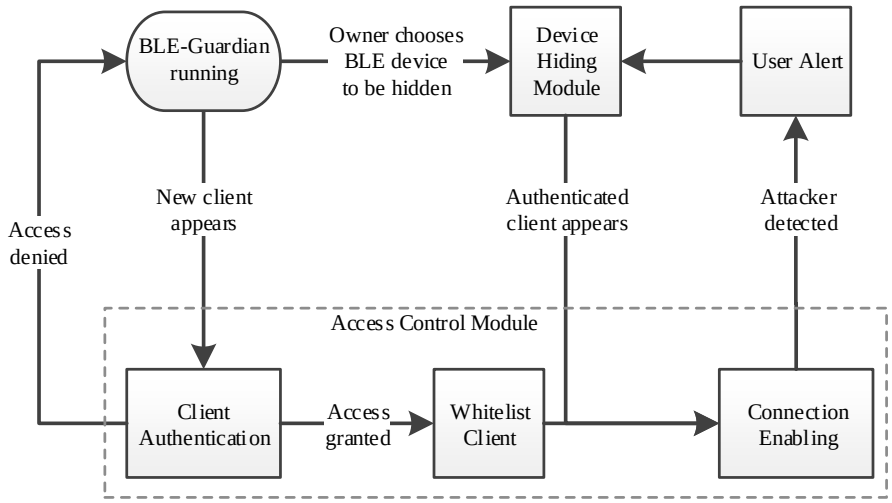


Figure 5.3: The modules of `BLE-Guardian` and their underlying interactions.

Fig. 5.3 shows the high-level operation of `BLE-Guardian` from the moment a user designates a BLE device to be protected all the way to enabling authorized client connection to the protected device. The high-level operation of `BLE-Guardian` takes the following sequence:

1. The user installs the hardware component along with the accompanying app on the gateway.
2. The user runs the app, which scans for BLE devices nearby. The user can then choose a device to hide.
3. The device hiding module of `BLE-Guardian` starts by learning the advertisement pattern of the target BLE device along with that of the other devices in the area. The device hiding module then applies reactive jamming to hide the device.
4. When a new client enters the area and wants to discover the user's devices, it communicates with the access control module so that the user can either grant or reject authorization.
5. If the user authorizes the client, the access control module advertises *privately* on behalf of the BLE device to let the authorized client scan and connect to it.

6. BLE-Guardian monitors if other unauthorized entities are attempting to connect to the BLE device; in such a case, it blocks the connection and alerts the user.

5.5.3 Device Hiding

The hiding module is responsible for rendering the BLE device invisible to other scanning devices. The hiding module jams the device's advertisement session to achieve this invisibility. In particular, it targets three types of advertisements, ADV_IND, ADV_NONCONN_IND, and ADV_SCAN_IND of Table 5.1, which are periodic and leak more information about the user as we indicated earlier.

Hiding the BLE device is, however, challenging for two reasons. The hiding module must jam the BLE device precisely at the moment it is advertising. Also, it must not disrupt the operation of other devices advertising in the same area.

5.5.3.1 Learning

The hiding module first learns the target BLE device's advertising pattern before jamming to hide its presence. The device's advertisement pattern comprises the advertising interval, advertising channel sequence, and the time to listen on the individual channels. Fortunately, the latter two parameters are deterministic and can be observed directly, which is not the case for the advertising interval. The BLE specification leaves it to the device to determine the advertising pattern, so that there are 15 possible permutations of the channel sequence.

As shown in Fig. 5.4, BLE-Guardian follows a process of elimination to identify the advertising sequence of the BLE device using a single antenna. In the worst case, it will take three advertising intervals to learn the entire advertising sequence of a BLE-equipped device. This corresponds to the longest path of Fig. 5.4, where BLE-Guardian monitors each channel for the maximum advertising interval of 10.24 seconds. At the same time, it would have identified the time the BLE device spends listening on each channel before

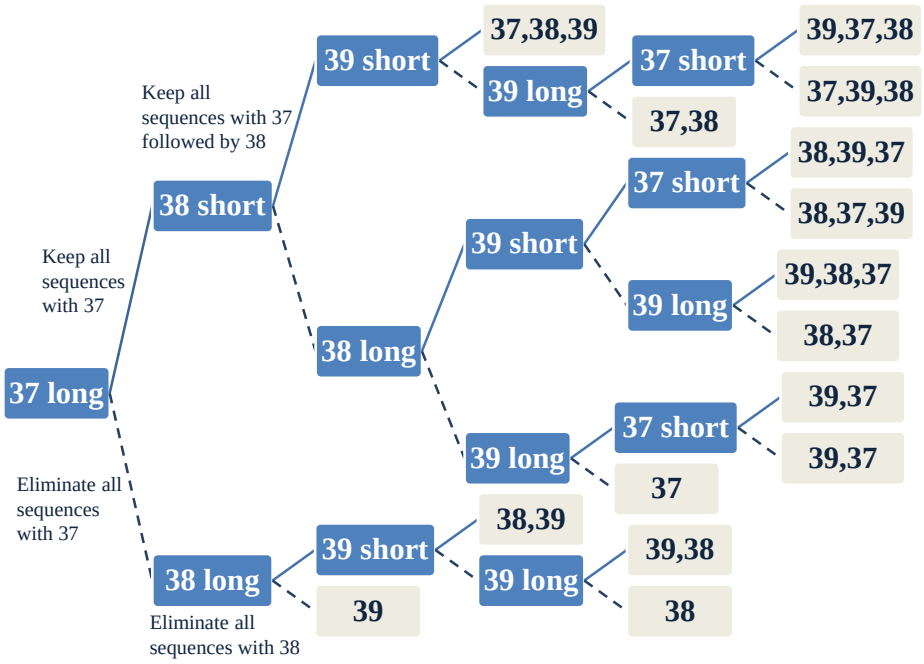


Figure 5.4: The learning algorithm followed by `BLE-Guardian`. The blue boxes refer to monitoring each channel either for a short period of time (less than 10ms) or for a longer period of 10.24 seconds. Depending on whether an advertisement is detected on the channel some sequences are eliminated till a sequence is decided on (gray boxes).

switching to the next channel.

While observing the advertising sequence of the BLE device, the hiding module keeps track of the interval separating the consecutive advertisements sessions. The hiding module observes a set of inter-advertisement intervals, $t_i = adv + p$, where adv is the actual advertisement interval as set by the device and p is a random variable representing the random delay such that $p \in unif(0, 10\,ms)$. Also, `BLE-Guardian` will perform the same process for all advertising devices in the same area at the same time to learn their advertising parameters as well. Learning other devices' advertising at the same time will be useful as evident below.

5.5.3.2 Actuation

After identifying the advertising pattern, the hiding module needs to just detect the start of the advertisement session. Then, it jams the advertising channels according to their

advertisement sequence. There is a caveat, though; the hiding module needs to detect the advertisement before it can be decoded. Otherwise, the rest of the jamming will not be effective.

From monitoring earlier advertisements, the hiding module obtains a set of t_i's of different devices' advertisements, including the BLE device to be hidden. The advertisement interval will be $adv = t_i - p$ for each observed inter-advertisement interval. Each observed advertisement will be used to improve the estimation of the advertisement interval. For N observed intervals, we have:

$$adv = \frac{1}{N}\sum_{i=1}^{N}(t_i - p) = \frac{1}{N}\sum_{i=1}^{N} t_i - \frac{1}{N}\sum_{i=1}^{N} p. \tag{5.1}$$

Let $P = \frac{1}{N}\sum_{i=1}^{N} p$, the random variable P is drawn from the distribution $\frac{1}{N}p * \frac{1}{N}p * \frac{1}{N}p \ldots \frac{1}{N}p$. Since the single random delays p are i.i.d., the mean of P will be equal to 5 (mean of the original distribution of p) and the standard deviation of $\sqrt{\sum_{i=1}^{N} \sigma_p}$ = $\frac{5}{N\sqrt{(3)}}$. The hiding module estimates adv as:

$$adv' = E(adv) = \frac{1}{N}\sum_{i=1}^{N} t_i - 5. \tag{5.2}$$

The standard deviation of P will get lower as N increases; it defines the error in the estimate of adv as defined by Eq. (5.1). Given previous N observed advertisements from the BLE device, the hiding module can predict the next advertisement to happen at $adv_{next} \in [adv_{low},\ adv_{high}]$ such that:

$$adv_{low} = T_N + adv' - e \tag{5.3}$$

$$adv_{high} = T_N + adv' + e + 10, \tag{5.4}$$

where T_N is the time of the last advertisement and e is the 90^{th} percentile value of P

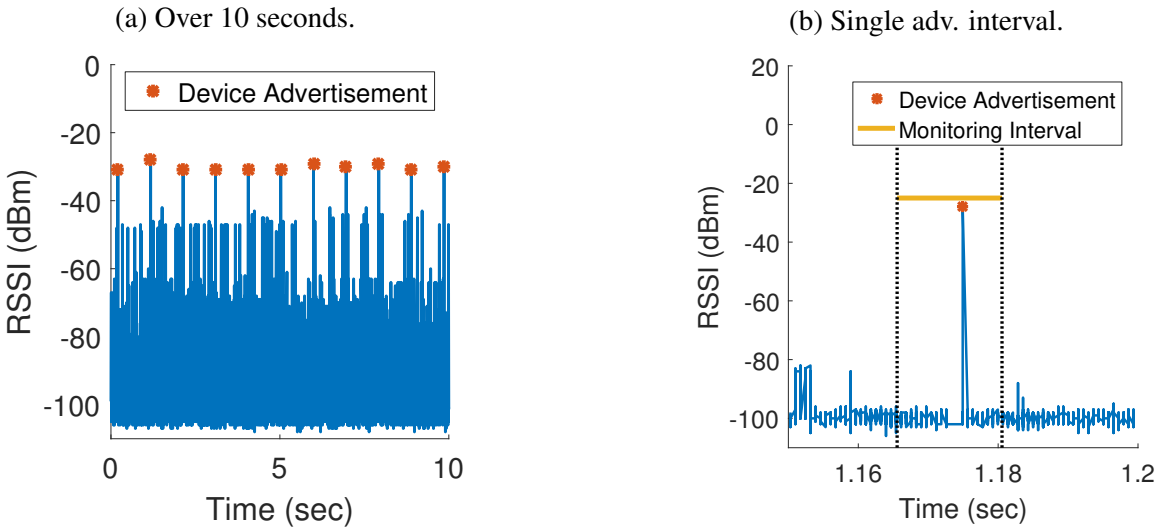


Figure 5.5: RSSI at channel 37 when a device is advertising at a distance of 1m at the interval of 960ms.

(symmetric around the mean) which approaches 0 as N increases (so that $adv_{high} - adv_{low}$ approaches $10ms$).

Starting from the last observed T_N of the target BLE device, the advertisement hiding module computes adv_{low} and adv_{high}. Also, it enumerates the list of other devices expected to advertise within the interval $[adv_{low},\ adv_{high}]$.

The device hiding module always listens on the first channel of the advertising sequence of the BLE device to be hidden. During the interval $[adv_{low},\ adv_{high}]$, the device hiding module will sample the RSSI of the channel very frequently (every 25μs). When the received RSSI is $-90dBm$ or higher (the peaks of Fig. 5.5a), `BLE-Guardian` determines that there is a transmission currently starting to take place. The device hiding module moves immediately to jam the channel on which it is listening. Since the transmission of a typical advertisement message takes 380μs to finish [166], jamming the channel will prevent the message from being decoded by other receivers.

At this point, two situations might arise; (1) the target BLE device is the only device expected to be advertising at this time instant, or (2) some other device is expected to be advertising in the same interval. In the first situation, the target BLE device is most

probably responsible for this transmission as part of its advertisement session. The device hiding module repeats the same process (sample RSSI and jam) over the rest of the channels to confirm that transmissions follow the device's advertising pattern. Fig. 5.5b shows an example interval where there is only one device advertising.

In the second situation, the device hiding module cannot readily ascertain whether the transmission belongs to the target BLE device or not. This will take place when the observed transmission sequence matches the advertising sequence of the target BLE device and some other device that is expected to advertise at the same interval. To resolve this uncertainty, immediately after jamming the advertising message (400μs after commencing jamming on the channel), the device hiding module lifts jamming and sends scan requests for devices other than the target device. The device hiding module then listens on the channel to observe if a scan response is received. Despite its advertisement being jammed, any device will still be listening on and will respond to scan requests. Depending on whether a scan response is received or not, `BLE-Guardian` can associate the transmission with the correct device, be it the target BLE device or some other device.

The device hiding module then adjusts the next monitoring interval according to the observed transmissions in the current intervals as follows:

$$adv_{low} = min(T_N) + adv' - e \tag{5.5}$$

$$adv_{high} = max(t_N) + adv' + e + 10, \tag{5.6}$$

where T_N represents the instants of the transmissions possibly matching the advertisement of the target BLE device in the current monitoring interval.

Note that we do not utilize the power level per se, or any physical-layer indicator, to indicate whether the same device is transmitting or not, as it is sensitive to the environment and the distance between `BLE-Guardian` and the target BLE device. To actually perform the jamming, the device hiding module continuously transmits at the maximum power for

the specified interval.

BLE-Guardian may jam the advertisements of non-target devices which might disrupt their operation, which we referred to as the second situation above. Nevertheless, because of the random delay introduced by the device before each advertisement, the aforementioned "collision" events become unlikely. In Appendix 5.7.1, we use renewal theory to show that the expected number of another device's advertisements within the expected advertising interval of the target BLE-equipped device will always be less than 1. This applies when BLE-Guardian protects a single BLE-equipped device. Our evaluation in Section 5.6 confirms this observation.

5.5.4 Access control

So far, BLE-Guardian has hidden the target BLE device, so neither authorized nor unauthorized entities have access to the device. It is the access control module that authorizes client devices and enables their access to the target BLE device.

5.5.4.1 Device Authorization

BLE-Guardian utilizes Bluetooth classic (BR/EDR) as an out-of-band (OOB) channel to authorize end-user devices intending to scan and access the target BLE device. BLE-Guardian runs in server mode on the gateway waiting for incoming connections, while the "authenticating" device will have BLE-Guardian running in client mode to initiate connections and ask for authorization. The choice of Bluetooth Classic as an OOB channel is natural; most end-user devices (such as smartphones) are dual-mode, supporting both BLE and Bluetooth classic. Moreover, Bluetooth classic already contains pairing and authentication procedures, eliminating the need for a dedicated authentication protocol. Last but not least, a Bluetooth-equipped end-user device will be able to communicate simultaneously over Bluetooth classic and BLE so that it can communicate with both BLE-Guardian and the target BLE device.

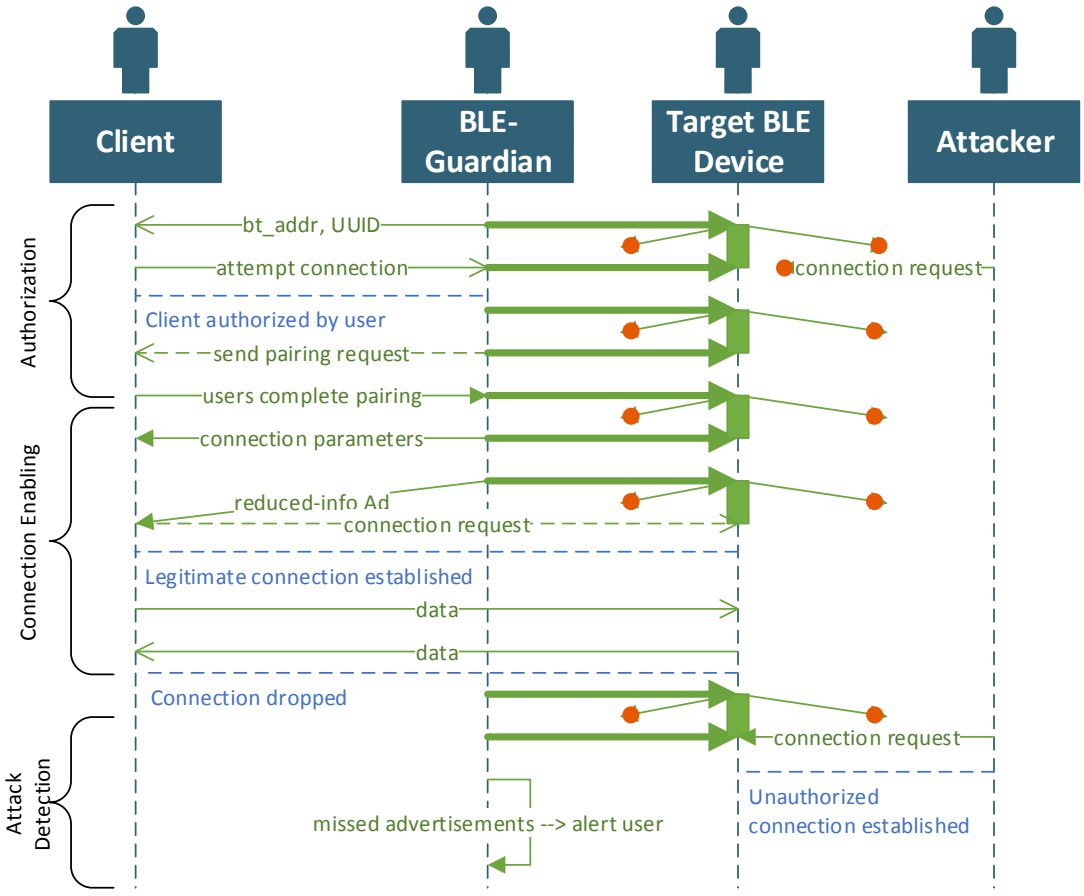


Figure 5.6: The sequence diagram of the access control module. Thin green lines from the target device designate the advertisements. Thick green lines from `BLE-Guardian` designate the jamming signal.

Fig. 5.6 depicts the interactions between `BLE-Guardian` and a client device when they connect for the first time. `BLE-Guardian` will be listening on the gateway over a secure RFCOMM channel with a specified UUID. The gateway, however, will not be running in discoverable mode so as to prevent others from tracking the user. It is up to the party interested in authenticating itself to obtain the Bluetooth address of the user's gateway as well as the UUID of the authentication service.

Once the client end-user device obtains the Bluetooth address and UUID, it can initiate a secure connection to the gateway. This will trigger a pairing process to take place if both devices are not already paired. `BLE-Guardian` relies on Bluetooth pairing process

to secure the connections between the gateway and the client device. For future sessions, an already paired client device can connect to BLE-Guardian without the need for any user involvement. The owner can also revoke the privileges of any client device by simply un-pairing it.

5.5.4.2 Connection Enabling

The device hiding module of BLE-Guardian jams the entire advertising sequence of the target BLE device, including the period it listens for incoming scan or connection requests so that it cannot decode them. Therefore, both unauthorized and authorized clients cannot access the target BLE device (the case of an adversary using high transmission power will be discussed later). Fig. 5.6 shows the procedure that BLE-Guardian follows to enable *only* the authorized clients access to the target BLE device.

Immediately after the last advertisement of a single advertisement session, when the target device is the only one expected to be advertising, the access control module lifts the jamming. This ensures that the BLE device will not be advertising until the next adverting session, and it is currently listening for scan and connection requests. Then, BLE-Guardian advertises on behalf of the target BLE device on the same channel. The advertisement message contains only the headers and the address of the previously hidden device. It is stripped of explicit identifiers, hence leaking only limited information about the BLE device for a brief period.

At the same time, BLE-Guardian will communicate to the authenticated client app the address of the BLE device and a *secret* set of connection parameters over the OOB channel. BLE-Guardian's app running on the client device will use the address and the parameters to initiate a connection to the BLE device. The connection initiation procedure is handled by the Bluetooth radio of the client device, which scans for the advertisement with the provided address. After receiving such an advertisement, it sends a connection request after which both devices will be connected.

The above procedure will not break the way BLE scans and connections take place. It doesn't matter from which radio the actual advertisement was coming. From the perspective of the BLE device, it will receive a scan or connection request while waiting for one. On the other hand, the client device will receive an advertisement message while also expecting one.

5.5.5 Security and Privacy Features

`BLE-Guardian` addresses the tracking and profiling threats discussed in Section 5.5.1.2. It hides the advertisements, which are used as the main means to track users. It only exposes the advertisement for a very short period when enabling others to connect. Furthermore, `BLE-Guardian` greatly reduces the profiling threat by hiding the contents of the advertisement which leak the device name, type, and other attributes.

A strong passive attacker [163] can still detect the "hidden" peripheral by recovering the real advertisement, so that it can connect to the BLE-equipped device. Distinguishing legitimate connection requests based on the Bluetooth address of the initiator is not effective; an attacker could easily spoof its Bluetooth address to impersonate the authorized client. Therefore, `BLE-Guardian` uses the connection parameters of the connection request to distinguish fraudulent connection requests from legitimate ones. Legitimate connection requests contain the set of "secret" connection parameters communicated earlier to the client.

The probability of the attacker matching a particular set of connection parameters is very low. According to the specification, there are more than 3 million possible combinations of values for the connection, slave, and timeout intervals. If the connection is established based on a fraudulent connection request, then `BLE-Guardian` prevents the connection from taking place. The connection request already contains the hopping sequence initiation. `BLE-Guardian` hops to the next channel and jams it so as to prevent the BLE device from receiving any message from the connected unauthorized device. The BLE device drops the connection since it receives no message on the channel.

Table 5.4: The protections offered by BLE-Guardian.

Adversary	Profiling Protection	Tracking Protection	Access Control	User Alert
Passive & Active	✔	✔	✔	✔
Strong Passive & Active	–	✔	✔	✔
Passive & Strong Active	✔	✔	–	✔
Strong Passive & Strong Active	–	✔	–	✔

An attacker might abuse this connection process by constantly attempting to connect to the BLE device, thus depriving the authorized client of access. This will always be possible, even when BLE-Guardian is not deployed. BLE-Guardian observes such a situation from a high frequency of fraudulent connection requests and alerts the user of this threat. As it will be evident in Section 5.6, an active attacker injecting messages to the advertising channel cannot affect the operations of BLE-Guardian.

A strong active adversary, however, can override BLE-Guardian's jamming and issue connection requests that the BLE-equipped device will decode. While jamming, BLE-Guardian runs in transmit mode and can not monitor the channel for incoming requests. Nevertheless, it detects that the BLE device is missing its advertising intervals, signifying that it was connected without BLE-Guardian's approval. In such a case, BLE-Guardian alerts the user of the existence of a strong adversary nearby.

Finally, Table 5.4 summarizes BLE-Guardian's capabilities when faced with the various adversaries described in Section 5.5.1.2.

5.6 Implementation and Evaluation

We now present a prototype of BLE-Guardian along with its evaluation.

5.6.1 Implementation

We implement BLE-Guardian using Ubertooth One radio which is an open platform for Bluetooth research and development. It can connect to any host that supports USB

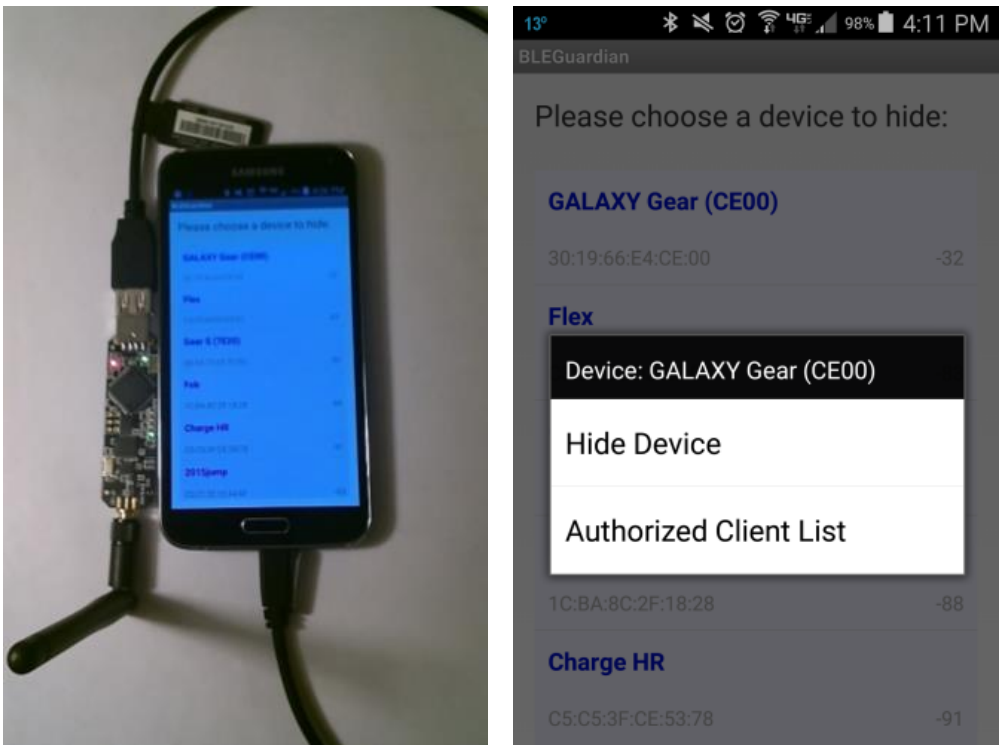


Figure 5.7: The deployment scenario for `BLE-Guardian` for a mobile user (left) and the main UI (right).

such as Raspberry Pi, Samsung's Artik-10, PC, smartphone (Fig. 5.7 (left)), etc. Since communication over USB incurs latency in the order of a few milliseconds, we implement most of `BLE-Guardian`'s functionalities inside Ubertooth One's firmware to maintain real-time operation.

We also implement the software component of `BLE-Guardian` on Linux and Android. Fig. 5.7 (right) shows a screenshot of the `BLE-Guardian` app while running on Android in server mode where the user can choose the device to protect and control its authorized client list. The app communicates the Bluetooth address of the chosen device to the Ubertooth One radio.

`BLE-Guardian` requires running in privileged mode on the client device in order to be able to connect with modified connection parameters. This is easily achievable on Linux-based clients, but might not be the case for mobile devices. In other words, `BLE-Guardian`, while running in client mode on Android, requires root access to be able

to issue connection requests with a set of specified connection requests. Also, BLE-Guardian (if running in privileged mode on the client device) can modify content of the advertisement message from the BLE scanner to the user-level application to reconstruct the original hidden advertisement. As such, user-level applications (on the trusted client) will receive the original advertisement, which does not break their functionality.

Maintaining BLE-Guardian is easy; it only requires updating the application running on the gateway which usually takes place without the user's intervention (e.g., mobile app updates). This application interacts with the hardware component and applies updates, if necessary, through pushing firmware updates, a process which is also transparent to the users.

5.6.2 Evaluation

To evaluate BLE-Guardian, we utilize Broadcom BCM20702A0 and Nordic nRF51822 chips as the target BLE devices (both transmitting at 4dBm) and the TI CC2540 dongle as the sniffer node. CC2540 is able to decode the messages on the three advertisement channels, even on those that fail the CRC check. We evaluate using Nordic and Broadcom chips instead of actual BLE products, because these products (such as Fitbits) are mostly powered by the same (Nordic and Broadcom) BLE chips.

5.6.2.1 Impact of Distance

Due to transmission power limitations (battery or regulatory bodies' constraints), there would always be a small area around the target BLE device where BLE-Guardian will not be able to enact the privacy protection. The transmission from the target BLE device covers the jamming signal of BLE-Guardian. Nevertheless, as the sniffer moves farther away from the target BLE device (in any direction), the jamming signal will cover the advertisements, provided that the BLE device and BLE-Guardian are not very far apart. So, there is a cutoff distance beyond which the adversary cannot scan, and connect to the

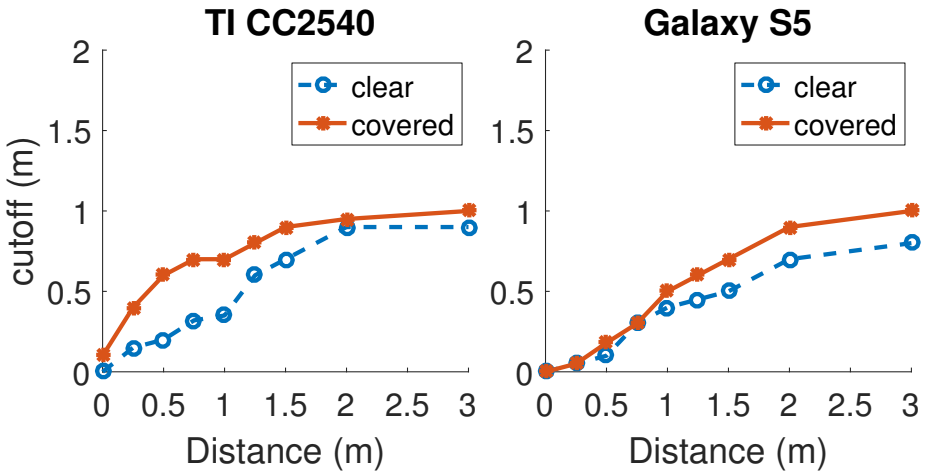


Figure 5.8: The cutoff distance as a function of the distance between `BLE-Guardian` and the target device.

target BLE device.

We study the cutoff distance of a target BLE device (advertising at 20ms) at different distances separating it from `BLE-Guardian` (between 0 and 3m). At each position, we move the sniffer node (either a CC2540 dongle or Samsung Galaxy S5) around the BLE device, and record the farthest distance at which it received any advertisement from the BLE device as to study the hidden terminal effect. Furthermore, we repeat each experiment twice, the first with `BLE-Guardian` clear of any obstacles and the second with it inside a backpack.

It is evident from Fig. 5.8 that the cutoff distance increases as `BLE-Guardian` and the BLE device become farther apart. In all of the cases, however, the cutoff distance is less than 1m, even when `BLE-Guardian` and the BLE device are 3m apart. This also applies when `BLE-Guardian` is inside the backpack which should reduce the effectiveness of its jamming. Sniffing with a smartphone has a shorter cutoff distance because the smartphone's BLE chip filters out advertisements failing the CRC check so that they are not reported to the OS.

The cutoff distance is enough to thwart tracking and profiling in several scenarios, especially when the user is moving (walking, jogging, biking or driving). In these scenarios,

BLE-Guardian is not farther than 1m from the target BLE device. An adversary has to get very close to the user, even if BLE-Guardian is covered in a coat or bag, to be able to scan or connect to the BLE device.

In other cases, the user has to keep his/her BLE devices (to be protected) close to BLE-Guardian in order to get the best privacy protection possible. Our experiments showed that BLE-Guardian and the target BLE device must be separated by a maximum distance of 5m so that an attacker beyond the cutoff distance will not be able to decode the advertisements. If BLE-Guardian and target BLE device are farther apart than this, then BLE-Guardian's jamming will not be able to cover the entire transmission area of the BLE device. In all circumstances, however, BLE-Guardian detects unauthorized connections and alerts the user accordingly.

5.6.2.2 Evaluation Setup

Beyond the cutoff distance, BLE-Guardian is capable of hiding the advertisements and controlling access to any target BLE device regardless of its advertising frequency. This protection, however, comes at a cost. In what follows, we evaluate BLE-Guardian's impact on other innocuous devices, the advertising channel, and the gateway. In the evaluation scenarios, we deploy the target BLE devices at distance of 1.5m from BLE-Guardian, and the sniffer between BLE-Guardian and the BLE devices (at a distance of 0.5m from BLE-Guardian). We evaluate BLE-Guardian when protecting up to 10 target devices with the following advertising intervals: 10.24 sec (highest possible), 5 sec, 2.5 sec, 1.25 sec, 960ms, 625ms, 312.5ms, 100ms, 40ms, and 20ms (lowest possible). Note that evaluating with 10 target devices constitutes an extreme scenario; according to our dataset, the average user is bonded to less than 4 devices, which would indicate the number of target devices (i.e. those to be protected).

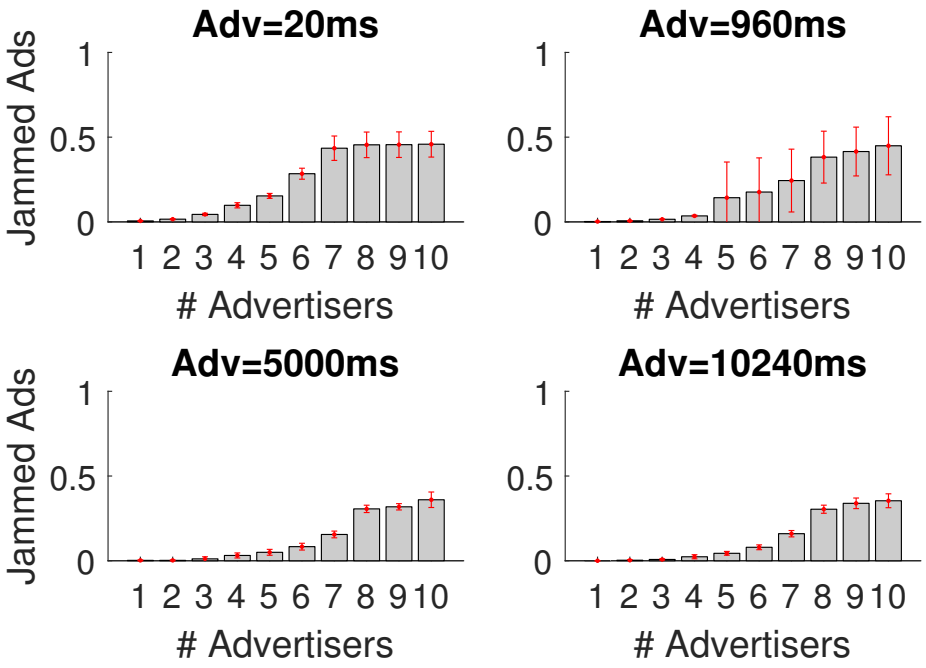


Figure 5.9: Portion of jammed advertisements of an innocuous BLE device when `BLE-Guardian` is running and protecting up to 10 advertisers.

5.6.2.3 Advertisement Hiding

Impact on Other Devices: We first evaluate the number of advertisements, not belonging to the target BLE device(s), `BLE-Guardian` will jam (Fig. 5.9). While accidentally jamming other devices doesn't affect the privacy properties of `BLE-Guardian`, it hinders the services they offer to other users. In particular, we study four scenarios with an innocuous (not the target) BLE device advertising at 20ms, 960ms, 5s, and 10.24s, and a varying number (between 1 and 10) of target devices, which `BLE-Guardian` protects. Each subset of target devices of size N (≤ 10) contains the top N advertising intervals from the list of Section 5.6.2.2.

There are two takeaways from Fig. 5.9. First, `BLE-Guardian` has little effect on other devices when it protects a relatively low number of devices, or when the advertising interval of the target BLE device(s) is larger than 500ms; in these cases, `BLE-Guardian` will be less active (bars corresponding to less than 6 target devices in the four plots of Fig. 5.9). Second, `BLE-Guardian` has a higher effect on the innocuous device with

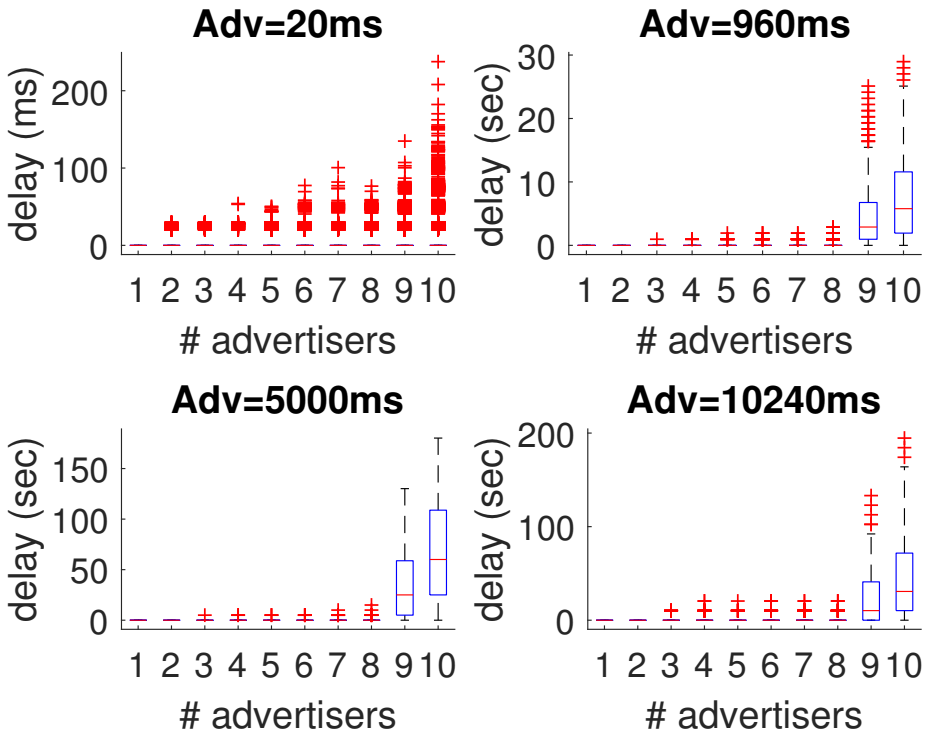


Figure 5.10: The delay of an authorized client in successfully connecting to the target device when `BLE-Guardian` is running.

higher advertising frequencies as observed from top-left plot of Fig. 5.9, especially when protecting a large number of devices (including those with 20 ms advertising interval).

In the latter case, `BLE-Guardian` is active for at least half of the time, representing the worst-case scenario of `BLE-Guardian`'s overhead where up to 50% of other devices' advertisements are jammed. However, since the advertisement frequency is high, even with a relatively high rate of jammed advertisements, the user's experience will not be drastically affected. On the other hand, when the target BLE device advertises at lower frequencies, the effect on the advertising channels and consequently other devices will be limited as evident from the rest of the plots of Fig. 5.9.

Impact on Authorized Access: To enable authorized connections, `BLE-Guardian` advertises on the behalf of the target BLE device only when it is confident that the target device is listening for connections. `BLE-Guardian` skips some advertising sessions which will introduce delays to authorized clients attempting connections as reported in Fig. 5.10.

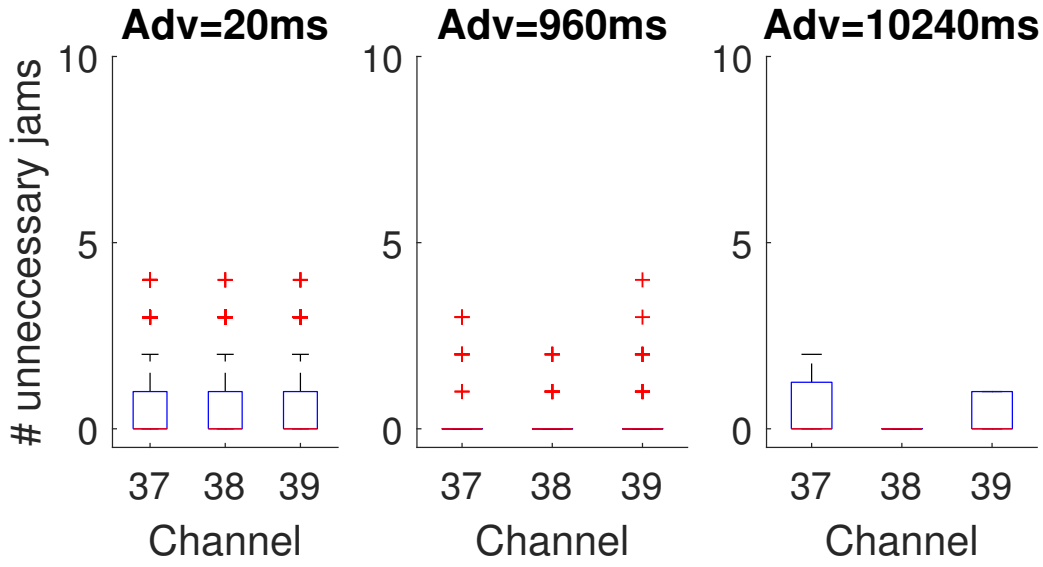


Figure 5.11: Unnecessary jamming instances with two advertisers at 20ms.

In this scenario, an authorized client is attempting connection to a target device advertising at 20ms, 960 ms, 5s, and 10.24 s, with an additional number of protected devices varying from 1 to 10. In the majority of the cases, the client has to wait for less than a second before successfully receiving an advertisement and issuing a connection. The only exception is the worst case consisting of `BLE-Guardian` protecting all of the 10 target devices (including devices advertising at intervals less than 100ms). The client might have to wait for up to multiple advertisement intervals before being able to connect. This is evident from the rightmost bar in each of the four plots of Fig. 5.10.

Impact on Advertising Channels Last but not least, we evaluate `BLE-Guardian`'s impact on the advertising channel, which, if high, might leak information about the existence of sensitive device(s). In this experiment, `BLE-Guardian` protects a single target device advertising at 20ms (the lowest possible), 960ms, and 10240ms (the highest possible). At the same time, two innocuous devices advertise at 20ms, in addition to other 15 devices not under our control advertising at different frequencies (minimum advertisement interval 30ms). In this scenario, `BLE-Guardian` will be active all the time since the two innocuous advertisers will force it to enlarge its monitoring interval between 20–30ms (while the advertising interval of the target device is only 20ms).

Fig. 5.11 shows the distribution of the number of unnecessary jammed instances in each interval when the target BLE device is expected to advertise. It is evident that in more than 50% of the intervals when BLE-Guardian is active, the number of unnecessary jamming instances events is 0, indicating a low overhead on the channel. When the target BLE device advertises at a lower frequency, BLE-Guardian is less active (middle and left plots of Fig. 5.11). These plots match the real-world scenarios observed from our data-collection campaign. Most commercial BLE devices advertise at relatively low frequencies (at intervals between 1 and 10s).

Finally, we evaluate the accuracy of predicting the next advertisement event of the target BLE devices. In all the experiments (including all scenarios), BLE-Guardian can predict the device's advertisements, i.e., the target BLE device advertised in the interval it is expected to. BLE-Guardian is also able to jam all the advertisements of the BLE device over the three advertising channels. This indicates that an attacker cannot modify the behavior of BLE-Guardian by injecting traffic into the advertising channels.

5.6.2.4 Energy Overhead

BLE-Guardian incurs no energy overhead for both the target BLE devices and the authorized clients. Nevertheless, energy overhead is a concern when BLE-Guardian is attached to a smartphone. We measured the energy overhead of BLE-Guardian using a Monsoon power monitor while running on a Samsung Galaxy S4 with Android 4.4.2. In the idle case, BLE-Guardian consumes 1370mW on average. The average power consumption rises to 1860mW while transmitting and 1654mW while receiving as shown in Fig. 5.12a. Fortunately, BLE-Guardian doesn't sense the channel or perform jamming frequently. Fig. 5.12b shows the average energy overhead when BLE-Guardian is protecting the set of ten devices (we described earlier) at different advertising intervals. In the worst case of 10 target BLE devices, including a couple advertising at the highest frequency possible, the energy overhead is limited to 16% regardless of whether there are

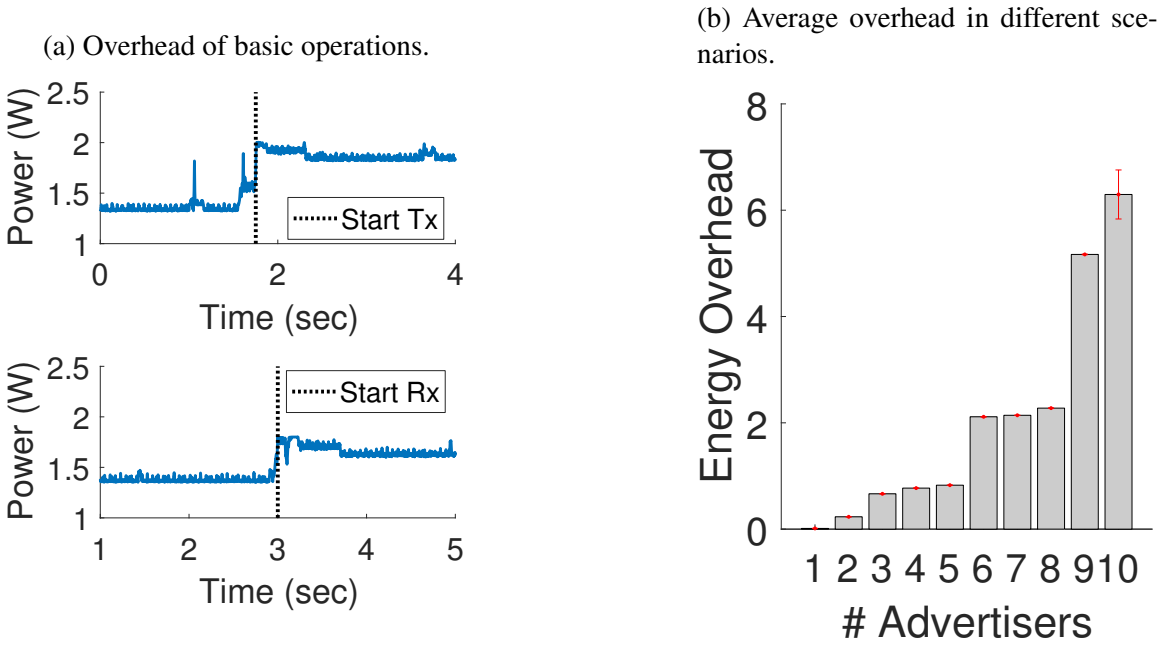


Figure 5.12: The energy overhead of `BLE-Guardian` running on Samsung Galaxy S4.

other advertisers in the area. In other cases, when there are less target devices and/or target devices are advertising at a lower frequency, the energy overhead is negligible.

5.7 Conclusion

BLE is emerging as the most prominent and promising communication protocol between different IoT devices. It, however, accompanies a set of privacy risks. An adversary can track, profile, and even harm the user through BLE-equipped devices that constantly advertise their presence. Existing solutions are impractical as they require modifications to the BLE-equipped devices, thereby making their deployment difficult. In this chapter, we presented a device-agnostic system, called `BLE-Guardian`, which addresses the users' privacy risks brought by BLE-equipped devices. `BLE-Guardian` doesn't require any modification to the protocol and can be implemented with off-the-shelf Bluetooth hardware. We implemented `BLE-Guardian` using Ubertooth One radio and Android, and evaluated its effectiveness in protecting the users' privacy. In future, we plan to explore

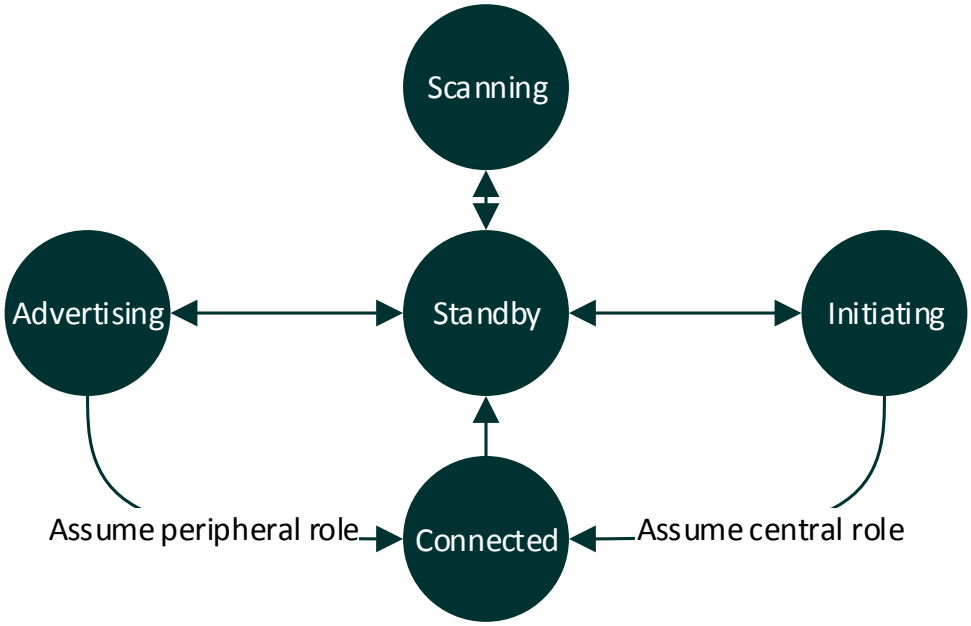


Figure 5.13: BLE states

the data plane by analyzing and reducing the data leaks from BLE devices to unauthorized clients. We further plan to extend `BLE-Guardian` to other wireless technologies tailored for IoT devices such as Zigbee.

Appendix A: A More Detailed BLE Primer

The BLE (Bluetooth 4.0 and newer) protocol has been developed by the Bluetooth SIG to support low power devices such as sensors, fitness trackers, health monitors, etc. Currently, more than 75,000 devices in the market support this protocol along with most of more capable devices such as smartphones, tablet, PCs, and recently access points [18].

5.7.1 BLE States

A BLE device assumes a central or peripheral role. A peripheral device is typically the one with lower capabilities and with the information to advertise. The central device, typically an AP, PC, or smartphone, scans for advertisements and initiates connections. Fig. 5.13 shows the states (and transitions between them) of a BLE device during its life-

time. A device can be in the standby, scanning, advertising, initiating, or connection state, depending on its role.

- **Standby state:** is the low power state of the device with both the transmitter and the receiver switched off. The BLE device wakes up to advertise or scan and then connect, depending on its role.
- **Advertising:** The device with information to offer, typically the one with fewer capabilities, will periodically broadcast advertisements to make other devices aware of its presence.
- **Scanning:** The device, typically with more capabilities, looking for information or services will run in the scanning state to listen for advertisements from other devices. If the device wants more information on others nearby, it issues scan requests in response to the advertisement message.
- **Initiating:** To connect to the advertising device, the initiating device sends a connection request message after receiving an advertisement.
- **Connection:** When the advertising device responds to the connection request from the initiating device, both will move to the connection state where they exchange messages at a negotiated time schedule and frequency-hopping sequence.

The BLE specification places a higher burden on the central device. It is responsible for initiating the connection and thus has to keep scanning until it receives an advertisement. Conversely, the peripheral (prior to its connection) sleeps for most of the time and only wakes up to advertise, which helps save its limited energy.

Appendix B: Fingerprinting Threat

The device analyzer dataset from Cambridge University [167] includes detailed smartphone usage statistics from more than 23K Android users. We had access to the data of

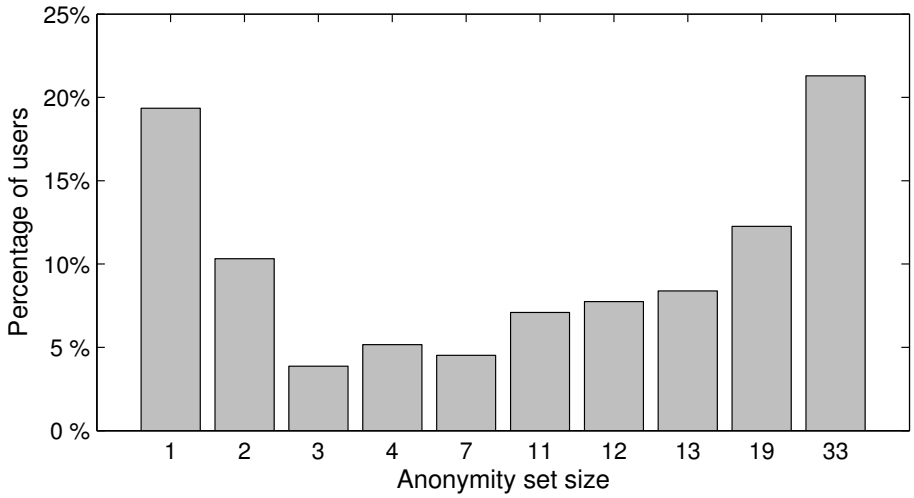


Figure 5.14: Percentage of users having a certain anonymity set size.

700 users spanning at least six months. It contains Bluetooth scan records; each record comprises the hashed Bluetooth address, the device's type (part of the advertisement), and whether the device is bonded to the user.

We analyzed these Bluetooth scan records to study the fingerprinting potential of Bluetooth devices about the user. We assume that if a user is bonded to a Bluetooth device, then s/he owns the device. We focused on the device types which are part of the advertisements and do not change when address randomization is employed. The question we aim to answer is whether the combination of the types of devices the user owns, regardless of any other information, is enough to identify the user. Device types include robots, peripherals, phones, PCs, wearables, cameras, toys, etc.

For each user, we enumerated the combination of device types (such as robots, peripherals, phones, PCs, wearables, cameras, toys, etc) s/he owns based on the scanned Bluetooth devices that s/he is bonded to. We then enumerated the number of users who share the same combination of device types. We found that around 20% of the users are uniquely identifiable just by the device type. Also, 45% of the users have an anonymity set of size less than 10.

This indicates that the types of devices users carry and interact with are enough to

identify them, thus acting as their "quasi-identifiers". They enable user tracking and fingerprinting regardless of whether the Bluetooth address is randomized or not.

Appendix C: Analysis of Device Hiding

`BLE-Guardian` may jam the advertisements of non-target devices which might disrupt their operation, which we refer to as the second situation in Section 5.5.3.2. Nevertheless, because of the random delay introduced by the device before each advertisement, the aforementioned "collision" events become unlikely. In what follows, we show that the expected number of another device's advertisements within the expected advertising interval of the target BLE-equipped device will always be less than 1, when `BLE-Guardian` protects a single BLE-equipped device.

One could view the advertising process of a single BLE-equipped device as a renewal process [168], where each event corresponds to an advertising session. The inter-arrival times, X_i, are nothing but the inter-advertising intervals defined as i.i.d. random variables such that $X_i \sim unif(adv, adv + 10)$. The n^{th} advertisement time $T_n = \sum_{i=1}^{n} X_i$ has the distribution defined by the n-fold convolution of distribution of X_i. As n increases, the probability distribution of the n-th advertisement spreads over a larger time interval defined as $A = [n.adv,\ n.(adv + 10)]$.

The device hiding module attempts jamming at an interval of width 10ms, as specified before. If this jamming interval falls within the expected advertising interval of some other device, A, then the second situation of Section 5.5.3.2 might occur. Nevertheless, as n increases the length of interval A increases and thus the expected number of advertisements, from a single device within 10ms should be less than 1. We show below how the expected number of advertisements in a 10ms interval drops between $n = 1$ and $n = 2$. We consider $m(t)$, the expected number of events up to time t, defined as $F_X(t) + \int_0^t m(t-x) f_X(x)\, dx$, where $f_S(s)$ is the probability distribution of X_i which is equal to $unif(adv, adv+10)$ and

$F_X(t)$ is the cumulative distribution function given as:

$$F_X(t) = \begin{cases} 0 & t < adv \\ \frac{t-adv}{10} & adv \leq t \leq adv + 10 \\ 1 & t > adv. \end{cases} \tag{5.7}$$

During the first advertising interval, $t \in [adv, adv + 10]$, the expected number of advertisements is $\frac{t-adv}{10} + \int_{adv}^{t} m(t - x)\, dx$ after substituting $F_X(t)$ and $f_X(x)$ with their corresponding expressions. After performing a substitution of variable of $y = t - x$, and since $m(t) = 0$ for $t < adv$, then $m(t) = \frac{t-adv}{10}$. So, if the expected advertising interval of the device hiding module overlaps with the first advertising interval of another advertising device, the expected number of events, $m(adv+10)-m(adv)$, will be 1, which is intuitive.

The second advertisement will take place at the interval $B = [2.adv,\ 2.(adv+10)]$, and we use a similar procedure to derive the expressions for $m(t)$ for $t \in [2.adv,\ 2.adv+10]$ and $t \in [2.adv,\ 2.(adv + 10)]$. If the expected advertising interval of the device hiding module overlaps with interval, B, then the expected number of advertisements $m(t + 10) - m(t)$ will drop to $\frac{1}{2}$. The same trend will follow for the subsequent advertising intervals; the expected number of another device's advertisements within the expected advertising of the target BLE device will always be less than 1. Our evaluation in Section 5.6. confirms this observation.

Finally, even if another device, with the same advertising parameters, starts advertising with the target BLE device at the same time, their advertising events will eventually diverge. After N advertisements from both devices, the distribution of $Ta_{N+1} - Tb_{N+1}$, the difference in time between the $N + 1$ advertising instants of both devices will be a random variable with mean 0 but with $\sigma = 2.N.\frac{5}{\sqrt{3}}$. As N increases, the standard deviation increases, which in turn decreases the probability of both advertising events taking place within 10ms. The 10ms-advertising interval is the length of interval that the device hiding

module expects the target BLE device to advertise.

CHAPTER VI

Conclusions

In this thesis, we study the location privacy threats in our electronic age. We focus on three representative scenarios: location-aware smartphone apps, indoor location-based services, and wearable devices in the context of the Internet of Things. In each of these scenarios, we highlight the shortcomings of state of the art in location privacy protection and propose four systems to address those deficiencies. We summarize the contributions of this thesis and present some future research directions.

6.1 Thesis Contributions

This thesis presents four systems, `LP-Guardian`, `LP-Doctor`, `PR-LBS` and `BLE-Guardian` that provide practical, theoretically sound, and usable location privacy protection.

Privacy Guarantees

All four proposed systems deliver privacy guarantees to thwart the tracking, profiling, and fingerprinting threats. `LP-Guardian` ensures location indistinguishability at both the low and high levels. At the low level, it limits apps' capability from continuously recording the user's mobility. Both `LP-Guardian` and `LP-Doctor` ensure that an adversary cannot identify the user's location within a specified area, eliminating the profiling threats. On the higher level, both guarantee the user's mobility pattern is indistinguishable from a

general set of individuals, thus thwarting the fingerprinting threat. PR-LBS employs differential privacy guarantees to prevent an indoor service provider from continuously tracking and profiling the shoppers. Last but not least, BLE-Guardian uses jamming to completely prevent an eavesdropper from tracking and profiling the bearer of BLE-equipped devices.

Practicality

The privacy guarantees that the proposed systems of this thesis provide do not come at the cost of practicality. LP-Guardian requires only modifying the Android's frameworks which can be achieved through rooting the Android device but requires no modification to the apps or APIs. LP-Doctor further improves the practicality of LP-Guardian by running completely in the user-level, without any further modifications. PR-LBS is compatible with the different modes of indoor localization. It acts as a broker between the localization engine and the service provider in the case of infrastructure-based localization. Also, it is fully compatible with the BLE-based localization in indoor environments through a user-level solution. BLE-Guardian protects the privacy of BLE users without introducing any modifications to the BLE-equipped devices or gateways, through the user of commercial-of-the-shelf hardware only.

Usability

Finally, the four proposed systems are user-centric as usability is a core design dimension mind. LP-Guardian and especially LP-Doctor minimize the frequency of prompting the users for privacy-related decisions. PR-LBS goes one step further by defining privacy profiles that allow the users to specify their privacy–utility trade-off at the bootstrapping stage. PR-LBS makes privacy-related decisions on behalf of the users and requires no run-time interactions from them. Similarly, using BLE-Guardian, the user has only to specify the BLE-equipped device to be hidden. Beyond that, BLE-Guardian

provides all the privacy protection with little burden on the users

6.2 Future Research Directions

Here, we discuss future research directions that can be built on top of this thesis.

6.2.1 Usable Privacy

On the shorter term, we will investigate the deployment challenges facing location privacy enhancing technologies. Proposed system might be effective, practical and usable, but will have little impact if users do not employ them on a daily basis. We plan to address these deployment challenges on two fronts. First, we will research mechanisms to increase the awareness of the privacy threats to regular users. Second, we will conduct field studies of some location privacy enhancing technologies to understand the psychological acceptability of these systems in real-world settings. In this study, we will also investigate whether a favorable privacy vs. utility trade-off be achieved in practice with different apps, users and scenarios.

On the longer term, a significant challenge related to the mass proliferation of interconnected devices is that of usable security and privacy. Traditional notice and choice mechanisms fail to protect users' privacy. Users are increasingly frustrated and overwhelmed with complex privacy policies, unreachable privacy settings, and a multitude of emerging standards. We will explore utilizing Conversational Privacy Bots (*PriBots*) [169] as a new way of delivering notice and choice through a two-way dialogue between the user and a chatbot. PriBots will improve on state-of-the-art by offering users a more intuitive interface to inquire about their privacy settings, thus allowing them to control their privacy. In addition to investigating the potential applications of PriBots, We will undertake the different challenges of the underlying system including the user interface (UI) and natural language processing (NLP) aspects. Pribots will be part of a more general direction related to automated privacy. We will explore mechanisms that make privacy decisions on the users'

behalf that meet their needs while reducing unnecessary user interactions.

6.2.2 Data Privacy beyond Location

In the growing computing paradigm of IoT, privacy threats will present unique challenges, mainly due to device and service provider heterogeneity as well as the lack of a unified privacy and security control design language and primitives. A vast and diverse set of manufacturers, developers, and service providers will dominate the IoT ecosystem, giving rise to different device architectures, communication technologies and paradigms, data types, and data sources and sinks.

Reducing the privacy threats from exposing specific user's data lies at the core of any privacy preserving mechanism. Quantifying the privacy risks in a particular setting is feasible because the data type and structure are well-defined. Device- and service provider-heterogeneity implies that we have to rethink the problem of privacy protection as the data source, sink, syntax, and semantics will vary from one device to the other. The current approaches to per-application privacy protection will not apply in for future device deployments.

Quantifying the privacy threats is only the first step toward building the privacy preserving mechanisms of the future. While privacy protection is a must, it should not come at the expense of the potential benefits of the IoT. Towards this, we plan to develop privacy threat models that could be either semantics-aware or semantics-agnostic to quantify the privacy threats from the ubiquitous data. These models will inform the design and implementation of online and practical privacy-preserving mechanisms that balance between user privacy and data utility. For example, security techniques such homomorphic encryption, privacy guarantees such differential and conditional privacy, information disclosure-based privacy metrics, and anonymization strategies could be employed and adapted to enable the design of such mechanisms.

6.2.3 Privacy through Service Provider Diversity

Device- and SP-heterogeneity presents an opportunity for privacy protection. We propose a novel concept of privacy, which is privacy through provider diversity. This approach relies on dispersing queries over multiple equivalent non-colluding service providers to minimize the amount of information each provider has for a single user. In this realm, we seek two orthogonal types of diversity: subquery diversity and inter-query diversity. In subquery diversity, the main query is partitioned into a set of subqueries, each subquery is evaluated at an independent service provider, and results are combined on the client side. While in inter-query diversity, our approach uses different providers across queries to minimize the amount of information a single provider has about a user.

In the current mobile ecosystem, it is usual to see multiple service providers competing in the same market. Different independent service providers offer similar services. For example, there are multiple navigation services for mobile devices (Google Maps, Bing Maps, MapQuest Mobile), localized search engines (Google, Bing, Yahoo), restaurant recommendation services (Google Places, Yelp, Urbanspoon), multiple online shopping services (Amazon, eBay), and news services (CNN, NYTimes). These providers offer an acceptable quality of service, so it is feasible to use them interchangeably. The primary motivation behind the idea of privacy through diversity comes from the concept of data minimization [170]. Data minimization is a good provider security practice because it limits the amount of information that can be leaked about the user in the case of a security breach. Our approach enforces this practice through query partitioning and dispersion among several non-colluding service providers.

Sub-query Diversity We exploit subquery diversity by exploiting the observation that a user's access to the service provider iss a join query, and that similar providers of the same service can be viewed as replicated databases. In distributed databases, a join query combines results from several tables based on a certain condition from each table, and the

final result is an intersection of the corresponding results. We adopt the same model in the mobile setting; we first identify different subqueries of a request to the service provider, decide to which provider each subquery will be sent, and finally transform this subquery into a format the service provider understands.

Navigation is an obvious example; a user might not be willing to expose her entire trip plan to a particular route planner. Suppose, for simplicity; this user is requesting a route from A to B that passes through C. She can ask one navigation provider for a path between A and C and another for a path between C and B, then combine the two routes on the client device. That way, neither providers can independently infer the user's entire path. Another example is a user looking for a restaurant in her vicinity that serves a particular type of food and satisfies a price range. A provider with these two pieces of information might be able to associate the user with some group affiliation and socio-economic status. Alternatively, a user might send the first provider a subquery specifying restaurants serving some food and send the other provider another subquery about restaurants satisfying the price range of interest. Then, the user can choose from the intersection of the answers to the query.

Inter-query Diversity In some situations, we can exploit an orthogonal type of diversity, namely inter-query diversity. In inter-query diversity, each query is sent to a different service provider depending on the current query context. This aims to minimize side information leakage, as a service provider poses a privacy threat when it can associate different contexts with the same user. For example, a user using the same search engine to look for nearby local business, do health-related searches, follow political and religious news and topics, and shop for specific merchandise will leak sensitive information about her. More importantly, by combining this information, the search engine may be able to identify the user uniquely.

Diversifying access to service providers will reduce the amount and utility of the information available to each provider. If a user opts to send queries to a different provider de-

pending on the context, then each provider will associate the user with a minimum number of contexts, preferably one, thus reducing the probability of identifying the user. Building on the above example, if a user wants to find a nearby local business, she can use Yellow Pages at home and Yelp at work, to prevent either provider from deducing the identifying <home,work> location pair. Moreover, a user can diversify her queries, depending on their contents, among similar providers such as following political news on CNN and religious news on NYTimes or doing health-related online shopping on Amazon and clothes shopping on eBay.

Both diversity mechanisms can be combined to provide a higher level of privacy, especially in location-based services. If for an LBS, there are a sufficient number of service providers, then they can be grouped into sets. Each set of providers can be used at a different PoI to achieve inter-query diversity. Furthermore, the user can split her query among the providers within the set to achieve subquery diversity. Such a technique prevents a service provider from both detecting the user's sensitive PoIs and inferring her personal interests, thus protecting her location and query privacy.

In all, we propose a novel approach to protecting the user's privacy in the mobile setting. This method constitutes a client-side framework which does not rely on any third-party infrastructure or cooperation between multiple users. The proposed framework scales better than existing approaches and is resistant to mistrust issues and selfish behaviors. This framework protects both the users' privacy and the service providers' liability by reducing the utility of location information they store if they are leaked or hijacked.

Bibliography

[1] Kassem Fawaz and Kang G. Shin. Location privacy protection for smartphone users. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 239–250, New York, NY, USA, 2014. ACM.

[2] Kassem Fawaz, Huan Feng, and Kang G. Shin. Anatomization and protection of mobile apps' location privacy threats. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 753–768, Washington, D.C., August 2015. USENIX Association.

[3] Kassem Fawaz, Kyu-Han Kim, and Kang G Shin. Privacy vs. reward in indoor location-based services. *Proceedings on Privacy Enhancing Technologies*, 2016(4):102–122, 2016.

[4] Kassem Fawaz, Kyu-Han Kim, and Kang G. Shin. Protecting privacy of ble device users. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 1205–1221, Austin, TX, August 2016. USENIX Association.

[5] Ericsson. Ericsson Mobility Report. https://www.ericsson.com/res/docs/2015/ericsson-mobility-report-june-2015.pdf, June 2015.

[6] D. Evans. The internet of things. http://www.cisco.com/web/about/ac79/docs /innov/IoT_IBSG_0411FINAL.pdf, April 2011.

[7] Andrew J. Blumberg and Peter Eckersley. On Locational Privacy, and How to Avoid Losing it Forever. https://www.eff.org/files/eff-locational-privacy.pdf, August 2009.

[8] Jie Xiong and Kyle Jamieson. Arraytrack: A fine-grained indoor location system. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, nsdi'13, pages 71–84, Berkeley, CA, USA, 2013. USENIX Association.

[9] Elizabeth Dwoskin and Greg Bensinger. Tracking technology sheds light on shopper habits mall operators, retailers monitor patterns and actions. http://online.wsj.com/news/articles /SB10001424052702303332904579230401030827722, December 2013.

[10] Souvik Sen, Jeongkeun Lee, Kyu-Han Kim, and Paul Congdon. Avoiding multipath to revive inbuilding wifi localization. In *Proceeding of MobiSys '13*, pages 249–262, 2013.

[11] Bluetooth SIG. Specification of the Bluetooth System. Version 4.2, December 2014. https://www.bluetooth.org/en-us/specification/adopted-specifications.

[12] Gagan Luthra. Embedded controllers for the Internet of Things. http://www.edn.com/design/sensors/4440576/Embedded-controllers-for-the-Internet-of-Things/, Oct 2015.

[13] VICTORIA TURK. The internet of things has a language problem. http://motherboard.vice.com/read/the-internet-of-things-has-a-language-problem, Jul. 2014. Accessed: 03-02-2016.

[14] Andrii Degeler. Bluetooth low energy: Security issues and how to overcome them. https://stanfy.com/blog/bluetooth-low-energy-security-issues-and-how-to-overcome-them/, Jun. 2015. Accessed: 02-02-2016.

[15] Leslie Hart. Telit Acquires Wireless Communications Assets to Boost Capabilities in Low-Power Internet of Things Market). http://www.businesswire.com/news/home/20160113005310/en/, Jan. 2016. Accessed: 01-02-2016.

[16] Digi-Key Technical Content. Cypress PSoC 4 BLE (Bluetooth Low Energy). http://www.digikey.com/en/articles/techzone/2015/dec/cypress-psoc-4-ble-bluetooth-low-energy, Dec. 2015. Accessed: 12-01-2016.

[17] Pushek Madaan. IoT for the smarter home. http://www.ecnmag.com/article/2015/05/iot-smarter-home, May. 2015. Accessed: 11-01-2016.

[18] Aruba Networks. Data Sheet: Aruba 320 series access points. http://www.arubanetworks.com/assets/ds/DS_AP320Series.pdf.

[19] Scott Thurm and Yukari Iwatani Kane. Your Apps Are Watching You. http://online.wsj.com/article/SB10001424052748704694004576020083703574602.html, December 2010.

[20] Philippe Golle and Kurt Partridge. On the anonymity of home/work location pairs. 5538:390–397, 2009. 10.1007/978-3-642-01516-8_26.

[21] Scott Lester. The Emergence of Bluetooth Low Energy. http://www.contextis.com/resources/blog/emergence-bluetooth-low-energy/, May 2015.

[22] Jan Henrik Ziegeldorf, Oscar Garcia Morchon, and Klaus Wehrle. Privacy in the internet of things: threats andchallenges. *Security and Communication Networks*, 7(12):2728–2742, 2014.

[23] Oliver Jan, Alan J. Horowitz, and Zhong-Ren Peng. Using global positioning system data to understand variations in path choice. *Transportation Research Record: Journal of the Transportation Research Board*, 1725(2000):37–44, 2000.

[24] T. Dalenius. Finding a needle in a haystack-or identifying anonymous census record. *Journal of Official Statistics*, 2(3):329–336, 1986.

[25] John Krumm. Inference attacks on location tracks. In *In Proceedings of the Fifth International Conference on Pervasive Computing (Pervasive), volume 4480 of LNCS*, pages 127–143. Springer-Verlag, 2007.

[26] Yves-Alexandre de Montjoye, César A. Hidalgo, Michel Verleysen, and Vincent D. Blondel. Unique in the crowd: The privacy bounds of human mobility. *Sci. Rep.*, 3, Mar 2013.

[27] C. Bettini, X. Wang, and S. Jajodia. Protecting privacy against location-based personal identification. *Secure Data Management*, pages 185–199, 2005.

[28] Hui Zang and Jean Bolot. Anonymization of location data does not work: a large-scale measurement study. In *Proceedings of MobiCom '11*, pages 145–156, New York, NY, USA, 2011. ACM.

[29] Vibhor Rastogi and Suman Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, SIGMOD '10, pages 735–746, New York, NY, USA, 2010. ACM.

[30] Osman Abul, Francesco Bonchi, and Mirco Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, ICDE '08, pages 376–385, Washington, DC, USA, 2008. IEEE Computer Society.

[31] Manolis Terrovitis and Nikos Mamoulis. Privacy preservation in the publication of trajectories. In *Proceedings of the The Ninth International Conference on Mobile Data Management*, MDM '08, pages 65–72, Washington, DC, USA, 2008. IEEE Computer Society.

[32] Rui Chen, Gergely Acs, and Claude Castelluccia. Differentially private sequential data publication via variable-length n-grams. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12, pages 638–649, New York, NY, USA, 2012. ACM.

[33] Joseph Meyerowitz and Romit Roy Choudhury. Hiding stars with fireworks: location privacy through camouflage. In *Proceedings of MobiCom '09*, pages 345–356, New York, NY, USA, 2009. ACM.

[34] B. Gedik and Ling Liu. Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *IEEE TMC*, 7(1):1–18, January 2008.

[35] B. Palanisamy and Ling Liu. Mobimix: Protecting location privacy with mix-zones over road networks. In *Proceedings of ICDE '11*, pages 494 –505, april 2011.

[36] J. Freudiger, M.H. Manshaei, J.-P. Hubaux, and D.C. Parkes. Non-cooperative location privacy. *IEEE TDSC*, 10(2):84–98, March 2013.

[37] K.P.N. Puttaswamy, Shiyuan Wang, T. Steinbauer, D. Agrawal, A. El Abbadi, C. Kruegel, and B.Y. Zhao. Preserving location privacy in geosocial applications. *IEEE TMC*, 13(1):159–173, Jan 2014.

[38] Saikat Guha, Mudit Jain, and Venkata N. Padmanabhan. Koi: A location-privacy platform for smartphone apps. In *Proceedings of NSDI'12*, pages 14–14, Berkeley, CA, USA, 2012. USENIX Association.

[39] Shahriyar Amini, Janne Lindqvist, Jason Hong, Jialiu Lin, Eran Toch, and Norman Sadeh. Caché: Caching location-enhanced content to improve user privacy. In *Proceedings of MobiSys '11*, pages 197–210, New York, NY, USA, 2011. ACM.

[40] Kristopher Micinski, Philip Phelps, and Jeffrey S. Foster. An Empirical Study of Location Truncation on Android. In *Mobile Security Technologies (MoST '13)*, San Francisco, CA, May 2013.

[41] Andrew Leonard. *Wearable Honeypot*. PhD thesis, Worcester Polytechnic Institute, 2015.

[42] Ping Wang. Bluetooth low energy-privacy enhancement for advertisement. 2014.

[43] Federal Trade Commission. Internet of Things, Privacy & Security in a Connected World. https://www.ftc.gov/system/files/documents/reports/federal-trade-commission-staff-report-november-2013-workshop-entitled-internet-things-privacy/150127iotrpt.pdf, Jan. 2015.

[44] John Krumm. Realistic driving trips for location privacy. In *Proceedings of Pervasive '09*, pages 25–41, Berlin, Heidelberg, 2009. Springer-Verlag.

[45] Aniket Pingley, Nan Zhang, Xinwen Fu, Hyeong-Ah Choi, S. Subramaniam, and Wei Zhao. Protection of query privacy for continuous location based services. In *INFOCOM'11*. IEEE, April 2011.

[46] John Krumm. Inference attacks on location tracks. In *Proceedings of PERVASIVE '07*, pages 127–143. Springer-Verlag, 2007.

[47] R. Shokri, G. Theodorakopoulos, J. Le Boudec, and J. Hubaux. Quantifying location privacy. In *IEEE Symposium on Security and Privacy (SP),2011*, pages 247 –262, May 2011.

[48] Joseph Meyerowitz and Romit Roy Choudhury. Realtime location privacy via mobility prediction: Creating confusion at crossroads. In *HotMobile*, 2009.

[49] Hua Lu, Christian S. Jensen, and Man L. Yiu. PAD: privacy-area aware, dummy-based location privacy in mobile services. In *Proceedings of MobiDE '08*, pages 16–23, New York, NY, USA, 2008. ACM.

[50] Tun-Hao You, Wen-Chih Peng, and Wang-Chien Lee. Protecting moving trajectories with dummies. In *Mobile Data Management, 2007 International Conference on*, pages 278 –282, may 2007.

[51] Reza Shokri, George Theodorakopoulos, George Danezis, Jean-Pierre Hubaux, and Jean-Yves Le Boudec. Quantifying location privacy: the case of sporadic location exposure. In *Proceedings of PETS '11*, pages 57–76, Berlin, Heidelberg, 2011. Springer-Verlag.

[52] Alastair R. Beresford, Andrew Rice, Nicholas Skehin, and Ripduman Sohan. Mockdroid: Trading privacy for application functionality on smartphones. In *Proceedings of HotMobile '11*, pages 49–54, New York, NY, USA, 2011. ACM.

[53] PlaceMask. Placemask location privacy, May 2014.

[54] Mingyan Li, Krishna Sampigethaya, Leping Huang, and Radha Poovendran. Swing & swap: User-centric approaches towards maximizing location privacy. In *Proceedings of WPES '06*, pages 19–28, 2006.

[55] Marco Gruteser and Dirk Grunwald. Enhancing location privacy in wireless LAN through disposable interface identifiers: A quantitative analysis. *Mob. Netw. Appl.*, 10(3):315–325, June 2005.

[56] Tao Jiang, Helen J. Wang, and Yih-Chun Hu. Preserving location privacy in wireless LANs. In *Proceedings of MobiSys '07*, pages 246–257, 2007.

[57] Apple - privacy built in. `https://www.apple.com/privacy/privacy-built-in/`.

[58] Microsoft Trustworthy Computing. Location based services and privacy. http://www.microsoft.com/en-us/download/confirmation.aspx?id=3250, January 2011.

[59] Kathryn Zickuhr. Location-based services. http://pewinternet.org/Reports/2013/Location.aspx, September 2013.

[60] Kassem Fawaz and Kang G. Shin. Location privacy protection for smartphone users. In *Proceedings of CCS '14*, pages 239–250, New York, NY, USA, 2014. ACM.

[61] Huiqing Fu, Yulong Yang, Nileema Shingte, Janne Lindqvist, and Marco Gruteser. A field study of run-time location access disclosures on android smartphones. In *Proceedings of USEC 2014*.

[62] Drew Fisher, Leah Dorner, and David Wagner. Short paper: Location privacy: User behavior in the field. In *Proceedings of SPSM '12*, pages 51–56, 2012.

[63] John Krumm. A survey of computational location privacy. *Personal Ubiquitous Computing*, 13(6):391–399, August 2009.

[64] James Ball. Angry birds and 'leaky' phone apps targeted by NSA and GCHQ for user data. http://www.theguardian.com /world/2014/jan/27/nsa-gchq-smartphone-app-angry-birds-personal-data, January 2014.

[65] K.G. Shin, Xiaoen Ju, Zhigang Chen, and Xin Hu. Privacy protection for users of location-based services. *Wireless Communications, IEEE*, 19(1):30 –39, february 2012.

[66] Miguel E. Andrés, Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of CCS '13*, pages 901–914, New York, NY, USA, 2013. ACM.

[67] rovo89. Xposed module repository, May 2014.

[68] William Enck, Peter Gilbert, Byung-Gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N. Sheth. TaintDroid: An information-flow tracking system for realtime privacy monitoring on smartphones. In *Proceedings of OSDI '10*, pages 1–6, Berkeley, CA, USA, 2010. USENIX Association.

[69] Ryan Stevens, Clint Gibler, Jon Crussell, Jeremy Erickson, and Hao Chen. Investigating user privacy in android ad libraries. In *Mobile Security Technologies (MoST '12)*, May 2012.

[70] Michael C. Grace, Wu Zhou, Xuxian Jiang, and Ahmad-Reza Sadeghi. Unsafe exposure analysis of mobile in-app advertisements. In *Proceedings of WISEC '12*, pages 101–112, New York, NY, USA, 2012. ACM.

[71] Theodore Book, Adam Pridgen, and Dan S. Wallach. Longitudinal analysis of android ad library permissions. In *Mobile Security Technologies (MoST '13)*, San Francisco, CA, May 2013.

[72] Paul Pearce, Adrienne Porter Felt, Gabriel Nunez, and David Wagner. Addroid: Privilege separation for applications and advertisers in android. In *Proceedings of ASIACCS '12*, pages 71–72, New York, NY, USA, 2012. ACM.

[73] Justin Brickell and Vitaly Shmatikov. The cost of privacy: Destruction of data-mining utility in anonymized data publishing. In *Proceedings of KDD '08*, pages 70–78, New York, NY, USA, 2008. ACM.

[74] U.S. Census Bureau. US Census Bureau 2010 Census Interactive Population Map. http://www.census.gov/2010census/popmap/, 2014.

[75] Nevena Vratonjic, Kvin Huguenin, Vincent Bindschaedler, and Jean-Pierre Hubaux. How others compromise your location privacy: The case of shared public ips at hotspots. In Emiliano Cristofaro and Matthew Wright, editors, *Privacy Enhancing Technologies*, volume 7981 of *Lecture Notes in Computer Science*, pages 123–142. Springer Berlin Heidelberg, 2013.

[76] A. Bamis and A. Savvides. Lightweight extraction of frequent spatio-temporal activities from GPS traces. In *Proceedings of RTSS '10*, pages 281 –291. IEEE, December 2010.

[77] Rubin Xu, Hassen Saïdi, and Ross Anderson. Aurasium: Practical policy enforcement for android applications. In *Proceedings of USENIX Security '12*, pages 27–27, Berkeley, CA, USA, 2012. USENIX Association.

[78] Marta C. González, César A. Hidalgo, and Albert-László Barabási. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, June 2008.

[79] Benjamin Livshits and Jaeyeon Jung. Automatic mediation of privacy-sensitive resource access in smartphone applications. In *Proceedings of USENIX Security '13*, pages 113–130, Berkeley, CA, USA, 2013. USENIX Association.

[80] Anandatirtha Nandugudi, Anudipa Maiti, Taeyeon Ki, Fatih Bulut, Murat Demirbas, Tevfik Kosar, Chunming Qiao, Steven Y. Ko, and Geoffrey Challen. PhoneLab: A large programmable smartphone testbed. In *Proceedings of SENSEMINE '13*, pages 4:1–4:6, New York, NY, USA, 2013. ACM.

[81] Clayton Shepard, Ahmad Rahmati, Chad Tossell, Lin Zhong, and Phillip Kortum. Livelab: Measuring wireless networks and smartphone users in the field. In *HotMetrics*, 2010.

[82] Baik Hoh, Marco Gruteser, Hui Xiong, and Ansaf Alrabady. Achieving guaranteed anonymity in gps traces via uncertainty-aware path cloaking. *IEEE Transcations on Mobile Computing*, 9(8):1089–1107, August 2010.

[83] Clayton Shepard, Ahmad Rahmati, Chad Tossell, Lin Zhong, and Phillip Kortum. Livelab: measuring wireless networks and smartphone users in the field. *SIGMETRICS Perform. Eval. Rev.*, 38(3):15–20, January 2011.

[84] Julien Freudiger, Reza Shokri, and Jean-Pierre Hubaux. Evaluating the Privacy Risk of Location-Based Services. In *Financial Cryptography and Data Security (FC)*, 2011.

[85] Lothar Fritsch. Profiling and location-based services (lbs). In Mireille Hildebrandt and Serge Gutwirth, editors, *Profiling the European Citizen*, pages 147–168. Springer Netherlands, 2008.

[86] Scott Thurm and Yukari Iwatani Kane. Your apps are watching you. http://online.wsj.com/article/SB10001424052748704694004576020083703574602.html, December 2010.

[87] Warwick Ashford. Free mobile apps a threat to privacy, study finds. http://www.computerweekly.com/news/2240169770/Free-mobile-apps-a-threat-to-privacy-study-finds, October 2012.

[88] Peter Hornyack, Seungyeop Han, Jaeyeon Jung, Stuart Schechter, and David Wetherall. These aren't the droids you're looking for: retrofitting android to protect data from imperious applications. In *Proceedings of CCS '11*, pages 639–652, 2011.

[89] R. Shokri, G. Theodorakopoulos, J. Le Boudec, and J. Hubaux. Quantifying location privacy. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 247 –262, may 2011.

[90] Gennady Andrienko, Aris Gkoulalas-Divanis, Marco Gruteser, Christine Kopp, Thomas Liebig, and Klaus Rechert. Report from dagstuhl: the liberation of mobile location data and its implications for privacy research. *SIGMOBILE Mob. Comput. Commun. Rev.*, 17(2):7–18, July 2013.

[91] Jaeyeon Jung, Seungyeop Han, and David Wetherall. Short paper: Enhancing mobile application permissions with runtime feedback and constraints. In *Proceedings of SPSM '12*, pages 45–50, 2012.

[92] Hazim Almuhimedi, Florian Schaub, Norman Sadeh, Idris Adjerid, Alessandro Acquisti, Joshua Gluck, Lorrie Faith Cranor, and Yuvraj Agarwal. Your location has been shared 5,398 times!: A field study on mobile app privacy nudging. In *Proceedings of CHI '15*, pages 787–796, 2015.

[93] Aaron Clauset, Cosma Rohilla Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Rev.*, 51(4):661–703, November 2009.

[94] Szabolcs Vajna, Blint Tth, and Jnos Kertsz. Modelling bursty time series. *New Journal of Physics*, 15(10):103023, 2013.

[95] Omer Tripp and Julia Rubin. A bayesian approach to privacy enforcement in smartphones. In *USENIX Security 14*, pages 175–190, San Diego, CA, 2014. USENIX Association.

[96] Cathy Goodwin. A conceptualization of motives to seek privacy for nondeviant consumption. *Journal of Consumer Psychology*, 1(3):261 – 284, 1992.

[97] E. T. Higgins. Self-discrepancy: a theory relating self and affect. *Psychological Review*, 94(3):319–340, Jul 1987.

[98] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 03 1951.

[99] Reza Shokri, George Theodorakopoulos, Carmela Troncoso, Jean-Pierre Hubaux, and Jean-Yves Le Boudec. Protecting location privacy: Optimal strategy against localization attacks. In *Proceedings of CCS '12*, pages 617–627, 2012.

[100] Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Optimal geo-indistinguishable mechanisms for location privacy. In *Proceedings of CCS '14*, pages 251–262, 2014.

[101] Adrienne Porter Felt, Serge Egelman, Matthew Finifter, Devdatta Akhawe, and David Wagner. How to ask for permission. In *Proceedings of HotSec'12*, 2012.

[102] Igor Bilogrevic, Kvin Huguenin, Stefan Mihaila, Reza Shokri, and Jean-Pierre Hubaux. Predicting Users' Motivations behind Location Check-Ins and Utility Implications of Privacy Protection Mechanisms. In *NDSS'15*, 2015.

[103] Angela Martin. Nordstrom no longer tracking customer phones. `http://cbsloc.al/1JIYNlR`, May 2013.

[104] Future of Privacy Forum. Mobile Location Analytics Code of Conduct. http://www.futureofprivacy.org/wp-content/uploads/10.22.13-FINAL-MLA-Code.pdf.

[105] Ludovic Privat. U.S. consumers reject in-store tracking said survey. `http://www.opinionlab.com/media\_coverage/u-s-consumers-reject-in-store-tracking-said-survey/`.

[106] Heng Xu, Hock-Hai Teo, Bernard Tan, and Ritu Agarwal. The role of push-pull technology in privacy calculus: The case of location-based services. *J. Manage. Inf. Syst.*, 26(3):135–174, December 2009.

[107] James Hannan. Approximation to Bayes risk in repeated plays. *Contributions to the Theory of Games*, 3:97–139, 1957.

[108] Daniela Pucci De Farias and Nimrod Megiddo. Combining expert advice in reactive environments. *J. ACM*, 53(5):762–799, September 2006.

[109] Apple Support. iOS: Understanding iBeacon. https://support.apple.com/en-gb/HT202880, Feb. 2015.

[110] Rowena Rodrigues, David Barnard-Wills, David Wright, Paul De Hert, Vagelis Papakonstantinou, Laurent Beslay, EC JRC-IPSC, Nicolas Dubois, and ECDG JUST. EU privacy seals project. *Publications Office of the European Union*, 2013.

[111] Parker Higgins and Lee Tien. Mobile tracking code of conduct falls short of protecting consumers. `https://www.eff.org/deeplinks/2013/10/mobile-tracking-code-conduct-falls\\-short-protecting-consumers`, October 2013.

[112] Levent Demir, Mathieu Cunche, and Cédric Lauradoux. Analysing the privacy policies of Wi-Fi trackers. In *Workshop on Physical Analytics*, Bretton Woods, USA, June 2014.

[113] Christopher Riederer, Vijay Erramilli, Augustin Chaintreau, Balachander Krishnamurthy, and Pablo Rodriguez. For sale : Your data: By : You. In *Proceedings of HotNets-X*, pages 13:1–13:6, New York, NY, USA, 2011. ACM.

[114] Arpita Ghosh and Aaron Roth. Selling privacy at auction. In *Proceedings of EC '11*, pages 199–208, 2011.

[115] Reza Shokri. Privacy games: Optimal user-centric data obfuscation. *Proceedings on Privacy Enhancing Technologies*, 2015(2):1–17, 2015.

[116] Ponnurangam Kumaraguru and Lorrie Faith Cranor. Privacy Indexes: A Survey of Westins Studies. Technical report, Carnegie Mellon University, Institute for Software Research International, 12 2005.

[117] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. A predictive differentially-private mechanism for mobility traces. In *Privacy Enhancing Technologies*, pages 21–41. Springer, 2014.

[118] Gilles Barthe, Boris Köpf, Federico Olmedo, and Santiago Zanella Béguelin. Probabilistic relational reasoning for differential privacy. In *Proceedings of the 39th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '12, pages 97–110, New York, NY, USA, 2012. ACM.

[119] Jason Reed and Benjamin C. Pierce. Distance makes the types grow stronger: A calculus for differential privacy. In *Proceedings of the 15th ACM SIGPLAN International Conference on Functional Programming*, ICFP '10, pages 157–168, New York, NY, USA, 2010. ACM.

[120] Konstantinos Chatzikokolakis, MiguelE. Andrs, NicolsEmilio Bordenabe, and Catuscia Palamidessi. Broadening the scope of differential privacy using metrics. In Emiliano De Cristofaro and Matthew Wright, editors, *Privacy Enhancing Technologies*, volume 7981 of *Lecture Notes in Computer Science*, pages 82–102. Springer Berlin Heidelberg, 2013.

[121] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin Heidelberg, 2006.

[122] Gerome Miklau and Dan Suciu. A formal analysis of information disclosure in data exchange. In *Proceedings of SIGMOD '04*, pages 575–586, 2004.

[123] B.A. Huberman, E. Adar, and L.R. Fine. Valuating privacy. *Security Privacy, IEEE*, 3(5):22–25.

[124] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338 – 353, 1965.

[125] Jelena Demko-Rihter and Igor ter Halle. Revival of high street retailing the added value of shopping apps. *The AMFITEATRU ECONOMIC journal*, 17(39), 2015.

[126] H. Jang, I. Ko, and J. Kim. The effect of group-buy social commerce and coupon on satisfaction and continuance intention – focusing on the expectation confirmation model (ecm). In *System Sciences (HICSS), 2013 46th Hawaii International Conference on*, pages 2938–2948, Jan 2013.

[127] Tobias Kowatsch and Wolfgang Maass. In-store consumer behavior: How mobile recommendation agents influence usage intentions, product purchases, and store preferences. *Computers in Human Behavior*, 26(4):697 – 704, 2010. Emerging and Scripted Roles in Computer-supported Collaborative Learning.

[128] Isaac M. Dinner, Harald J. Van Heerde, and Scott Neslin. Creating Customer Engagement Via Mobile Apps:How App Usage Drives Purchase Behavior. *Social Science Research Network Working Paper Series*, October.

[129] Ju-Young M. Kang, Jung Mee Mun, and Kim K.P. Johnson. In-store mobile usage: Downloading and usage intention toward mobile location-based retail apps. *Computers in Human Behavior*, 46:210 – 217, 2015.

[130] Sales Force. 2014 Mobile Behavior Report. https://www.exacttarget.com/sites/exacttarget/files/deliverables/etmc-2014mobilebehaviorreport.pdf, Feb. 2014.

[131] Kasey Lobaugh;Jeff Simpson;Lokesh Ohri. The New Digital Divide: Retailers, shoppers, and the digital influence factor. http://www2.deloitte.com/content/dam/Deloitte/us/Documents/consumer-business/us-rd-thenewdigitaldivide-041814.pdf, 2014.

[132] Dan Kosir. Mobile apps vs. mobile web: What retailers need to know. http://clearbridgemobile.com/mobile-apps-vs-mobile-web-what-retailers-need-to-know/, Aug. 2015.

[133] Revital Libfrand. Retail Mobile App Users Visit Brick-and-Mortars More Often. http://blog.compariscope.com/retail-mobile-apps-12x-the-number-of-in-store-visits, Jan. 2016.

[134] Catherine Boyle. Mobile Messaging TrendsTapping into SMS, Mobile Email and Push. http://www.slideshare.net/eMarketerInc/emarketer-webinar-mobile-messaging-trendstapping-into-sms-mobile-email-and-push-25068768, Aug. 2013.

[135] Peifeng Yin, Ping Luo, Wang-Chien Lee, and Min Wang. App recommendation: A contest between satisfaction and temptation. In *Proceedings of WSDM '13*, pages 395–404, New York, NY, USA, 2013. ACM.

[136] aestetix and Christopher Petro. CRAWDAD data set hope/amd (v. 2008-08-07). Downloaded from http://crawdad.org/hope/amd/, August 2008.

[137] Travis Goodspeed and Nathaniel Filardo. CRAWDAD data set hope/nh_amd (v. 2010-07-18). Downloaded from http://crawdad.org/hope/nh_amd/, July 2010.

[138] Injong Rhee, Minsu Shin, Seongik Hong, Kyunghan Lee, Seongjoon Kim, and Song Chong. CRAWDAD data set ncsu/mobilitymodels (v. 2009-07-23). Downloaded from http://crawdad.org/ncsu/mobilitymodels/, July 2009.

[139] James Little and Brendan O'Brien. A technical review of cisco's wi-fi-based location analytics. http://www.cisco.com/c/en/us/products/collateral/wireless/mobility-services-engine/white_paper_c11-728970.pdf, July 2013.

[140] Derek Top. Indoor Location Firm Nomi Faces Layoffs; Privacy Concerns To Blame? http://t.co/e7Gp7mU1Sz, Aug. 2014.

[141] Bluetooth SIG. Bluetooth SIG Analyst Digest 2H 2014. https://www.bluetooth.org/en-us/Documents/Analyst2014. Accessed: 10-02-2016.

[142] Susan Kuchinskas. Bluetooth's smart future in telematics. http://analysis.tu-auto.com/infotainment/bluetooths-smart-future-telematics, March 2013.

[143] Andrea Peterson. Yes, terrorists could have hacked Dick Cheneys heart. https://www.washingtonpost.com/news/the-switch/wp/2013/10/21/yes-terrorists-could-have-hacked-dick-cheneys-heart/, Oct. 2013.

[144] John Pescatore. A SANS Analyst Survey: Securing the "Internet of Things" Survey. https://www.sans.org/reading-room/whitepapers/analyst/securing-internet-things-survey-34785, Jan. 2014. Accessed: 18-01-2016.

[145] Federal Bureau of Investigation. Internet of Things Poses Oppotunities for Cyber Crime. https://www.ic3.gov/media/2015/150910.aspx, Sep. 2015. Accessed: 18-01-2016.

[146] Article 29 Data Protection Working Party. Opinion 8/2014 on the on recent developments on the internet of things. http://ec.europa.eu/justice/data-protection/article-29/documentation/opinion-recommendation/files/2014/wp223_en.pdf, Sep. 2014. Accessed: 18-01-2016.

[147] Kashmir Hill. 'Baby Monitor Hack' Could Happen To 40,000 Other Foscam Users. http://www.forbes.com/sites/kashmirhill/2013/08/27/baby-monitor-hack-could-happen-to-40000-other-foscam-users, Aug. 2013. Accessed: 18-01-2016.

[148] Bruce Schneier. The internet of things is wildly insecure – and often unpatchable. http://www.wired.com/2014/01/theres-no-good-way-to-patch-the-internet-of-things-and-thats-a-huge-problem, Jan. 2014. Accessed: 18-01-2016.

[149] Ry Crist. Samsung swings for the fences with a new smart fridge at ces. http://www.cnet.com/products/samsung-family-hub-refrigerator, Jan. 2016. Accessed: 18-01-2016.

[150] Shrirang Mare, Jacob Sorber, Minho Shin, Cory Cornelius, and David Kotz. Hide-n-sense: Preserving privacy efficiently in wireless mhealth. *Mobile Networks and Applications*, 19(3):331–344, 2014.

[151] Shyamnath Gollakota, Haitham Hassanieh, Benjamin Ransford, Dina Katabi, and Kevin Fu. They can hear your heartbeats: Non-invasive security for implantable medical devices. In *Proceedings of the ACM SIGCOMM 2011 Conference*, SIGCOMM '11, pages 2–13, New York, NY, USA, 2011. ACM.

[152] Wenbo Shen, Peng Ning, Xiaofan He, and Huaiyu Dai. Ally friendly jamming: How to jam your enemy and maintain your own wireless connectivity at the same time. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 174–188, May 2013.

[153] Vijay Srinivasan, John Stankovic, and Kamin Whitehouse. Protecting your daily in-home activity information from a wireless snooping attack. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, UbiComp '08, pages 202–211, New York, NY, USA, 2008. ACM.

[154] Homin Park, Can Basaran, Taejoon Park, and Sang Hyuk Son. Energy-efficient privacy protection for smart home environments using behavioral semantics. *Sensors*, 14(9):16235, 2014.

[155] M.R. Schurgot, D.A. Shinberg, and L.G. Greenwald. Experiments with security and privacy in IoT networks. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2015 IEEE 16th International Symposium on a*, pages 1–6, June 2015.

[156] Marco Gruteser and Dirk Grunwald. Enhancing location privacy in wireless lan through disposable interface identifiers: A quantitative analysis. *Mobile Networks and Applications*, 10(3):315–325, 2005.

[157] Ben Greenstein, Damon McCoy, Jeffrey Pang, Tadayoshi Kohno, Srinivasan Seshan, and David Wetherall. Improving wireless privacy with an identifier-free link layer protocol. In *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, MobiSys '08, pages 40–53, New York, NY, USA, 2008. ACM.

[158] Aveek K. Das, Parth H. Pathak, Chen-Nee Chuah, and Prasant Mohapatra. Uncovering privacy leakage in ble network traffic of wearable fitness trackers. In *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications*, HotMobile '16, pages 99–104, New York, NY, USA, 2016. ACM.

[159] R. Want, B.N. Schilit, and S. Jenson. Enabling the internet of things. *Computer*, 48(1):28–35, Jan 2015.

[160] Ishtiaq Rouf, Rob Miller, Hossen Mustafa, Travis Taylor, Sangho Oh, Wenyuan Xu, Marco Gruteser, Wade Trappe, and Ivan Seskar. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In *Proceedings of the 19th USENIX Conference on Security*, USENIX Security'10, pages 21–21, Berkeley, CA, USA, 2010. USENIX Association.

[161] Muhammad Naveed, Xiaoyong Zhou, Soteris Demetriou, XiaoFeng Wang, and Carl A Gunter. Inside job: Understanding and mitigating the threat of external

device mis-bonding on android. In *Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS)*, pages 23–26, 2014.

[162] Simone Margaritelli. Nike+ FuelBand SE BLE Protocol Reversed. http://www.evilsocket.net/2015/01/29/nike-fuelband-se-ble-protocol-reversed/, Jan 2015.

[163] N.O. Tippenhauer, L. Malisa, A. Ranganathan, and S. Capkun. On limitations of friendly jamming for confidentiality. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 160–173, May 2013.

[164] T. OConnor and D. Reeves. Bluetooth network-based misuse detection. In *Computer Security Applications Conference, 2008. ACSAC 2008. Annual*, pages 377–391, Dec 2008.

[165] Mike Ryan. Bluetooth: With low energy comes low security. In *Proceedings of the 7th USENIX Conference on Offensive Technologies*, WOOT'13, pages 4–4, Berkeley, CA, USA, 2013. USENIX Association.

[166] Robin Heydon. *Bluetooth low energy: the developer's handbook*. Prentice Hall, 2012.

[167] Daniel T. Wagner, Andrew Rice, and Alastair R. Beresford. Device analyzer: Large-scale mobile data collection. *SIGMETRICS Perform. Eval. Rev.*, 41(4):53–56, April 2014.

[168] D.R. Cox. *Renewal theory*. Methuen's monographs on applied probability and statistics. Methuen, 1962.

[169] Hamza Harkous, Kassem Fawaz, Kang G. Shin, and Karl Aberer. Pribots: Conversational privacy with chatbots. In *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*, Denver, CO, June 2016. USENIX Association.

[170] FCC Wireless Telecommunications Bureau. Location-based services; an overview of opportunities and other considerations. http://www.fcc.gov/document/location-based-services-report, May 2012.