Revealing and exploiting privacy vulnerabilities in users' public wireless packets

Abhishek Kumar Mishra

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2023IPPAX087

Thèse de doctorat

ÉCOLE POLYTECHNIQUE

IP PARIS

Revealing and exploiting privacy vulnerabilities in users' public wireless packets

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à l'École polytechnique

École doctorale n°626 l'Institut Polytechnique de Paris (ED IP Paris)
Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Paris, le 19 Octobre 2023, par

ABHISHEK KUMAR MISHRA

Composition du Jury :

Katia Jaffrès-Runser Professor, Université de Toulouse (Toulouse INP ENSEEIHT)	Président
Hamed Haddadi Reader, Imperial College London (Department of Computing)	Rapporteur
Katia Jaffrès-Runser Professor, Université de Toulouse (Toulouse INP ENSEEIHT)	Rapporteuse
Viktoria Fodor Professor, KTH (Network and Systems Engineering)	Examinatrice
Mathieu Cunche Associate Professor, INSA-Lyon (Privatics)	Examinateur
Aline Carneiro Viana Directrice de recherche, Inria (Tribe)	Directrice de thèse
Nadjib Achir MCF, Inria (Tribe)	Co-directeur de thèse
Marco Fiore Professor, Imdea Networks (Network Data Science)	Invité

ABSTRACT

The increasing proliferation of Wireless Fidelity (WiFi) and Bluetooth Low Energy (BLE) networked devices broadcasting over-the-air unencrypted public packets has raised growing concerns regarding users' privacy. Such public packets consist of management frames, like probe-requests and beacons, necessary for devices to discover available wireless networks and enhance user experience. Revealing the MAC address of a device through public packets allows adversaries to follow the device and do behavioral profiling. Modern devices periodically change/randomize their advertised MAC addresses. Nevertheless, attacks on MAC address randomization have been carried out, demonstrating that randomized addresses from a device can be associated with as little information as the timestamps of their advertised public packets.

In this thesis, we identify key flaws that lead to the MAC association. To measure the severity of identified flaws by looking at the performance of current MAC association attacks, we need large-scale traces of public packets with "ground truth" information regarding randomized addresses from the same device. We assert the flaws by employing our proposed simulation framework to generate large-scale WiFi and BLE passive sniffing traces. We reveal that current device randomization is ineffective and needs revision.

In addition to key flaws identifications, we address the unreliability of existing association frameworks with respective trace collection scenarios to understand the factors contributing to variable association performance. We conduct case studies and introduce benchmarks for evaluating the performance of any association framework. We show the need for a new and effective WiFi MAC association framework, and finally, we develop and benchmark a novel association framework to determine its expected performance with any new input probe-request dataset.

Once achieving effective MAC association, we reveal the inference of user locations from passively sniffed probe-requests. In this thesis, we identify the limitations of the Received Signal Strength Indicator (RSSI) in accurately inferring user trajectories as a series of timestamped locations due to its high variability. Considering this, we propose a novel concept called "bounded trajectories." Bounded trajectory refers to an area where a particular user is probable to be present across time. We analyze and model the errors associated with radial-distance estimation to derive bounded trajectories that offer high inclusiveness of users' actual trajectory and narrow width throughout its course.

RÉSUMÉ

La prolifération croissante des dispositifs réseau WiFi (Wireless Fidelity) et Bluetooth Low Energy (BLE) diffusant des paquets publics non chiffrés par voie hertzienne a suscité de plus en plus de préoccupations concernant la confidentialité des utilisateurs. Ces paquets publics se composent de trames de gestion, telles que les trames *probe-requests* et les trames *beacons*, nécessaires aux appareils pour découvrir les réseaux sans fil disponibles et améliorer l'expérience utilisateur. La révélation de l'adresse MAC d'un appareil à travers les paquets publics permet de suivre l'appareil et de réaliser un profilage comportemental.

Les appareils modernes changent périodiquement leur adresse MAC annoncée, de manière aléatoire (randomization). Néanmoins, des attaques contre la randomisation des adresses MAC ont été menées, démontrant que des adresses aléatoires provenant d'un appareil peuvent être associées avec aussi peu d'informations que les horodatages de leurs paquets publics. Des vulnérabilités ont été découvertes dans les champs d'information des paquets qui pourraient révéler des informations privées sur l'utilisateur, telles que la détection de la langue des SSID WiFi diffusés et même des aspects sociologiques des personnes, comme la nationalité, l'âge et le statut socio-économique. L'état de l'écran allumé/éteint du smartphone peut être classifié en utilisant les motifs des sondes WiFi. De même, en utilisant les balises BLE, montre que le profilage de l'utilisateur, le détournement des balises, l'inférence de présence et même le harcèlement de l'utilisateur sont possibles.

Dans cette thèse, nous identifions les principales failles qui peuvent conduire à l'association des adresses MAC. Afin de mesurer la gravité des failles identifiées en examinant les performances des frameworks d'attaques d'association MAC existantes, nous avons besoin de traces d'écoute passive à grande échelle des paquets publics avec une "vérité terrain" concernant les adresses aléatoires générés par le même équipement. Finalement, nous mettons en évidence ces défauts et en proposons un outil de simulation pour générer des traces d'écoute passive WiFi et BLE à grande échelle. Nous révélons que la randomisation actuelle des appareils n'est pas suffisamment efficace et doit être révisée.

En plus de l'identification des failles clés, nous abordons l'imprécision des frameworks d'association existants avec des scénarios de collecte de traces respectifs pour comprendre les facteurs contribuant aux performances de l'association. Nous avons mené des études de cas afin d'en tirer des benchmarks pour évaluer les performances de tout framework d'association. Finalement, nous proposons et évaluons un nouveau framework d'association afin de déterminer ses performances attendues avec divers ensemble de traces en entrée.

Une fois l'association MAC obtenue, nous révélons l'inférence des emplacements des utilisateurs à partir des trames *probe-requests* captées passivement. Nous identifions les limites de l'indicateur de force du signal reçu (RSSI) dans l'inférence précise des trajectoires des utilisateurs en raison de sa grande variabilité. En tenant compte de cela, nous proposons un nouveau concept appelé "trajectoires bornées". Une trajectoire bornée désigne une zone où un utilisateur particulier est probablement présent dans le temps. Nous analysons et modélisons les erreurs associées à l'estimation de la distance radiale pour déduire des trajectoires bornées qui offrent une grande inclusivité de la trajectoire réelle des utilisateurs et une largeur suffisamment étroite tout au long de son parcours.

Après une association MAC réussie, de nouvelles opportunités pour tirer des informations précieuses à partir de paquets publics deviennent disponibles. Une inférence très recherchée implique l'extraction d'informations sur la mobilité des utilisateurs par le biais de la surveillance passive des données réseau. En utilisant des dispositifs de reniflage grand public tels que le Raspberry Pi, il est possible de capturer des paquets publics à grande échelle. Par exemple, en déduisant les emplacements des utilisateurs et, finalement, leurs trajectoires à partir des demandes de sondage WiFi, des applications précieuses peuvent être trouvées dans divers domaines tels que l'utilisation de l'espace urbain dans la prévention des épidémies, la réponse aux catastrophes, et l'activité socio-économique.

To my beloved mom (Rekha mishra) and dad (Jiwan Mishra),

This thesis stands as a testament to the unwavering support and boundless love you have showered upon me throughout my academic journey. Your sacrifices, encouragement, and belief in my abilities have been the guiding light that led me through the challenges of this research.

In dedicating this thesis to you, I want to express my deepest gratitude for the countless sacrifices you made to ensure that I had the opportunities and resources to pursue my dreams. Your unwavering faith in my potential has been a driving force behind my accomplishments.

As I take this moment to reflect on my academic achievements, I do so with immense pride, knowing that I owe it all to you. This thesis is not just a culmination of my hard work, but a testament to the incredible parents who instilled in me the values of perseverance, dedication, and a thirst for knowledge.

With all my love and gratitude,
Sonu

Acknowledgements

I take great pleasure in expressing my gratitude to the numerous individuals who have played a pivotal role in the realization of this thesis.

Foremost, I extend my heartfelt thanks to all the members of the INRIA Saclay research center and the IPP Doctoral School. Their warm welcome and the invaluable opportunity they provided me over the past three years have been instrumental in conducting the research presented in this thesis.

I would like to convey my deepest appreciation to the esteemed members of my thesis committee. I would like to thank Professor Hamed Haddadi from Imperial college, renowned for his inspirational contributions in the field of privacy for reviewing my manuscript. I am grateful to Professor Katia Jaffrès-Runser of Toulouse INP for her thorough review, relevant feedback, and acting as a president of the jury. I would like to thank Associate Professor Mathieu Cunche from INSA-Lyon and Professor Viktoria Fodor from KTH for serving as an examiner and providing critical insights of this thesis.

I am indebted to my thesis advisors, Dr. Aline Carneiro Viana and Dr. Nadjib Achir, for entrusting me with the opportunity to pursue this Ph.D. Without their valuable support, this thesis would not have been possible to realize. I must also express my profound gratitude to Professor Marco Fiore from Imdea, who not only made my research experience enjoyable during my visit to Madrid but also kindled my passion for research with valuable insights.

To my friends, I owe a debt of gratitude for their unwavering support and companionship. I would like to thank all those who provided the encouragement, support, and inspiration required to bring this project to fruition.

Last but certainly not least, I extend my thanks to my parents, to whom I dedicate this thesis. I am so grateful to my sister and my brother. Their unwavering trust and love have been my constant source of strength and motivation.

Contents

List of Figures

List of Tables

List of Acronyms

BLE	Bluetooth Low Energy
MAC	Media Access Control
AP	Access Point
SSID	Service Set Identifier
BT	Bluetooth Classic
RSSI	Received Signal Strength Indicator
SNR	Signal-to-noise Ratio
GATT	Generic Attribute Profile
IPSP	Internet Protocol Support Profile
LTE	Long-Term Evolution
IP	Internet Protocol
MIMO	Multiple Input Multiple Output
OFDMA	Orthogonal Frequency Division Multiple Access
TWT	Target Wake Time
BSSID	Basic Service Set Identifier
IBT	Inter-burst Time
IE	Information Element
SEQ	Sequence Number
UUID-Es	Universally Unique Identifier-enrollees
GPS	Global Positioning System
CDR	Call Detail Record
GIS	Geographic Information System
LoRa	Long Range
IFS	Inter-frame space
PNL	Preferred Network List
IRK	Identity Resolving Key
LAP	Lower Address Part
EWMA	Exponentially Weighted Moving Average
SA	Simulated Annealing
MOE	Margin Of Error

Chapter 1

Introduction

Contents

The rapid growth of WiFi and BLE networked devices has led to increasing concerns about user privacy. This thesis investigates the various privacy issues associated with these devices, ranging from the protection of receiver location to anonymity and traceability. Sniffing public wireless traffic, particularly probe-requests and beacon messages, has become a straightforward task, raising questions about user privacy.

Public packets are management frames that are necessary for the devices to find available networks and improve the user experience. Although initial measures were taken, such as *MAC address randomization*, recent studies have highlighted doubts about the effectiveness of these measures. In modern randomized WiFi devices, MAC addresses periodically change in order to avoid tracking from adversaries listening to their packets. Devices that perform MAC address randomization can hide the device's identity to some extent. This feature has been the backbone of user privacy in wireless networks, especially BLE and WiFi.

MAC address randomization in mobile devices has been thoroughly studied. It has been shown that it is not enough to safeguard user privacy. There have been attacks on fingerprinting devices using as minimal information as the timestamps of advertised public packets [6]. Further vulnerabilities have been discovered in the information fields of the packets that could reveal private information of the user like language detection on the broadcast WiFi SSIDs and even the sociological aspects of the people like nationality, age, and socioeconomic status [7]. Smartphone Screen ON/OFF State can be classified using WiFi Probe patterns [8]. Similarly, using BLE beacons, [9] show that user profiling, beacon hijacking, presence inference, and even user harassment are possible. Combining both BLE and WiFi's public packets

of the same user could lead to more devastating breaches in privacy. Both communities need to jointly come up with improved regulations and recommendations in the standard to ensure user privacy.

Defeating MAC address randomization allows the association of randomized addresses from the same device. Martin et. al claim to effectively defeat randomization for around 96% of android devices [10]. An artificial intelligence-based approach shows that 91% of the WiFi devices could be tracked [11]. Bluetooth Classic (BT) does not randomize the addresses and has already been shown to be de-anonymized [12]. Even MAC address randomization in BLE has been claimed to be defeated specifically for Apple devices [13] and generalized devices [1]. 100% device association for a small set of devices on sniffing public packets in a controlled environment (inside Faraday cage) is claimed [1]. We need to identify the validity of association approaches in large-scale real-world sniffing scenarios. Large-scale network traces with *ground truth* of devices generating the randomized MACs need to be collected or generated in order to ensure the reliability of such association frameworks in literature. If proven unreliable, we urgently need an efficient state-of-the-art address association framework.

Finally, after an effective MAC association, new avenues of utility inferences can be explored from public packets. One of the coveted inferences in obtaining user-mobility information from passive sniffing of network data. Public packets can be listened to on a large scale using off-the-shelf sniffers such as Raspberry Pi. Inferring user positions and eventually trajectory from WiFi probe-requests, for instance, can help out a lot of domains such as urban space usability in epidemic prevention [14, 15], disaster response [16], and socioeconomic activity [17].

1.1 Thesis Challenges

We begin by formulating three overarching questions, as depicted in Figure 1.1. The challenges associated with these questions form a central focus of this thesis. In the subsequent subsections, we explore the specific contributions that address these challenges.

Q_1 Are there privacy concerns from public wireless frames? - In the thesis, our initial inquiry revolves around examining the level of privacy risks associated with the existing design of public wireless frames. We specifically concentrate on identifying vulnerabilities in the design of two types of frames: WiFi probe-requests and BLE beacons. These frames pose potential threats to user privacy because they include the device's MAC address. Despite the periodic randomization of MAC addresses in these frames, there remains a possibility of establishing correlations or associations between a single user-device and

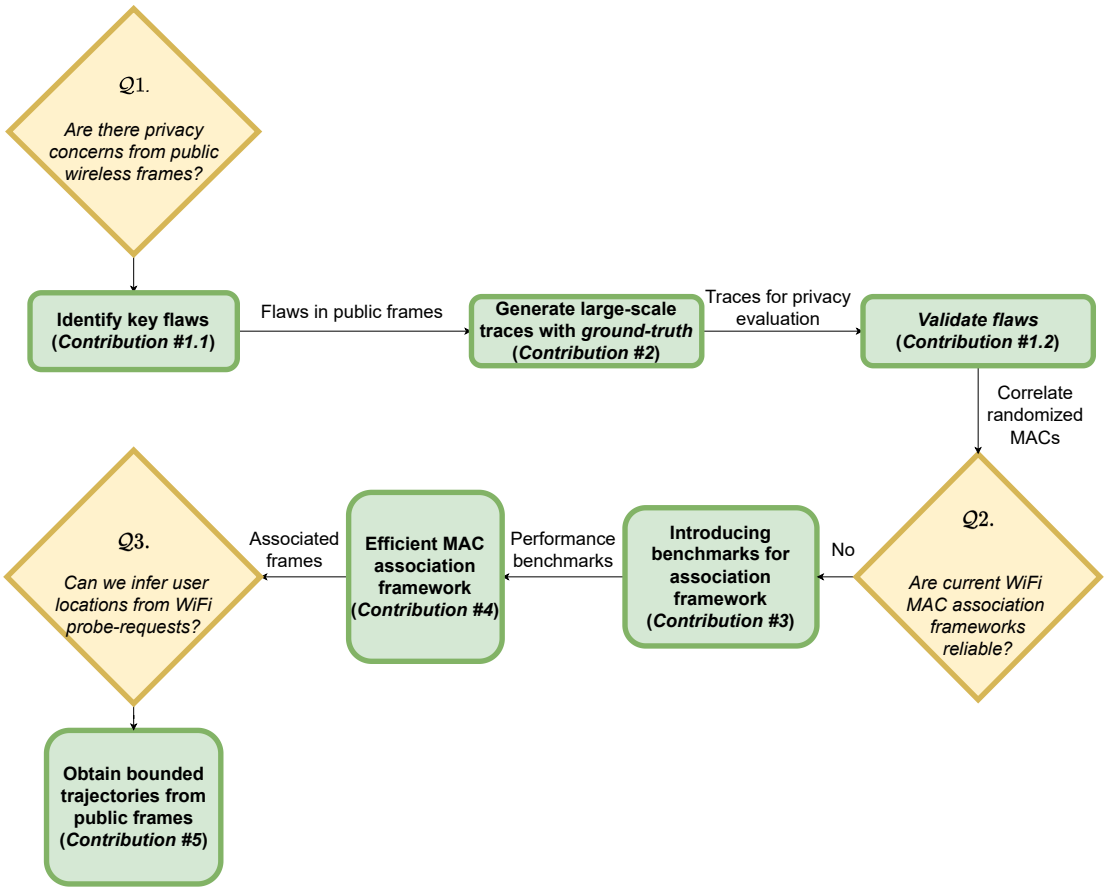


Figure 1.1: Thesis challenges.

their respective MAC addresses, primarily due to the current design and implementation of these frames.

The challenge lies in identifying the flaws in the current design of public frames. We need to gain further insights from the content and randomization behavior of probes. We need to generate large-scale passive sniffing BLE and WiFi traces. Collecting large-scale traces in the real-world requires permission from the authority in most geographical locations, along with financial and deployment overheads. Most importantly, in the passive real-world collection, we lack the *ground-truth* of randomized addresses from the same device. In order to validate any concerns in the design leading to the successful association of MACs, we need this ground-truth. New simulation frameworks must be introduced to generate realistic traces. Once flaws are identified and addressed, we can have recommendations/suggestions for current WiFi and BLE standards.

Q_2 Are current WiFi MAC association frameworks reliable? - The second question that we raise is investigating the already present MAC association frameworks in literature. We focus on the association concerning WiFi probe-requests from hereon in the thesis. The question of reliability arises from two implications of the current privacy provisions in WiFi:

(a) Current literature claims high but variable performances in MAC association either with their own small-scale controlled probe-request datasets or using indirect accuracy measures. For instance, [2] obtains a discrimination accuracy $> 80\%$ inside a shopping mall while [3] gets an accuracy of up to 75% in laboratory settings. [18] construct transmitter fingerprint that is 80% to 67.6% percent unique for 50 to 100 observed devices in a music festival and laboratory scenarios with a varied number of devices. The high accuracy, if true makes the user more susceptible to being tracked by an adversary. Frameworks lack large-scale traces with *ground-truth* for their evaluation. We need to conduct case studies to identify this unreliability, understand the cause of variable association performance with respect to input datasets, and, introduce benchmarks for any generic frameworks performance.

(b) If current MAC association frameworks are unreliable, we need a new and effective association framework. The framework should use direct accuracy metrics over large-scale real-world traces. Moreover, it should be bench-marked in order to know its expected performance with respect to any new input probe-request dataset.

Q_3 Can we infer user-locations/trajectories from WiFi probe-requests? - Assuming we have an effective MAC association framework, we ask the third question regarding the possibility of inferring user locations from passively sniffed WiFi probe-request frames. The received signal strength indicator (RSSI) of collected packets is the commonly used metric for passive user localization in literature. Besides offering the possibility to approximate the source-distance [19], RSSI has hardware generality, which brings measurement flexibility from any generic off-the-shelf hardware like Raspberry Pi. However, most current works in literature for RSSI-based localization focus on indoor scenarios [20–23], rather than outdoor. Outdoor settings have relatively high user mobility as well as incur unpredictable mobile obstacles like vehicles, which makes the trajectory inference harder.

When focusing on outdoor scenarios, there has been an attempt to use machine learning techniques to lower the distance estimation error [24].

Nevertheless, errors still go up to $16m$, even when considering semi-controlled environments like an isolated field, with a custom probe-request emitter. Hence, distance-estimation errors given by current related literature result in the significantly imprecise inference of users' locations. Such imprecision results in the location approximations of users that are uncertain, which is amplified in dense scenarios, with highly variable RSSI values.

We need to conduct a thorough examination of existing literature about the inference of user trajectories from network resources, with a specific focus on utilizing the RSSI as a potential means to leverage spatial information from captured frames. We have to ascertain the limitations of current solutions in accurately inferring user trajectories. Consequently, we need to introduce a novel framework to address this limitation. The new framework should circumvent the issue of uncertainty in RSSI-based location inference, enabling the determination of correct trajectories even in scenarios with unfavorable sniffer deployment conditions.

1.2 Thesis Contributions

In the following, we dig deeper into the identified questions while elaborating more on the thesis contributions.

1.2.1 Are there privacy concerns from public wireless frames?

Most of the existing related works suggest flaws in randomization [10] [11] [1] [6] or suggest inferring insights [25] [13] [9] from the transmitted public data in the packets. Encryption-based defense of WiFi 5 packets is suggested by [26], but as we discuss later in this thesis, such solutions are not feasible due to resource constraints. To the best of our knowledge, none of the current work gives a global view of the privacy issues in the design of public wireless packets itself. Current works do not look into countermeasures of timing-based attacks, which are more generic and effective than we will see in the upcoming sections. We find out key design flaws in WiFi probe-requests and BLE beacons.

To validate the above flaws, we need to first look into the issue of the lack of large-scale traces with ground truth of randomized MACs from the same device. We focus on BLE in the subsection.

We address a similar ground-truth issue for WiFi by introducing the framework *WiSurve*, which we detail in Chapter 6 while discussing an important use case of large-scale traces: inferring bounded trajectories from WiFi probe requests. There, we address the issue of getting a full-fledged *ground truth of mobile user locations*. Such ground-truth is almost impos-

sible in passive monitoring using WiFi sniffers: It requires knowing, in advance, the position of a representative number of mobile devices (i.e., the ground truth) passing by the deployed sniffers.

Coming back to BLE, amidst raising privacy intrusion findings, there has been an absence of frameworks to test these suggestions in scalable real-world conditions. Current BLE simulators are mostly focusing on throughput, latency, and signal-to-noise ratio (SNR) features rather than the security and privacy aspects of the standard. There has been an inability to incorporate real-world device parameters into the simulation framework. Without these advancements, it is impossible to generate a realistic BLE trace that considers integral factors like MAC address randomization. This is because the implementation of address randomization is dependent on the device manufacturer. Lack of controlled simulated traces presently halts the retrieval of *ground truth* in large-scale scenarios which is needed to successfully evaluate device fingerprinting solutions and propose adjustments in the standard to guarantee the user's privacy.

We develop the BLE simulator, `SimBle` to circumvent the current lack. To the best of our knowledge, none of the current available BLE simulators support and consider privacy aspects, specifically MAC address randomization. In this thesis, we first study and present different privacy guidelines across released Bluetooth standards. `SimBle` incorporates standard-compliant MAC address randomization, that is capable of emulating any BLE device. This is made possible as *SimBle* introduces the notion of *device class*, which differentiates various kinds of devices like phones, smartwatches, and headsets based on the frequency of transmitted beacons.

Additionally, existing simulation tools are not designed to generate sniffed public BLE traces. This limitation arises from the fact that simulation time drastically increases when handling a large number of devices, as the number of simulation events grows. However, our focus is solely on the complete processing of broadcast packets at the sniffer. To tackle this challenge, `SimBle` proposes optimized sniffing techniques that eliminate exponential run-time while generating an identical trace (cf. Chapter 3.2.1.1). The successful conquest of large-scale traces with *ground-truth* helps us in effectively addressing $\mathbf{Q}_1$.

Contribution #1 [$\mathbf{P}_3$]

1. Classification of current attacks in the literature based on different broad methodologies utilized by adversaries.
2. Revealing key design flaws in current WiFi and BLE public packets.
3. Validating the flaws in BLE utilizing large-scale realistic traces generated using our simulation framework.
4. Solutions and recommendations to rectify the flaws we detect in the design.

Contribution #2 [$\mathbf{P}_4$][$\mathbf{P}_5$]

1. Analysis of various privacy features in the BLE standard that are essential to be incorporated into simulations.

2. Design, implementation, and, the release of a new BLE simulation stack called `SimBle` in NS-3, which prioritizes user privacy and accurately represents the devices within the simulation.

3. Investigation of the only existing generic BLE MAC address association algorithm in the literature using `SimBle` for large-scale scenarios.

4. Development of an open-source simulator called `WiSurve`, capable of generating passively sniffed WiFi traces with *ground-truth* of user-locations.

1.2.2 Are current WiFi MAC association frameworks reliable?

Recent literature heavily investigates the *MAC address association*, i.e. co-relating randomized MAC addresses emitted by a particular device [2, 3, 18, 27, 28]. MAC association opens critical issues on users' privacy and henceforth brings the necessity for the understanding of such association, giving insights on the potential improvement to currently vulnerable WiFi standards, thus bringing stronger user privacy protection. The challenging issue concerning association strategies is identifying their weaknesses and effectiveness, which are not available. This thesis deals with this lack, which to the best of our knowledge, is the first work in literature.

Despite the promise shown by these address association frameworks, we show in this thesis their performances are highly sensitive to the input datasets, also showing there is space for WiFi standard's improvement. In this context, the *unreliable* nature of association frameworks calls for the need for *benchmarks* for measuring their real ability in providing a certain level of performance with respect to any new datasets (scenarios). This ensures WiFi users' privacy in a given scenario. After identifying the reliability of current association frameworks and obtaining benchmarks, we need *a generic characterization* of MAC association and *a new robust algorithm.*

The major hurdle in the probe request association happens in dense scenarios, resulting in higher simultaneous changes (*complexity*) in MAC addresses that we observe. The probability of a disappearing trail being successfully associated with an appearing trail decreases as the number of new address trails appearing in the sniffing zone is high. It increases the ambiguity, i.e., decreases the effectiveness of used signatures for distinguishing the correct association. This thesis identifies this unreliability and introduces a novel framework, named `Bleach`. It is efficient in run-time and

deployment and ensures its performance (accuracy greater than 60%) with growing *complexities* in the input probe-request dataset with randomized MAC addresses.

Contribution #3 [$\mathbf{P}_1$]

1. We introduce the benchmarks for WiFi privacy by giving a general characterization of the effectiveness of MAC address association. We explore and verify this unreliability and identify its causes. We show the impact of such causes on MAC address randomization.

2. We introduce a novel metric (conflict size) to ensure the reliability of frameworks. We use this metric to formalize the *complexity* that a framework is handling with respect to an input dataset. We discuss and validate the potential of randomization complexities in acting as an association framework's performance benchmarks.

3. The introduced randomization complexity distributions in the last step act as a *yardstick* for the level of user-privacy breach, when an adversary utilizes a particular address association framework. Benchmarks enable the understanding of the usefulness of MAC association frameworks while knowing their limitations, which are privacy-preserving for users.

Contribution #4 [$\mathbf{P}_6$]

1. We investigate and identify unreliability issues in state-of-the-art and call for the need for awareness to improve current address association frameworks. We generically characterize the MAC association to make the understanding better and the comparison feasible among the existing and future works in the MAC association.

2. Using metrics that we introduce to evaluate a signature, we identify effective time and frame-based signatures, in order to co-relate probes with randomized MACs originating from a single device.

3. `Bleach` associates randomized probe requests based on the obtained signatures and the distance metrics between them. We evaluate our framework for a wide range of scenarios, varying the degree of observed *conflicts*. We henceforth also investigate the performance of any association framework, to predict performances in any new dataset that we encounter.

4. We plan to release the open-source code of our framework on usage demo and potentially a few anatomized datasets too. We showcase that `Bleach` works reasonably well in scenarios with a high degree of *complexity*, i.e. extensive changes of MAC addresses in the sniffing zone.

1.2.3 Can we infer user locations from WiFi probe-requests?

Assuming successful MAC association, we move forward to accomplish the challenge of inferring user locations from WiFi probe-requests. Trajectory estimation of individuals is a cornerstone for different fields [29–31]. Localizing users in space and time is the first step toward trajectory inference, which involves spatiotemporal network data collection. This collection may focus on data directly extracted from smart devices [32] or carried-on dedicated beaconing devices [33] that depend on volunteer recruiting. This later is a time-consuming and non-scalable task. Alternatively, collection may be performed at Base Stations or Access Points [34], but suffers from access regulations by operators to access their infrastructure and recorded logs.

Using non-intrusive (passive) wireless measurements is a solution to circumvent the aforementioned issues [35]. It consists of deploying some sniffers in a particular area to passively measure the wireless activities therein –i.e., to collect public traffic (e.g., probe-requests) intrinsically sent by users' mobile WiFi interfaces.

RSSI is the most widely used metric for user-location inference from passive sniffing of WiFi probe-requests. Although considered the *"Achilles heel"* of RSSI-based user-localization solutions, *we transform localization errors induced in RSSI measurements into allies*, the first work of this type in literature to the best of our knowledge.

Our framework named `Allies` relies on an in-depth investigation and novel modeling of *(i) the severity, (ii) the non-stationary behavior, and (iii) the variability of RSSI-based distance-estimation errors* in *outdoor* passive measurements. The resulting error's comprehension leads us to identify and formalize *a span of possible user locations* associated with measurements in different time intervals. We denominate such span of possibilities as *bounds of locations* of a user. Bounded locations are then used to infer *bounds of users' full-fledged trajectories* over time that we designate as the user's *bounded trajectory*.

Contribution #5 [$\mathbf{P}_2$]

1. We carefully review the literature on inferring user trajectories from network resources and using RSSI as a potential candidate to exploit spatial information from captured frames. We identify the inability of current solutions in inferring accurate user trajectories. Henceforth, we introduce the novel concept of bounded trajectories

2. We characterize errors in radial-distance estimation. We find out that errors can broadly be divided into two major components: i) *Span error* and ii) *Environment error*, and could be modeled. While *Span error* showcases growing average estimation errors with increasing distance from the sender, *environment error* captures the error variability during short periods of time.

3. We build upon the modeled radial-distance errors to find bounded trajectories with a high degree of inclusiveness and low width. The resulting bounds circumvent the issue of uncertainty in RSSI-based location inference and give correct trajectories' bounds even in worst sniffer deployment scenarios.

1.3 Thesis outline

After the introduction, the thesis is divided into seven chapters.

- **Chapter 2** explains the state-of-the-art and the prerequisite notions for the rest of the thesis.
- **Chapter 3** focuses on interrogating privacy concerns in public WiFi and BLE frames. It starts with classifying the current attacks. Then it proposes key design flaws in public frames and introduces methodologies to obtain large-scale passively sniffed traces with *ground-truth*. Following, it validates the proposed flaws to finally give recommendations/suggestions to the standard/device manufacturers. [**Contribution #1 and #2**]
- **Chapter 4** first attempts to answer the question about the reliability of current WiFi address association frameworks. Following, it conducts case studies to validate the unreliability. To come up with performance benchmarks, it investigates the reason behind the variability in the performance of used signatures. Next, it introduces and discusses benchmarks that aid in predicting the performance of a framework with respect to a new input dataset. [**Contribution #3**]
- **Chapter 5** comes with state-of-the-art WiFi MAC association framework which is efficient and reliable. [**Contribution #4**]
- **Chapter 6** proposes a new framework that is capable of inferring *bounded* trajectory of users using WiFi probe-requests. The proposed bound are correct and exhibit a high degree of utility. [**Contribution #5**]
- **Chapter 7** provides overall conclusions and directions for future work.

Chapter 2

Background

Contents

Threats to user privacy through public wireless packets encompass a wide range of personal aspects. To begin with, we examine the existing privacy measures in two prominent wireless network protocols: BLE and WiFi. Both protocols rely on MAC address randomization as a fundamental safeguard for user privacy. However, the effectiveness of MAC address randomization has been challenged by the technique known as MAC address association, which allows the correlation of randomized MAC addresses with their originating devices.

While MAC association in WiFi has been extensively studied, the reliability of such association frameworks remains uncertain. This is primarily due to their limited scope of association and the absence of reliable reference data for randomized MAC addresses originating from the same device. To ensure the validity of these frameworks, it is crucial to utilize diverse and large-scale data sources for performance validation. Additionally, there exists the potential for passive inference of user trajectories from public wireless frames, a topic we delve into in this chapter.

In this thesis, we focus on WiFi probe-requests when considering the MAC association frameworks and the passive trajectory inference as BLE MAC addresses have already been shown to be associated with high accuracy [1]. WiFi MAC association is relatively more challenging due to the higher density of active devices in a region as well as the more frequent MAC

randomization. In this chapter, we also discuss the state-of-the-art related to the thesis-related topics.

2.1 Wireless network protocols

BLE and WiFi are prominent wireless protocols that complement each other in the wireless communication landscape. BLE excels in low-power, short-range applications that require periodic data exchange, while WiFi offers higher data rates and an extended range for internet connectivity and data-intensive tasks. In the following, we briefly look at both protocols while giving attention to their public packets which could potentially raise privacy concerns due to the advertisement of the device's MAC address.

2.1.1 BLE

Bluetooth has been there for quite some time now, but it is the Bluetooth Low Energy (BLE) variant [36] that is used by the majority of IoT devices. BLE has undergone significant evolution since its introduction. BLE was introduced as a part of the Bluetooth 4.0 specification in 2010. It was designed to address the need for low-power wireless connectivity for small devices, such as wearables, sensors, and Internet of Things (IoT) devices.

One of the primary goals of BLE was to provide improved power efficiency compared to classic Bluetooth. BLE achieved this by optimizing the protocol and introducing energy-saving modes. This made it suitable for battery-powered devices with limited power resources.

BLE initially introduced a set of core Bluetooth profiles that defined standardized ways for devices to communicate and interact. These profiles covered common use cases like data transfer (Generic Attribute Profile - GATT), heart rate monitoring, and proximity sensing.

Bluetooth 4.1 and 4.2 updates brought enhancements to BLE. Bluetooth 4.1 introduced features like improved coexistence with WiFi and LTE, while Bluetooth 4.2 added security enhancements and introduced the concept of Internet Protocol Support Profile (IPSP) for IP-based communication over BLE. Introduced in 2016, Bluetooth 5 brought significant improvements to BLE. It offered higher data transfer speeds, extended range, and enhanced advertising capabilities. Bluetooth 5 also introduced the concept of Bluetooth Mesh, enabling large-scale device networks for applications like smart lighting and home automation.

Overall, the evolution of BLE has focused on improving power efficiency, expanding functionality, increasing data transfer speeds, enhancing security and privacy, and enabling new use cases. These advancements have made BLE a crucial technology for various applications in the IoT, wearables, healthcare, smart home, and industrial sectors.

BLE beacons: When a particular BLE device communicates, it keeps sending advertising beacons on three public channels specified by the standard. These packets include a link-layer MAC address, which acts as an identifier of the device[[37], p. 69]. Various addressing modes have been specified in the standard [[38], p. p. 2988] which are briefly described next.

In BLE, we identify the devices using a device address and an address type [[38], p. 2988]. This means that whenever we compare two device addresses, the same 48-bit addresses do not guarantee the same device. This is because the two addresses could have different types. The address type could either be a public device address or a random device address, which are both 48 bits long. The device has the freedom to use at least one or both types of device addresses. Public device addresses are traditional MAC addresses that are created in accordance with *Universal addresses* section of the IEEE 802-2014 standard [39].

2.1.2 WiFi

WiFi began with the introduction of the IEEE 802.11 standard in 1997 [40]. The initial version, 802.11, supported a maximum data rate of 2 Mbps using frequency bands in the 2.4 GHz range. In 1999, the 802.11b standard was released, offering a significant improvement with a maximum data rate of 11 Mbps. It gained widespread adoption due to its higher speed and compatibility with existing 802.11 devices. The 802.11a standard operates in the 5 GHz frequency band and provides a maximum data rate of 54 Mbps. In 2003, the 802.11g standard combined the best features of 802.11a and 802.11b, operating in the 2.4 GHz band and offering a maximum data rate of 54 Mbps.

The 802.11n standard, released in 2009, brought significant advancements in speed and range. It introduced MIMO (Multiple Input Multiple Output) technology, allowing for multiple antennas to transmit and receive data simultaneously. 802.11n supported maximum data rates of up to 600 Mbps, providing better coverage and improved reliability. In 2013, the 802.11ac standard was introduced, operating in the 5 GHz band and offering even higher data rates than 802.11n. It introduced wider channel bandwidths, advanced MIMO techniques, and beamforming technology to improve performance. 802.11ac supports maximum data rates of up to several gigabits per second.

The 802.11ax standard, also known as WiFi 6, was released in 2019. It focuses on improving network efficiency in crowded environments, offering higher data rates, reduced latency, and increased capacity. WiFi 6 introduces OFDMA (Orthogonal Frequency Division Multiple Access) to enable simultaneous data transmission to multiple devices, as well as other advanced features like Target Wake Time (TWT) for power efficiency.

The evolution of WiFi has continuously focused on increasing data rates,

improving coverage and range, enhancing network efficiency, and addressing the needs of emerging applications. Each new generation of WiFi brings faster speeds, lower latency, and better performance, enabling a wide range of wireless communication and connectivity scenarios in homes, offices, public spaces, and various industries.

WiFi probe-requests: WiFi-enabled devices discover nearby wireless networks / *access points* using a method called active scanning. With the active scan, mobile devices find available networks by sending management frames known as *probe-request* frames. These frames are broadcasted periodically to reduce energy drainage.

The main components and information contained in a typical WiFi probe-request frame are as follows:

- **Frame Control:** This field includes control information about the frame, such as the frame type, subtype, and various flags. For a probe-request frame, the type would be "Management" (0x0) and the subtype would be "Probe-Request" (0x4).
- **Duration/ID:** This field specifies the duration for which the medium should remain reserved for this frame or an association identifier if used in an association process.
- **Destination MAC Address:** The MAC address of the receiving device, which would typically be a broadcast address (ff:ff:ff:ff:ff:ff) since the probe-request is sent to all nearby APs.
- **Source MAC Address:** The MAC address of the client device that is sending the probe-request frame.
- **BSSID:** The Basic Service Set Identifier (BSSID) is the MAC address of the AP that the client is associated with or the broadcast address if the client is not currently associated with any AP.
- **Sequence Control:** This field is used for frame sequencing and acknowledgment purposes.
- **Timestamp:** A timestamp value that represents the time at which the probe-request was sent.
- **Beacon Interval:** This field indicates the time interval between beacon frames sent by the AP.
- **Capability Information:** This field contains information about the capabilities of the client device, such as supported data rates, encryption methods, and power management capabilities.

- **SSID Information Element:** This is a variable-length field that contains the Service Set Identifier (SSID) of the network that the client is seeking. It can include the SSID itself or be left empty to indicate a wildcard request for all available networks.
- **Supported Rates Information Element:** This field specifies the data rates supported by the client device.
- **Extended Supported Rates Information Element:** This field provides additional information about supported rates, including higher data rates that the client device can handle.
- **Vendor-Specific Information Element:** This field allows for the inclusion of vendor-specific information or additional parameters for specific purposes.

These are the main components of a WiFi probe-request frame. When a client device sends a probe-request, nearby APs can receive and respond to it with a probe response frame, providing information about their capabilities, supported rates, and other network parameters.

2.2 Privacy provisions in public packets

The backbone for protecting user privacy in public wireless packets is the continuously changing advertised MAC addresses of a device which we also call MAC randomization. Next, we focus on BLE beacons and WiFi probe-requests in terms of their privacy provisions.

2.2.1 BLE beacons

To avoid the user leaking the identifier to the world, recent BLE standards have continuously forced all devices to update their publicly advertised MAC addresses. Public device addresses are more prevalent, but it is the random device address that is privacy-preserving. Random device addresses could either be static or private. A static address is a 48-bit randomly generated address meeting specific standard requirements. On the other hand, private addresses are again either resolvable or non-resolvable[[38], p. 2991]. These specific subtypes are identified by the two most significant bits of the random device address, as shown in the Table 2.1.

BLE device's Identity Address is one of a Public device address or a Random static device address. When a device is continuing with Resolvable private addresses, it must also possess an Identity Address.

The key to privacy provided by the BLE link layer is using private addresses, which we described in the previous sub-section[[38], p. 3201]. BLE recommends devices to generate a resolvable private address. The link layer

Address [47:46]	Address Sub-Type
0b00	Non-resolvable private address
0b01	Resolvable private address
0b10	Reserved for future use
0b11	Static device address

Table 2.1: Sub-types of random device addresses

corresponding to the host sets a timer and regenerates a new resolvable private address when the timer expires. Moreover, once the Link Layer is reset, a new resolvable private address is generated, and the timer is allowed to start with an arbitrary value in the allowed range. To maintain the efficiency of connection establishment, the standard recommends setting the timer to 15 minutes.

BLE [41] [38] does not allow private devices to use its Identity Address in any advertising packet. The Host could instruct the Controller to advertise, scan, or initiate a connection using a resolvable private address after enabling the resolving list.

The state machine for the link layer of BLE consists of various states[[38], p. 2985]. A device could be found in either of these states. For instance, advertising, scanning, and initiation states have different guidelines by the standard. In the advertising state, the link layer is allowed to perform device filtering based on the device address of the peer device to minimize the number of devices to which it responds. This could be done according to a local *white list* which contains a set of records comprising of both the device address and the device address type (public or random) [[38], p. 3202]. If the device is in a scanning or initiating state, it is recommended to use private addresses. The scanning device should use the resolvable or non-resolvable private address as the device address. Whenever a scanning device receives an advertising packet that contains a resolvable private address for the advertiser's device address, after address resolution, the scanner's filter policy decides whether to respond with a scan request or not.

Despite the above privacy provisions in BLE, it has been exposed to the MAC address association. MAC association refers to defeating the anonymization techniques used by the devices and being able to track a particular device. Recently many strategies have been suggested to achieve this goal of associating different private addresses advertised publicly from the same device [1] [42] [43] [13]. For instance, [42] [43] show that manufacturers like Apple and Microsoft leak partial identifiers in the data field of public packets, which can be easily exploited. In [13], authors reverse engineer continuity protocol messages of Apple devices. They show that fingerprinting the device, as well as behaviorally profiling users, is possible by using the contents of public BLE messages. They also demonstrate that

predictable frame sequence numbers in them leave the possibility of tracking Apple devices across space and time.

2.2.2 WiFi probe-requests

WiFi MAC address randomization is a privacy provision implemented in some devices and operating systems to enhance user privacy and security while connecting to wireless networks. The MAC address is a unique identifier assigned to network interfaces, including WiFi adapters, by the manufacturer.

Fig. 2.1 illustrates the active scanning process from a WiFi device over time, when searching for a connection. Mobile devices send probe-requests on each available channel to obtain replies from all accessible access points. Each device does multiple rounds of active scanning over the available channels. Active scanning rounds last for a duration in the order of a few seconds depending upon the number of known access points that it could direct to and the available channels. By default, when devices transmit probe-requests, they use their physical or *true* MAC address, which can be easily tracked and used to identify and track the device and its user. However, with MAC address randomization, the device generates a random MAC address for each WiFi connection, effectively concealing the true MAC address.

The purpose of MAC address randomization is to prevent the long-term tracking of a device's movements and activities by various entities, such as WiFi network operators, advertisers, and malicious actors. It helps protect user privacy by making it difficult to associate a specific MAC address with a particular device or individual.

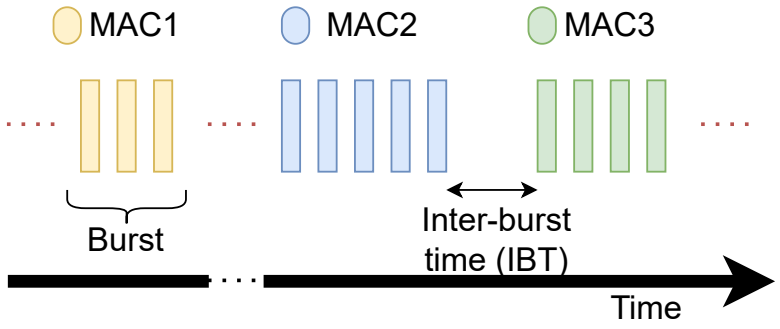


Figure 2.1: A device's randomised probe-requests.

As we notice in Fig. 2.1, multiple rounds contain a *burst* of probe-requests with the MAC address of individual probes within a burst remaining consistent. At the same time, the MAC is likely to change (randomize) in subsequent or after a certain number of bursts. The number of bursts advertising a MAC is variable and is generally dependent on the manufacturer

and the device's state. The period between successive bursts, and inter-burst time (IBT) is variable. Most modern WiFi devices do not advertise their true MAC address unless they broadcast the randomized one.

Legacy devices though could still send their true MAC addresses. The availability and implementation of MAC randomization may vary depending on the manufacturer and the specific WiFi adapter in use. As we see in the subsequent subsection, Literature already suggests that randomized MAC addresses in probe-requests could be co-related to the sender device using various attack methodologies.

2.3 WiFi MAC address association

WiFi MAC address association refers to co-relating multiple randomized MACs emitted from the same device. As illustrated in Fig. 2.2, two MAC addresses advertised from the same device, MAC1 and MAC2 in probe-requests could be associated. It could utilize various metrics associated with the probe-request transmission and reception at the sniffer. The metrics could include information such as the received signal-strength (RSSI1 and RSSI2), information elements contained in the probe-requests (IE1 and IE2), the difference between the sequence numbers of two frames (SEQ1 and SEQ2), as well as the frame reception timings of various probe bursts.

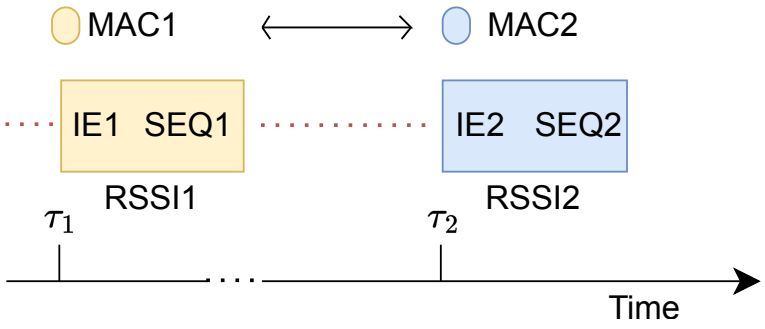


Figure 2.2: MAC association of WiFi probe-requests

The literature for address association exploits two main directions: i) the *leaks* in system design or protocols and ii) device *signatures* extracted from the probe-request transmission itself. *Signatures* are information extracted from metrics obtained from probe-request transmission, which are capable of distinguishing a device successfully from a population.

Early works in address association utilize the information leaks in protocols or system design to co-relate randomized MAC addresses. [28] and [44] do the reverse engineering of the universally unique identifier-enrollees (UUID-Es) of probe-requests to find the true MAC addresses using precom-

puted hash tables. [28] exploit the auto-connect feature of certain devices to malicious but popular SSID names, as they could potentially leak their true MAC address. Certain devices assign consecutive MAC addresses for BLE and WiFi, which can be exploited to reveal the WiFi MAC address [10]. But these schemes are not generic and rely upon flaws in the system design which is usually fixed by the manufacturer upon attention. As already illustrated in Fig. 2.2, recent association frameworks broadly use four metrics for signatures, namely: the information element (IE), the temporal information, the sequence numbers, and, the RSSI.

The first metric leveraged by the frameworks is the content in the information element (IE) field of the probe-requests to fingerprint devices [28], [18], [45], [46]. The field contains information regarding device capabilities [45] or the service set identifier (SSID) [18], [46], which is exploited for the association. Works select specific combinations of IE fields to maximize the effectiveness of the used signature [28].

On the other hand, some works extract the temporal information from the received probe-requests, such as inter-arrival times in order to distinguish various devices with randomized MACs. The hypothesis is that various devices will have different unique patterns of this temporal behavior due to manufacturer differences and builds [47], [48], [3], [49], [50].

Alternatively, works also exploit the continuity of sequence numbers to distinguish between various randomized MACs that change during a particular duration. These frameworks rely upon the range of sequence numbers that could potentially be advertised in the next frame from different transmitting devices [28], [51], [52]. Finally, there are works that utilize the RSSI vectors recorded by various sniffers listening to the same frame. The idea is devices that change MAC addresses, will have similar and unique RSSI vectors, eventually helping in address association [2].

2.4 Trajectory inference from public wireless frames

Trajectory investigation has attracted considerable literature attention. With mobile phones becoming proxies for human presence, network resources have been exploited to investigate users' mobility [53]. Spatio-temporal mobility datasets are nowadays acknowledged as a common tool to study users' trajectories: e.g., mobile phone records [54, 55], GPS [56], WiFi [15, 29, 30], and BLE data [57].

The pioneering work [55] uses anonymous call detail records (CDRs) from mobile phones to infer human mobility patterns in an urban area. The study analyzes CDRs from millions of individuals to understand movement patterns, commuting flows, and predict future locations. The study by [58] proposes a trajectory inference method based on user mobility features extracted from mobile phone records. The authors apply clustering

algorithms to identify significant places and predict users' next locations using their historical mobility patterns. While [59] focuses on trajectory similarity analysis and matching in large-scale spatial networks using mobile phone records. The authors propose efficient algorithms for trajectory similarity joins, enabling the identification of similar movement patterns and trajectory associations.

The work by [60] focuses on inferring transportation modes from GPS data collected from smartphones. They show that using information from pervasive WiFi access points and Bluetooth devices can enhance GPS and geographic information to improve transportation detection on smartphones. When broken out by individual modes, WiFi features improve the detection accuracy of bus trips, train travel, and driving compared to GPS features alone and can substitute for GIS features without decreasing performance. [61] use the time- and geo-coordinates associated with a sequence of posts/tweets, to discover people and community behavior. They infer interesting locations and frequent travel sequences among these locations in a given geo-spatial region, as shown from the detailed analysis of the collected geo-tagged data.

However, mobile phone records are only accessible through network infrastructure and service providers. Inferring trajectories based on GPS data requires access to users' devices. User location sharing is privacy-sensitive and hence is a difficult task to infer mobility through it on a large scale. Moreover, the majority of works that require the active collection of traces through mobile devices require volunteer recruiting which is cumbersome and lacks scalability.

Instead, we base our investigation on datasets that are non-intrusive and independent of third parties. We focus on collection performed via WiFi passive sniffing. Passive sniffing refers to the wide-scale deployment of sniffers in public spaces which listen to WiFi public management frames on multiple frequency channels. Management frames are legal to capture with the user's consent in most geographical locations. We stick to public frames for inferring user trajectories. Merging different views of sniffers capturing the same frame from a user device helps to localize the device which in time can be extended to infer the mobility.

RSSI is the most widely used localization metric, with no hardware-specific requirements. In the indoor controlled environment, previous works attempt to localize users through RSSI-based distance measurements [20, 22, 24, 62]. On the other side, in outdoor scenarios, equal attempts mainly focus on Zigbee [63] and LoRa [64] technologies. The literature lacks works leveraging RSSI for outdoor users' localization, mainly due to its imposed uncertainties. Uncertainties incur significant errors in distance estimation, making users localization a challenging task.

Solutions using RSSI for estimating users' trajectories contain localization errors arising from inaccurately measured distances between the user

and the regarded sniffer. Past works have tried to mitigate such errors to some extent using various smoothing techniques for RSSI [65–68]. However, smoothing itself causes bias (cf. Section 6.1.2). Another approach [69] adapts the path-loss model parameters by deploying fixed reference nodes and reference tags placed at known positions. This approach requires extra hardware and is primarily developed for indoor localization.

Although the employed effort, errors remain an issue. Without effective modeling and characterization of localization errors, inference of user's mobile trajectories through passive sniffing is imprecise and unreliable. To the best of our knowledge, none of the literature solutions focus on (i) the characterization of localization errors in outdoor scenarios and (ii) its leveraging for users' trajectory inference.

2.5 Literature datasets

Majority of MAC association frameworks in literature use datasets for evaluation which are collected in controlled environments [3, 18, 28, 45, 46, 51] like a laboratory and are not public. Own collected datasets specifically on small-scale leads not only to replicability issues but also makes the framework unreliable with respect to a new input dataset with a large number of devices.

In this thesis, we utilize a highly diverse and public *Sapienza* dataset as well as a recent large-scale collection inside a big shopping mall, named *HongKong* dataset. There is a *labeled dataset* [5] that we utilize too.

1. Sapienza dataset: Wireless probe-requests were gathered in Rome from February to May 2013, during which mobile devices sought to switch to WiFi connectivity whenever available automatically. To facilitate this process, devices stored a list of the network names (SSID) to which the user typically connected. Periodically, these SSIDs were broadcasted through probe-Requests to search for accessible networks. Several intriguing questions naturally emerged as a consequence: "What information can be inferred from smartphone probes?" and "Can meaningful relationships among individuals be deduced solely from their smartphones' probes?"

To address these inquiries comprehensively, [70] conducted a probe collection campaign in Rome, specifically targeting a university campus as well as city-wide, national, and international events. The campaign spanned three months and relied solely on commodity hardware to amass approximately 11 million probes sent by approximately 160 thousand unique devices. The released dataset contains anonymized traces in .pcap format, allowing for further analysis and investigation.

The data collection campaign specifically targeted various scenarios to gather diverse datasets. These scenarios included: i) events for national audiences, such as political meetings (*Politics1* and *Politics2*); ii) events for

international audiences, such as the Pope's Angelus at St. Peter's Square, Vatican (*Vatican1* and *Vatican2*); iii) a shopping mall (*mall*), iv) a train station (*trainstation*), and v) a university campus (*university*). The first four scenarios comprised one-shot events, each lasting from 40 minutes to 6 hours. For these events, a team of 5 researchers conducted the physical data collection using their laptops. The fifth dataset was obtained through a fixed hardware setup positioned within a university campus. This setup allowed for continuous data collection over a period of 6 weeks. [70] use scapy to dissect and anonymize all MAC addresses and SSIDs in the dataset.

2. HongKong dataset: HongKong dataset is derived from the collection performed by a customized sniffing system used in [2]. The system comprises multiple WiFi sensors and a centralized server. For the sensors, they utilized commercial WiFi Access Points (APs) specifically GL-AR150 with OpenWrt 18.06. These sensors capture probe-requests using the libpcap library. The captured frames are then transmitted via Ethernet to the server for storage and further processing. The server itself is built on a PC equipped with an Intel Core i7 3.6 GHz CPU and 16 GB RAM.

Experiments were conducted in a large shopping mall located in Hong Kong. The experiments encompassed an entire floor with an area of approximately 8000 m2. Figure 9 illustrates the floor plan, indicating the placement of the WiFi sensors, totaling 21 sensors. These sensors were installed on the ceiling and operated on channel 1 of the 2.4 GHz frequency band. The system has been operational for over three months, collecting data continuously. During a typical business hour on a weekend day, the system captured approximately 20,000 frames, corresponding to around 5,000 unique MAC addresses.

3. Labeled dataset: Unlike *Sapienza* datasets, the labeled probe-request dataset [5] provides a wide range of current mobile devices with additional information such as the device's *mode* when transmitting. This dataset has 22 popular device models in practice, which were sniffed when present in various device *modes* (see Tab. 2.2). This is the first open-source probe-requests dataset, *labeled* with the *ground truth* of randomized addresses. It contains 20-minute duration captures of known devices using a Raspberry Pi-based sniffer.

Mode	Screen ON	Power-saving ON	WiFi ON
A	Yes	No	Yes
S	No	No	Yes
PA	Yes	Yes	Yes
PS	No	Yes	Yes
WA	Yes	No	No
WS	No	No	No

Table 2.2: Device's *modes* [5]

There are active-screen modes (A, PA, and WA) and inactive-screen modes (S, PS, and WS). In power-saving modes (PA and PS), the device additionally keeps the power-saving setting active, while in WA and WS modes, the device also has the WiFi interface switched off. Each device configuration is observed in the three non-overlapping channels (1,6, and, 11) of the 2.4 GHz frequency band. The details of capture conditions for each device mode can be found in [5]. In the following, we discuss the metrics concerning WiFi probe-request emissions that facilitate the association.

Awareness requirement: Despite the above three major probe-request datasets in the literature that we use, there is a necessity of developing new simulators to come up with *synthetic datasets*. These datasets make it possible to have the flexibility of sniffer deployments, user density, and, mobility; when replicating real-world passive sniffing. Synthetic datasets help in replicating MAC address randomization and related privacy provisions to yield large-scale traces with *ground-truth* of randomized MACs from the same device. Moreover, an effective simulation solution for passive sniffing could potentially tackle the issue of *ground-truth* of user-device locations when transmitting probe-requests. This helps in validating the solutions related to the inference of user trajectories from public wireless frames.

2.6 Conclusion

BLE and WiFi are widely deployed wireless protocols that transmit public frames, such as beacons and probe-requests, which contain MAC addresses and can make users vulnerable to tracking. BLE provides privacy through the use of private addresses generated through MAC address randomization. WiFi devices utilize active scanning with MAC address randomization in probe-requests, but research suggests the potential for MAC address association attacks.

WiFi MAC address association involves correlating randomized MAC addresses using metrics like received signal strength, information elements, sequence numbers, and RSSI vectors. Evaluation of association frameworks often relies on limited-scale datasets, posing challenges for scalability and replicability. Trajectory investigation utilizes various spatiotemporal datasets, but accessing certain data sources like mobile phone records and GPS data can be challenging due to privacy concerns.

WiFi passive sniffing provides localization through RSSI, but errors in outdoor scenarios and lack of modeling and characterization of localization errors remain challenges. Future research should focus on addressing these challenges and leveraging localization error characterization for accurate trajectory inference in passive sniffing scenarios.

In the next chapters, we address all the above-mentioned issues ranging from i) investigating privacy flaws in current wireless packets, ii) generat-

ing large-scale datasets with *ground-truth*, iii) questioning the reliability of current MAC association frameworks, iv) developing a novel and reliable association framework, and v) inferring trajectories effectively from passively collected probe-requests.

Chapter 3

Privacy concerns from MAC association

Contents

With growing privacy concerns over the last decade, two of the most notable wireless technologies – i.e., BLE and WiFi – are being more and more investigated regarding privacy vulnerabilities. The literature suggests that using various attack methodologies, randomized MAC addresses in beacons, and probe-requests could be co-related to the sender device. This process is called the MAC address association. The majority of user-privacy attacks in BLE and WiFi are concerned with MAC association.

In this thesis, we explore this problem, prospect the related consequences, and alert the need for privacy-preserving public frames. We identify key flaws in the current design of public frames. We discuss them as the cause of privacy issues that require the community's attention. We address the flaws and discuss potential solutions that facilitate the devices to protect

user privacy. We also give recommendations based on the findings to the standard in Section 3.4.

3.1 Flaws in BLE and WiFi public frames

Before investigating the current flaws in BLE and WiFi public frames leading to MAC association, we look at the classification of current attacks, majorly MAC association. Literature can be classified into three broad categories based on the attack methodology used by the adversary:
1. Timing-based attacks: These kinds of attacks rely upon the temporal information that could be extracted from the observed public packet sequence from a device [6]. The adversary's aim here is to extract metrics that are characteristic of a device and remain consistent over a period of time irrespective of address randomization. Examples of such metrics are Inter-frame space (IFS), inter-burst duration, etc.
2. Frame-field attacks: Here, an attacker learns the information fields in the frames that are generally sent in the clear to classify and subsequently fingerprint devices [26]. There are flags, client capability information, Manufacturer names, frame type, etc., which possess the potential of being part of a fingerprint.
3. Inference attacks: In the case of inference attacks, an adversary observes the activity of a user along with frames it sends over a period of time to infer private information [7] [8]. Preferred network list(PNL) in the WiFi probe requests, variation in two attempts of probes, etc., are some of the information that helps the attacker to learn regarding the user under threat.

3.1.1 Key design flaws

Learning from the above-classified attacks, we identify the following key flaws in the current design and implementation of public frames.

1. *Ineffective address randomization* - MAC address randomization, if implemented effectively, can prevent user tracking to some extent. The current implementation of randomization is not adaptive to the user's surroundings and is predictive.

2. *Uniform timing parameters* - Parameters specific to the timing of advertised public frames are uniformly distributed across the device population. Most of them are manufacturer-specific and even vary within a brand. This makes users fall prey to fingerprinting and profiling.

3. *Inadequate privacy measures in clear-text frames* - The majority of current public packet fields are sent in plain text. They lack necessary

privacy measures to prevent inference, even though many of these fields contain potentially private-intruding information.

The first flaw of the ineffective MAC address randomization in current BLE and WiFi devices is the most prominent vulnerability that needs to be addressed. The second flaw of the presence of uniform timing parameters in public frames is also closely knit to the ease of address association or, conversely, defeating MAC randomization. We need first to replicate real-world sniffed traces that an adversary could potentially acquire to associate randomized addresses from a target device. The third flaw questions the presence of plain text information in different fields of public packets and questions the feasibility of various encryption techniques.

3.1.2 Asserting the flaws

After identifying the flaws, we need to assert them through various means such as i) looking at the potential of literature attacks on available public datasets, ii) analyzing the current MAC randomization schemes, and iii) investigating the packet content of probe-requests and beacons.

While the last two means are relatively straightforward, the availability of public datasets which are passively sniffed and are on a relatively large-scale remains an open issue in the literature. The issue lies in government regulations on passive sniffing and the non-feasibility of large sniffer deployments. If we successfully develop a device-specific privacy-preserving simulation, we could easily produce traces that resemble real passive-sniffing scenarios. This has profound implications.

It enables us to practically evaluate any MAC address-based device-fingerprinting or privacy-intrusion solutions that are suggested in the literature. It is essential in validating the ineffectiveness of current MAC randomization. Specifically for BLE, where high association accuracy has been claimed for controlled environments [1], we design `SimBle`, a simulator based on top of NS-3, that provides large-scale traces with *ground-truth* of randomized MAC addresses from a device. We emulate devices that follow network and device privacy provisions of BLE.

While asserting the flaws, we utilize all three means that we mention before though we give major attention to evaluating the effectiveness of literature attacks on MAC randomization. This chapter primarily focuses on BLE, while also including recommendations for WiFi. Subsequent chapters will discuss in detail the efficiency of current WiFi MAC association frameworks and propose a benchmark for such frameworks with respect to input datasets (cf. Chapter 4). We continue to propose a new MAC association framework that is efficient and reliable across a set of input datasets (cf. Chapter 5).

In the next sections, we first develop and validate `SimBle` to verify the

ineffectiveness of current address randomization in BLE. Then, in Section 3.4, we come back to discuss the key design flaws in detail, considering both BLE and WiFi. We further proceed to provide solutions/recommendations based on the observed flaws.

3.2 Asserting flaws in BLE beacons

To assert the identified flaws in BLE beacons, we first address the need for large-scale datasets with *ground-truth*, for evaluating the potency of BLE MAC association attacks. The designed simulation framework `SimBle` is capable of generating a wide range of real-world passive-sniffing scenarios while incorporating the standard compliant and device-specific MAC randomization provisions.

3.2.1 `SimBle`: Obtaining BLE traces with ground-truth

The framework of `Simble` generates real-world privacy-preserving traces and uses them to test the BLE standard's privacy. The framework can be divided into three major components: 1) Formalisation, 2) Simulation, and 3) Evaluation. *Formalisation* handles two main issues: capturing the *device heterogeneity* in the population and emulating a particular standard's privacy provisions.

Simulation takes care of deploying real-world nodes belonging to different manufacturers, incorporating device mobility and most importantly, optimizing the run-time of the sniffer-capture to practical bounds. *Evaluation* collects per-sniffer traces, extracts the *ground truth* information and subsequently evaluates standard's privacy-intruding solutions. We detail *Formalisation* in this section while we describe *Simulation* and *Evaluation* in 3.2.3. `SimBle` accepts any user-specific parameters like those related to mobility and address randomization.

In the following, we first look at different design aspects of `Simble` and then we present our `Simble` architecture. We present a study of privacy provisions currently proposed by the standard. Finally, we identify the factors that must be taken into account for designing a simulator that respects user privacy. The simulator should not only care about including resolvable private addresses that are integral to BLE privacy but also bring together other MAC address randomization-related aspects. The proposed simulation stack `Simble`, is thus designed in such a manner that adding further privacy-specific features in the future is relatively straightforward.

3.2.1.1 Design considerations

The first aspect that we should take into consideration is the *device heterogeneity*. The second aspect that we focus on is to make sure that the gener-

ated traces are *realistic*. Finally, we also want to ensure that the framework is efficient and capable of generating large-scale traces in practical time.

A. Device heterogeneity: As discussed earlier in the previous section, different vendors have the freedom with some bounds in the implementation of the BLE stack in the device. For example, Apple picks from the range of values to decide how frequently the device changes a randomized MAC address. We need to distinguish each device introduced in `Simble` so that the simulation would be able to replicate its behavior in terms of privacy features. In the following, we define the device's type through two points: the device's class and the supported standard version.

1. *Notion of Device Class:* We find a property to classify the device into various groups where the behavior is similar irrespective of manufacturer. This property is the *frequency of transmitting beacons*, which is characteristic of a device with a maximum variation of 10ms [41, p. 2751]. The base value of the beacon transmission period is between [20 ms; 10.24 s]. Based on this property, we classify BLE devices into the following *device classes*:

    - *Frequent Emitters*: For this class, the frequency of transmitting beacons is from a normal distribution of mean 50 ms and standard deviation 10 ms. This represents a highly active device like earbuds. We expect these kinds of devices to also swap their randomized MAC address quickly.
    - *Moderate Emitters*: These are devices with a moderate frequency of advertisements. We describe them to be from a normal distribution of mean 300 ms and standard deviation 25 ms. From our experimentation, most smartphones, especially iPhones, are falling into this category.
    - *Semi-Moderate Emitters*: These are devices that are still active in transmitting regular beacons on broadcast channels. They follow a normal distribution of mean 500 ms and standard deviation 25 ms. This class again mainly includes phones.
    - *Low Emitters*: These are devices that are least active in sending out advertisements. We define them to have inter-beacon transmission intervals from a normal distribution of mean 2 s and standard deviation 500 ms. Smartwatches generally fall in this category.

    A user, when instantiating a node in `Simble` could choose any of the stated device classes. If the user enables beacons, nodes automatically set their behavior to that of the specified class. However, we give the

flexibility to specify the exact beacon frequency of a device if a user knows it beforehand through experimentation.

2. *BLE standards:* The frequency of changing a randomized MAC address does depend on the standard. In the most prevalent release currently in terms of the number of devices, the BLE 4.0, for instance, devices change their MAC addresses every 15 minutes [37]. In recent releases like BLE 5.2, devices are allowed to change their address before 15 minutes too. Therefore, it is crucial to specify a BLE node with its standard before using its privacy features in the simulation. `Simble` gives the user the option to mention the standard they want to run on top of the declared node, which enables controlling the privacy features associated.

B. Realistic trace generation: BLE passive sniffing generally refers to listening on public channels using sniffers. `SimBle` introduces a framework for the user to deploy an arbitrary number of sniffers and nodes to be placed in a sniffing zone. On top of it, different mobility models could be installed on BLE nodes' varying densities, which enables recreating realistic environments. Hence, we could emulate real-world BLE sniffing. Finally, introducing privacy in BLE simulation automatically answers the search of *ground truth* of randomized-address traces. The framework generates *ground truth* trace by matching each device's generated private addresses to the *Node ID*, which acts as a unique identifier to the device in simulation time.

C. Optimizing trace generation: We identify a major issue in the generation of real-world traces inside a simulation. As the number of nodes increases, the number of simulation events due to processing inter-node frames also increases quadratically. This has a significant impact on the time and resources needed for simulation. But we are only interested in the node-sniffer interaction in the case of public packet capture.

`Simble` addresses this problem and gives the user the flexibility to specify a *flag* in simulation, which induces filtered and optimized handling of broadcast frames at nodes. This reduces the simulation duration significantly and thus makes trace collection feasible. We discuss more on this and look at the obtained gain in performances in Section 3.2.2.3.

3.2.1.2 Architecture

After having figured out the design, we have a brief look into the architecture of a BLE *Node* inside `Simble` in Figure 3.1. As discussed earlier in Section 1.2.1, we use the base BLE stack of [71]. Components of *NetDevices* except the *PrivacyManager* were defined in the base stack. *Application* and *Packet socket interface* are NS-3 wide entities not specific to BLE. We created the new component, *PrivacyManager* that takes care of all BLE privacy features.

A node in `Simble` carries the same meaning as in NS-3. It is a physical entity with a unique integer ID and contains *NetDevices* and *Applications*.

In this thesis, we could think the *Node* to be equivalent to a device/hardware in the real world. We show in Figure 3.1 a single instance of *Application* and *NetDevice* for illustration but could be multiple in principle. *NetDevice* is an integral object of a node representing a physical interface on it. Here, we are interested in the Bluetooth interface. *NetDevice* communicates with the help of interfaces to the *Application*. *Packet socket interface* connects the application interfaces to the *NetDevice* here. IPv4/IPv6 stack could also be installed by the user on the node in parallel. Let's have a brief look at the roles of other components of NetDevice which were already present in the base BLE stack [71].

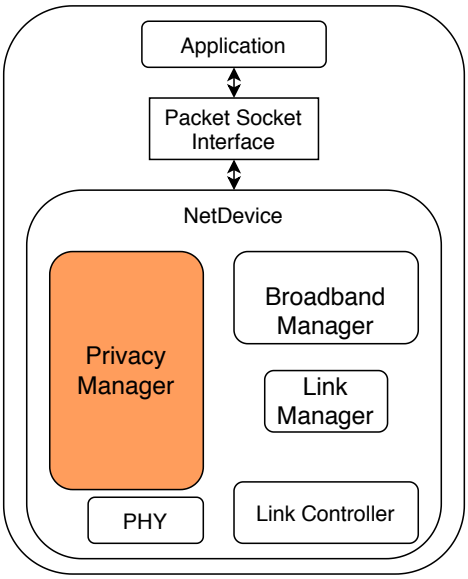


Figure 3.1: Architecture of a node in `Simble`

BroadbandManager helps add a link to the list of links that can be associated with a NetDevice. A link here refers to a BLE association between two nodes. It also handles checking if there are new frames in the NetDevice queue and forwards them to the right *LinkManager's* queue. *LinkManager* is the entity associated with a particular BroadbandManager. It setups a link to a specific receiver with the role(Master/Slave) as expected at the end of the setup process. LinkManager also manages *TransmitWindow* which is the next time the device can send a packet over the associated link.

LinkController is majorly responsible for monitoring and handling the re-transmissions and state changes in the link. It checks if the *ACK* was re-

ceived for the sent packet and also fires a list of callbacks to other NetDevice objects if the link changes. Lastly, *PHY* mainly takes the responsibility of handling link bandwidth, bit-rates, transmission power, and bit errors. We introduce a new module, *PrivacyManager* in `Simble` which takes care of all the privacy-related aspects of a device. In the forthcoming section, we discuss how MAC address randomization is managed by the *PrivacyManager*.

3.2.1.3 Privacy provisions in `SiMBle`

Hereafter, we describe the *PrivacyManager* implementation and the MAC address randomization of BLE. We describe in detail the implementation of *PrivacyManager* or, to be specific, the MAC address randomization. All the introduced algorithms follow the BLE standard guidelines [38].

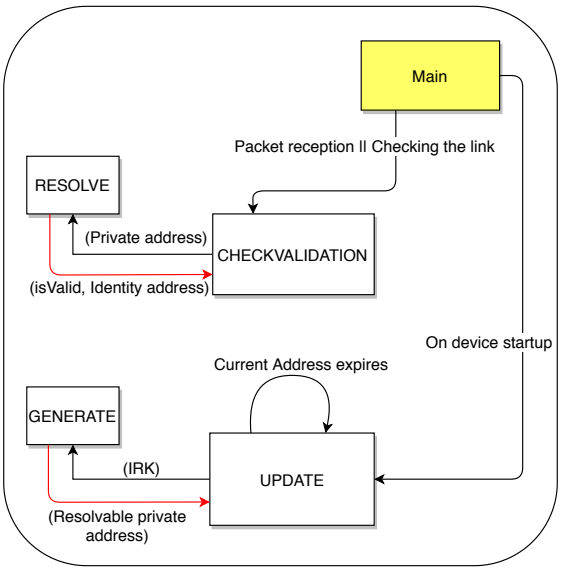


Figure 3.2: *PrivacyManager* in `Simble`

An overview of the *PrivacyManager* is illustrated in Figure 3.2. *Main* in the figure represents the base class of the *PrivacyManager* from which member functions are called. We could observe in the figure that the function UPDATE is called on the device startup. UPDATE generates new Resolvable private addresses for the calling node using the function GENERATE. It recursively calls itself after the expiration of the time associated with the current private address. In the event of packet reception or checking of the existence of a link to a destination, CHECKVALIDATION is called. On every call, it checks with RESOLVE with a particular private address. RESOLVE returns in turn the validity status and the device's identity address, which generated the private address. In the following, we describe the functions of *PrivacyManager* in detail.

KEY generation and distribution: *PrivacyManager* focuses on supporting Resolvable private addresses – the center of all privacy provisions in current BLE release [38] (cf. Section 2.2.1) For a node to generate a resolvable private address, it must have either the Local Identity Resolving Key (IRK) or the Peer Identity Resolving Key (IRK). This 128-bit key is proof of possession of a particular private address. In real devices, IRKs are exchanged through specific control messages. In `Simble`, we generate IRK randomly at each Node when it is initialized in the simulation. The delay caused in the key exchange for real hardware is emulated by *swapDelay* which we describe in the next section. Simultaneously, the Node also generates an Identity Address, which is a unique identifier of the device.

In this thesis, the Node or the *NetDevice* essentially means the same in terms of BLE-associated parameters. This is because the remaining modules inside the node (i.e., the socket and the application modules), are not dependent on the BLE standard itself. Finally, before creating links in `Simble` and installing an application on top declared nodes, each node updates a list in their respective *NetDevice*. This list contains (IRK: Identity Address) pairs of each of the fellow BLE nodes instantiated in the simulator.

We detail the randomized address generation and resolution inside `SimBle` in A.1 and A.2 respectively.

3.2.2 Validating `SimBle`

In the following, we focus on validating the private address generation and resolution, as well as the optimized trace collection which enables us to use `SimBle` in validating BLE flaws in Section 3.2.3.

3.2.2.1 Private address generation

To know if `Simble` can emulate a real-world trace, we first collect real traces obtained from real experimentation. Then, we compare the difference between real traces obtained from capturing public frames from actual devices to that of traces generated from initializing similar behavior devices inside the simulator. This comparison aims to show that `Simble` could emulate the same behavior in terms of randomized MAC advertisements and the transmission of public frames.

As a sniffer, we use the Bluetooth chipset of the Raspberry Pi 4B to capture Bluetooth public frames. Capture is done in a controlled environment inside a Faraday cage. We choose two devices Apple iPad Pro 3 and iPad Mini 2, emitting public frames in the cage for 40 minutes using BLE 4.1, which is captured by the Raspberry Pi. We are mainly interested in captured timestamps and LAP (lower address part) of the advertised beacons in the collected traces. LAP refers to the least significant 24 bits of a BLE MAC address. Even though we do trace collection in non-public

Parameter	Value
Simulation area	10*10
Packet size	20 bytes
Simulation duration	2410 seconds
Packet sending Duration	2400 seconds
Path loss model	nakagami
Num of nodes	N
Mobility model(nodes)	static
Num of sniffers	M
Mobility model(sniffer)	static
beacon interval	2 seconds
Connection Interval	6.25ms
Swap delay	10* beacon interval
BLE standard	BLE 4.1

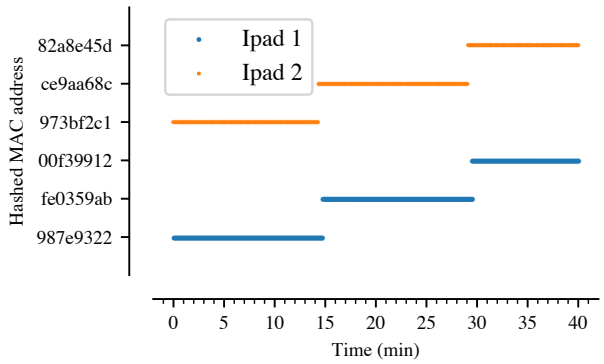
Table 3.1: Simulation parameters for `Simble` validation

environments, we still present hashed values to protect the device's privacy.

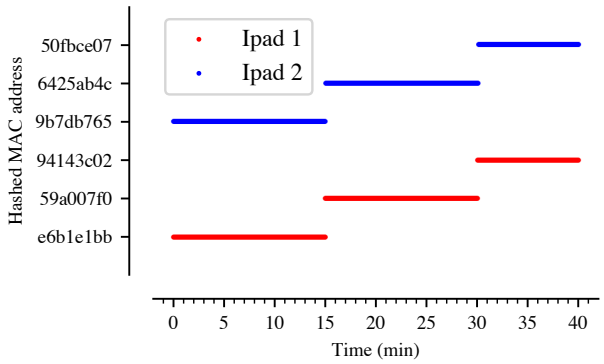
While for the devices inside the simulator, we assign the BLE standard in initialization as release 4.1, which fixes the interval of MAC address regeneration to 15 minutes. Afterward, we install a broadcast application on top of spawned nodes. We assign the frequency of beacon transmissions in the application as the mean device broadcast interval observed from the real-world sniffer capture. We found this value to be 2 seconds. Moreover, we place a sniffer at the center of a square area of 10 meters in which initialized emitting devices are statically present. Sniffer captures three public BLE channels. The chosen area's size is kept small to avoid transmission errors because of the distance between the devices and the sniffer. This is because errors are not present in the Faraday cage real-world experiment described earlier. The simulation parameters are illustrated in Table 3.1.

The first observation is related to the changing of the MAC addresses. In this case, for the real experiments, we turn on the Bluetooth of the two IPad devices at the start of sniffing since the otherwise first change in MAC address would be random, and it would be hard to use that trace for validation. As we can see in Figure 3.3a, randomized MAC addresses change every 15 minutes along with the capture duration. Like real IPad devices, IPads emulated inside the simulation change their MAC addresses after 15 minutes, shown in Figure 3.3b.

After validating the role of *PrivacyManager* in private address generation, we validate if the rest of the BLE stack could emulate the chosen real device. We do this by looking at the inter-packet times for public frames observed at the sniffer inside the `Simble` and the real world. We maintain



(a) Real-World



(b) SimBle

Figure 3.3: Observed public packet addresses in real-world vs `Simble` by two devices.

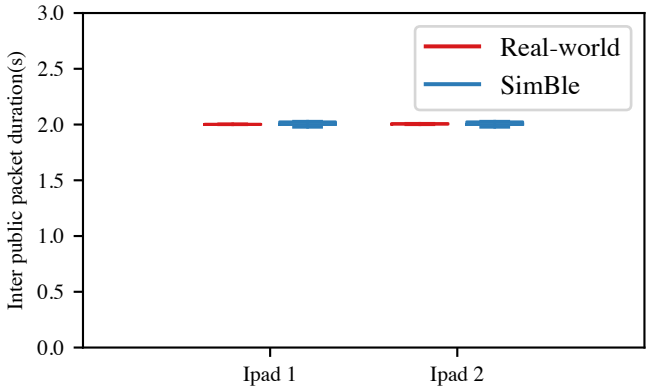


Figure 3.4: Real-world vs SimBle in inter public packet times

the same experimental setup and generated traces. We observe in Figure 3.4 that for both the devices, real-world and `Simble` inter-packet intervals at

the sniffer have the mean value of 2 seconds. A deviation of 20 milliseconds is expected for the sniffers as they capture on either of three public BLE channels at random and may miss some public frames on one of the three channels. A public packet on Bluetooth is broadcasted on all three public channels within a time frame of 20 milliseconds. This validates the overall working of public frames in `Simble`.

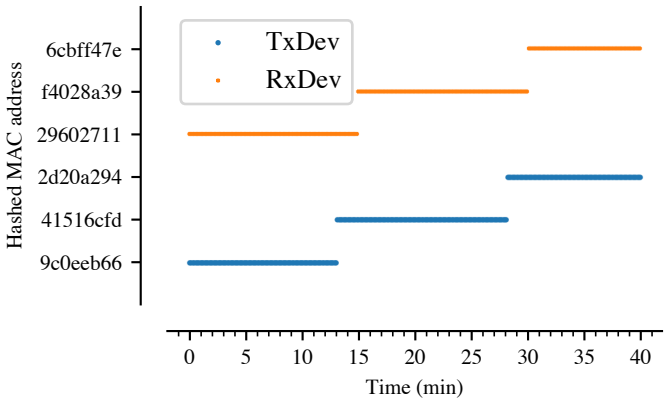


Figure 3.5: Sent and received data frames by two paired BLE devices inside `Simble`

3.2.2.2 Private address resolution

To validate the resolution of private addresses in `Simble`, we consider a simple scenario, where a transmitter and receiver nodes are paired inside it. This allows us to look into global trace obtained by send and receive logs and deduce if the data communication was continuous in spite of the sender and receiver changing their MAC addresses.

As we can see in Figure 3.5, the sender changes its private address around 13 minutes. However, the receiver BLE application continues to process and receive frames as it could resolve the new private address to the sender's Identity Address, having possession of its *IRK*. Similarly, around 32 minutes, we observe that the receiver changes its private address. Still, it is communicated to the sender through beacons, and hence, the sender this time around resolves and verifies the receiver's private address. Therefore, the sender could be seen sending its data to the receiver seamlessly. This experiment thus ensures that `Simble`'s [Alg. 9] is functional in handling BLE MAC randomization.

3.2.2.3 Optimized trace-collection

We discussed in Section 3.2.1.1 the need to optimize the trace-collection procedure to obtain traces in a reasonable time. We validate the improvement

brought by `Simble` in terms of run-time by increasing the density of devices up to 1 device per square meter around a sniffer for a simulation duration of 30 seconds. The density is varied by increasing the number of devices up to 100 in 100 square meters around the sniffer. As we can observe, in Figure 3.6, optimized sniffing gives a performance gain in simulation run-time up to a factor of 100. In conclusion, since we generally have to simulate a considerably longer duration to test BLE privacy provisions as most MAC addresses change around 15 minutes, `Simble` can optimize the sniffing to generate traces in a reasonable amount of time.

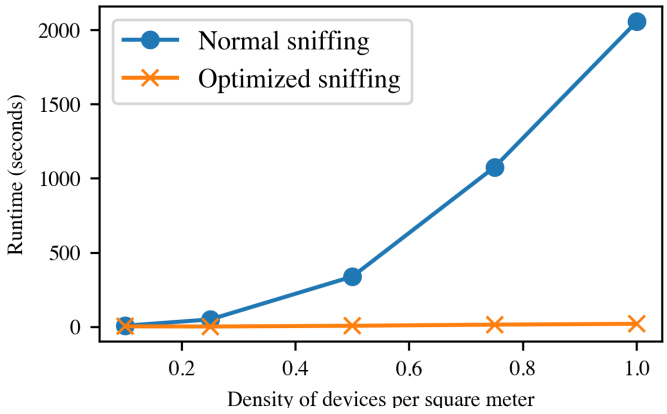


Figure 3.6: Performance gain in run-time with optimized sniffing inside simulation

3.2.3 Using `Simble` to assert BLE flaws

Contrary to the above BLE strategies [42] [13] [43] which target specific devices like Apple, [1] propose a method which associates MAC addresses from a device based on emitted public frames. This makes [6] independent of the device vendor and generic for any BLE device as it just relies on beacons and whatever the used application. They identify devices across time using an identifier that discriminates a subset of devices at a given time, that is, a weak identifier, and achieve close to 100% accuracy for controlled environments. Therefore, *we decided to implement and study performances of [1] when using `SimBle`, since to the best of our knowledge, it is the only generic BLE MAC address association strategy currently available in the literature.* We evaluate it using the traces and the *ground truth* generated by `SimBle`.

The association strategy proposed in [1] involves three steps. Firstly, when analyzing sniffed traces of public BLE frames, the authors identify conflicts where multiple devices change their randomized MAC addresses simultaneously, forming *conflict clusters*. They also define *conflict clusters*

as the time that separates consecutive private addresses from a device. Secondly, they use Linear Assignment with distances between weak identifiers, specifically the characteristic time derived from the fixed part of the time between advertising frames, to associate appearing and disappearing MAC addresses within the *dswap* interval. Finally, they resolve MAC address conflicts to establish associations between randomized addresses. We evaluate the effectiveness of the association strategy with respect to the number of devices, the diversity of devices, and the degree of mobility in the sniffing zone.

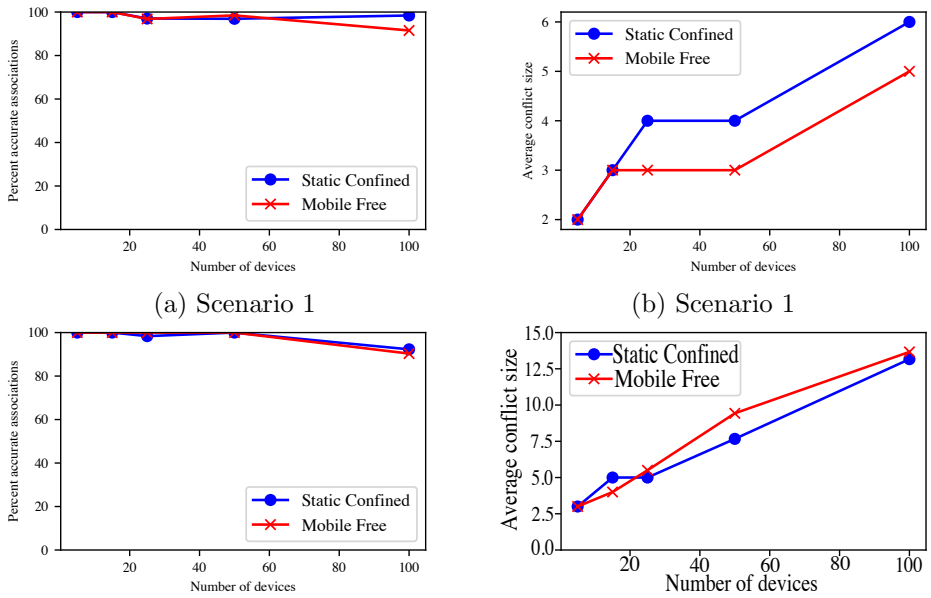
Evaluation: In the following, we evaluate the accuracy of MAC address association and growth of conflict cluster size for various realistic scenarios. In **scenario 1**, we choose *BLE 4.1*, since it is the most prevalent BLE release in devices today. We also choose a *single device class*, which is smartphones. Smartphones largely fall into the device class *moderate emitters* as stated earlier in Section 3.2.1.1. The randomization interval in BLE 4.1 is set to 15 minutes. In **scenario 2**, we consider *BLE 5.2* and *multiple device classes*. Here we emulate a diverse range of devices supporting the latest release, BLE 5.2, in them. We choose this BLE standard because, unlike BLE 4.1, vendors can keep private address generation intervals to be less than 15 minutes. Though standard advises avoiding smaller values for randomization intervals than 15 minutes as it could affect performance due to connection times. We deliberately keep the randomization interval a uniform distribution in the range (3, 15) minutes to observe how [1] performs when more and more vendors start to quicken private address generation. We evaluate all the scenarios for the following mobility profiles:

1. *Static-Confined:* Here the devices are static and are always present in the sniffing zone.

2. *Mobile-Free:* In this profile, devices are mobile and are free to leave and enter the sniffing zone. We try to mimic human mobility by using a random-walk mobility model with a speed of 1.5 m/s and direction change after every 2 s.

We generate all the traces and associated *ground truth* by simulating several BLE devices and a sniffer for 40 minutes using `Simble`. We prefer a longer duration than multiple simulation runs of a small duration as it gives detailed insight into how conflicts evolve with time. It is essential to note how accurately [1] resolves the MAC addresses from a single device in the capture duration. For *Static-Confined* mobility-profile, we place a sniffer in the center of a square of 100 square meters and vary the number of BLE devices/nodes up to 100. We choose this area to make sure that nodes are always in the sniffing range of the sniffer. As shown in Table 3.1, we use the *Nakagmi* path loss model and consider the successful BLE transmission

range to be around 20 meters. While in the case of *Mobile-Free* mobility-profile, we deliberately take a square of 2500 square meters and place the sniffer in the middle of it. BLE nodes are performing random-walk in that area and thus move in and out of the sniffing range.



(a) Scenario 1 (b) Scenario 1

(c) Scenario 2 (d) Scenario 2

Figure 3.7: Accuracy of MAC address associations and average conflict size observed by MAC association strategy [1] on `Simble` generated traces.

Results:

1. **Scenario 1:** First, we observe how well the algorithm [1] can defeat MAC randomization and correctly associate private addresses for BLE 4.1 with *moderate emitters*. MAC addresses change after every 15 minutes in BLE 4.1. For average conflict sizes below 10, we expect the algorithm to perform well both in run-time and accuracy. We observe in Figure 3.7a that the accuracy of association is above 98% for *Static-Confined* mobility-profile. Even in the case of *Mobile-Free* nodes, minimum accuracy of around 91% is seen for 100 devices. Average conflicts increase with an increase in the number of devices as expected in Figure 3.7b, but they are well beneath the bound of 10 conflicts. Hence, the accuracy of MAC address association is very high for both mobility profiles.

2. **Scenario 2:** We go for multiple device classes but with private address changes possible before the interval of 15 minutes. We expect the conflict sizes to rise and hence a decrease in accuracy for a large

number of devices. For 100 devices accuracy of MAC address associations decrease to around 89% for both mobility profiles as seen in Figure 3.7c. Conflict sizes increase to a maximum value of 13 as seen in Figure 3.7d, but it is still not large enough to degrade the efficiency of the association strategy [1].

The case study shows that the current MAC address randomization proposed by the BLE standard is not enough to safeguard user privacy. The association strategy [1] can successfully defeat the randomization procedure and correctly fingerprint close to 90% *of the devices even in highly dense and mobile scenarios. An adversary could set up multiple sniffers strategically and easily track a particular user's device.*

The works that do BLE MAC address association or device-fingerprinting are threats to privacy provisions of BLE [1] [42] [43] [13] as these strategies lead to tracking of users. *Only* `Simble` *can allow the community to compare the effectiveness of any two of these available solutions.* This is because we need exact/identical conditions for comparing the evaluations. It is not only hard for experiments/test beds to emulate identical conditions but are also not scalable. Moreover, as discussed earlier, finding *ground truth* for experimentally obtained traces is practically impossible for large-scale testing. In the following section, we address each of the identified design flaws in detail.

3.3 Asserting flaws in WiFi probe-requests

To assert the flaws in WiFi prone-requests, we use the same methodology as BLE. We focus on assessing the literature attacks' viability on a wide range of datasets while also examining existing MAC randomization methods, and exploring the content of probe-requests in detail.

We dig deeper into asserting WiFi flaws in Chapter 4 where we evaluate two address association frameworks in literature which we refer to as 1) `Infocom2021` [2] and ii) `WiSec16` [3]. The chosen attacks cover all association strategies such as the usage of sequence numbers (SEQ), information elements (IE), packet-reception timings, and RSSI - based signatures. We utilize various data-collection scenarios to see the effective MAC association threats possessed by the current frameworks.

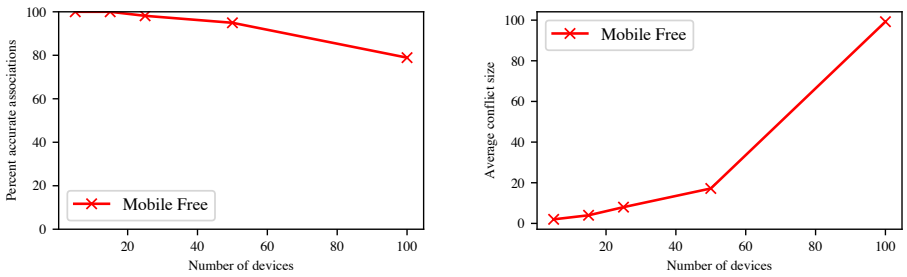
The current frameworks do associate with considerable accuracy though we observe a substantial discrepancy between the performances obtained by these frameworks when confronting them with different contextual environments. As with BLE, where we use `SimBle` to generate passively-sniffed traces with *ground-truth*, we pseudo-randomize publicly available large probe-request datasets by manually changing the device's MAC address every four bursts (cf. Chapter 2.2.2) of probes. We introduce a novel MAC association framework for WiFi in Chapter 5 which indeed achieves

high accuracy in defeating MAC randomization, asserting the flaws that we identify in this chapter.

In the next section, while discussing solutions to mitigate flaws related to WiFi probe-requests, we rather use analytical methods and the available public datasets to assert our arguments.

3.4 Mitigating BLE and WiFi flaws

We address the design flaws concerning current standard provisions that we identify in the previous section and discuss solutions to them, respectively.



(a) Accuracy (b) Average conflict size

Figure 3.8: BLE MAC association [1] with *randomization interval* of 3 minutes

1. Choice of randomization interval: Timing-based attacks on BLE MAC address randomization take the benefit of the current interval after which a device changes the public identifier. The more frequently we perform the randomization, the more probable it is for a higher number of devices in the population to change their MAC addresses around the same time. We denote this number by *Conflict size*. The higher this average value, the more difficult it is to associate MAC addresses from the same device for the adversary. We redefine the notion of conflict size for WiFi MAC randomization too in Chapter 4.

When considering WiFi, there are quite different MAC randomization schemes for Linux, iOS, and Windows [6]. Linux lets the driver or firmware generate per-burst random MAC addresses. In iOS, randomization is limited to probing and only happens when the device is unassociated and in sleep mode. Windows 10 changes the MAC address when the device connects or disconnects from a network and when it restarts. As timing attacks in WiFi also defeat randomization by up to 75% [6], the standard needs to have a consensus on making randomization mandatory for the manufacturers while specifying lower duration, preferably every few bursts.

In BLE, the standard currently recommends keeping the random MAC

address for at least 15 minutes [38]. We look at the performance of [1] with respect to the size of the randomization interval. We use `Simble` for reducing the *randomization interval* of the device population to 3 minutes and evaluate the performance against the standard 15-minute duration. We pick 3 minutes as the optimal lower value, as a much smaller interval will cause longer connection times in real devices. We vary the number of devices inside `Simble` up to 100, for *static-confined* and *mobile-free* mobility profiles.

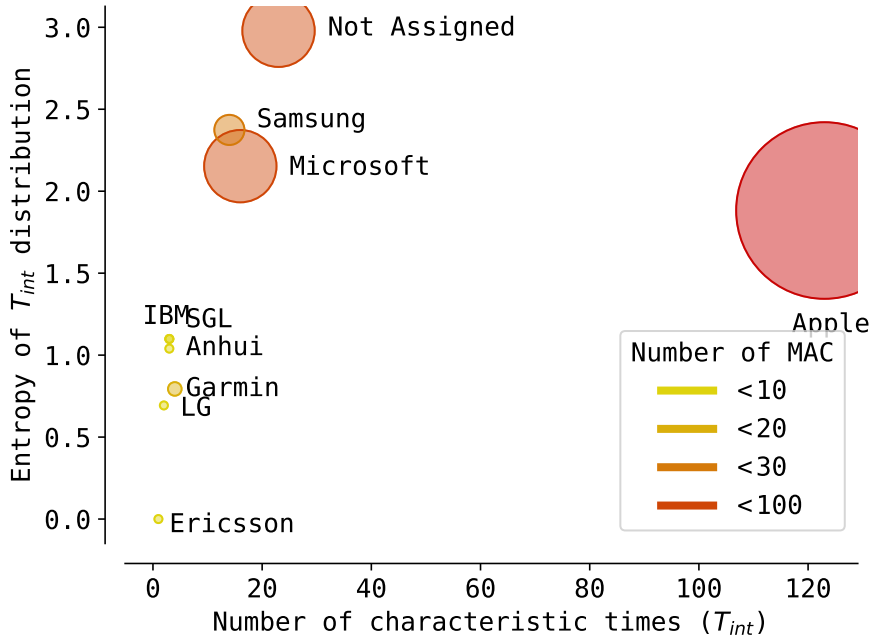
We observe in Figure 3.8 that, indeed, accuracy decreases to a minimum of around 91% to 78% with *conflict size* growing from 6 to 97 when decreasing the randomization interval. Based on this observation, we recommended lowering the BLE *randomization interval* while caring for slightly increased connection times as a consequence. Thus, we can optimize the current IRK (Identity resolving key) [38] exchange, for instance, in BLE, to allow devices to change addresses frequently without compromising performance.

2. Discriminating power of weak identifiers: *Weak identifiers* are the features deduced from the advertised frames which are capable of discriminating a subset of device-population. The majority of the works rely on *timing-based signatures* to differentiate two devices that change their MAC at the same time, as it works as a fingerprint per device [6] [1]. Inter-frame space(IFS) in WiFi and inter-beacon interval in BLE are the most promising weak identifiers that comprise timing-based signatures.

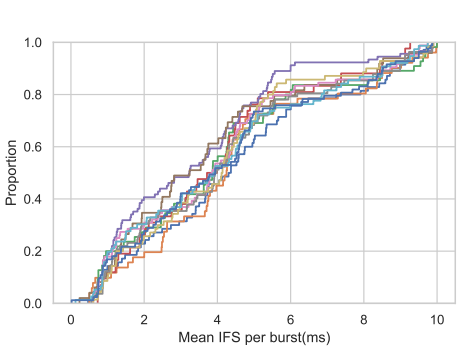
Inter-beacon interval in BLE consists of a constant part plus a pseudo-random value in the range $[0, 10]$ ms. [41, p. 2751] The device regenerates the random value every burst, but the constant part seldom changes and could be estimated [1]. We call in this thesis this weak identifier as the *characteristic time* or T_{int}. We collect a highly dense dataset that contains more than 2500 MAC addresses in less than 10 seconds to evaluate the vulnerability of the current BLE against this identifier. A powerful weak identifier should be as uniform over a set of values as large as possible. The Shannon entropy is a direct measure of the uniformity of the distribution.

Figure 3.9a displays the entropy of the characteristic time distributions, over the space of devices, per brand, as a function of the number of characteristic times used. For each brand, a bigger circle signifies multiple devices of the same brand in conflict more probable. The more characteristic times a brand possesses, the more devices could be simultaneously differentiated. Contrary to the intuitive belief that more identifier means more individual privacy, introducing many characteristic times increases identification chances. With the result in Figure 3.7b we see that, in most cases, the characteristic time is sufficient as a weak identifier, as conflict clusters rarely grow past the 10 devices with current provisions. For such small clusters, [1] already defeating BLE MAC randomization by up to 100%

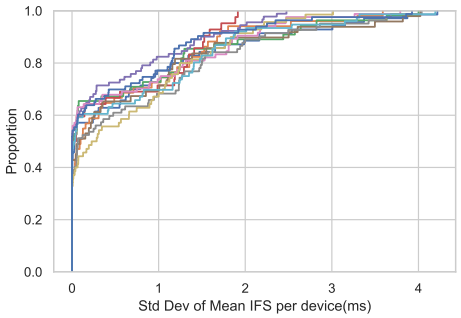
For WiFi, we also investigate the IFS in a probe-request burst using the Bologna probe-request university dataset that captures probe requests from



(a) Discriminatory level of the advertising timing for each device Manufacturer in BLE



(b) CDF of mean IFS per burst for WiFi probe requests



(c) CDF of standard deviation in Mean IFS per burst per device for WiFi probe requests

Figure 3.9: Behavior of *Weak identifiers* (Better seen in color)

3917 MAC addresses. Signatures deduced from the IFS could discriminate up to 75% We see in Figure 3.9b that mean IFS is almost uniformly distributed in the range $[0, 10]$ *ms* for various traces in the dataset. But at the same time, the standard deviation of mean IFS per burst is less than 1 ms for around 80 percent of devices, as seen in Figure 3.9c. This makes means IFS per burst is susceptible to being used as a fingerprinting solution

in WiFi. Again, we recommend the WiFi standard to force manufacturing brands to use similar probe-request bursts in IFS.

3. Confidential packet information fields: To stop the frame field and inference attacks, encryption is the most intuitive solution for limiting device fingerprinting using public frames. We need the following key properties for encrypting public frames: 1) *Universality-* The solution should be compatible across various wireless standards. 2) *Un-correlation-* We have to ensure that two different frames from the device across time are not linked to the same source. 3) *Efficiency-* To save time and energy, a minimum number of exchanges should happen in the control information transfer phase of the proposed protocol. Also, each exchange must happen within realistic bounds to ensure the usability of the control frames. 4) *Conformity-* The structure of the frames must still conform to that of the standard format to hide the presence of security measures from the attacker.

We require an efficient key exchange protocol to establish a symmetric key for further exchanges, while we need different keys for successive frames from a source to ensure the property of un-correlation. We analyze next the need and the feasibility of encryption to ensure confidentiality in BLE and WiFi, respectively. BLE beacons contain very weak identifiers like manufacturer names and event types. We test these identifiers' potential to discriminate among the randomized MAC addresses. We find out that the manufacturer information and the advertisement type can only resolve less than 5 percent of MAC address conflicts seen in the dataset. Therefore, we conclude that it is not necessary to pseudonymize any packet fields in the BLE Beacons.

On the contrary, in WiFi, we have a significant number of attacks on user privacy based on inferring from public frames like probe requests. McKinion et. al [72] do a widespread evaluation of existing frameworks for security pitholes in probe requests. Probe-request-based device identification on IEEE 802.11ac is reduced considerably after the application of stream cipher-based encryption [26]. They test the solution with Two Dell OptiPlex 3600 mini Workstations are used as a client device and AP. We notice that Diffie-hellman key exchange takes $2.4 - 2.6s$, while transmission of the encrypted packet needs $0.4 - 0.6s$ [26].

As we already see in Figure 3.9b, the mean IFS per burst of probe requests in a real dataset is in the order of a few milliseconds. The burst duration is around $10ms$ practically, and we know that most of the devices change their mac addresses after every few bursts. Active scans in WiFi are meant for fast re-connection to known networks. The overhead of $500ms$ in sending encrypted probes, even in basic exchanges of a stream cipher, is not realistic. The protocols will get heavier if we introduce more security guarantees to stop replay attacks, for instance. We should also consider the packet losses during the key exchange and the timeouts that it would

induce. Moreover, we must support the broadcast probe requests from the client too. Due to these resource constraints, we argue the un-feasibility of classical encryption as a contender for providing confidentiality in WiFi probes.

Instead, we propose the exploration of the following solutions in the future to ensure the practicality and usability of WiFi probe requests:

1. Introducing controlled noise to the information fields in the probe requests can effectively diminish the success rate of fingerprinting attacks.

2. Replacing the entries in Preferred Network Lists (PNLs) advertised by the probe requests with pseudo-identifiers, agreed upon by each client-AP pair beforehand, presents another viable option.

3.5 Conclusion

In this chapter, we focus on investigating potential vulnerabilities, particularly in the context of MAC address association, which involves linking randomized MAC addresses in beacons and probe-requests to their sender devices. This association poses a considerable threat to user privacy.

We delve into this problem, thoroughly exploring its implications and highlighting the pressing need for privacy-preserving public frames. We identify crucial flaws in the current design of public frames, which contribute to privacy issues requiring immediate attention from the community. By addressing these flaws and discussing potential solutions, we aim to empower devices to better safeguard user privacy. We present our recommendations based on the findings.

In literature, the majority of association frameworks utilize the identified flaws to associate MAC addresses. Before identifying more potential vulnerabilities and proposing a better-performing association framework, we need to assess the current frameworks and their accuracy with respect to varying input datasets. In the next chapters, we focus on WiFi probe-requests datasets for MAC association for two reasons. First, the BLE device's MAC addresses have already been shown to be associated with very high accuracy in literature [1]. Second, effective MAC association is relatively more challenging in WiFi than BLE due to the higher density of WiFi devices in public spaces, implying a larger number of observed randomized public frames.

Chapter 4

Benchmarks for association frameworks

Contents

MAC randomization is a backbone for preserving user privacy in WiFi, as devices change their identifiers (MAC addresses). MAC association frameworks in the literature are able to associate randomized MAC addresses with a device. Such frameworks facilitate the continuation and validity of works based on device-based identifiers.

In this chapter, we first question and verify the *reliability* of MAC association frameworks with respect to the datasets (scenarios) used for their validation. We define the *reliability* as the ability of a MAC association framework to accurately identify MAC addresses belonging to the same device independently of the contextual scenarios described in the testing datasets.

Indeed, we observe a substantial discrepancy between the performances obtained by association frameworks when confronting them with different contextual environments, revealing their unreliability. We identify the device heterogeneity in the input scenario as the cause of varying accuracy obtained by association frameworks. Henceforth, we propose a novel metric: *randomization complexity*, capable of successfully catching the *degree of randomization* in evaluated datasets. Existing and new frameworks can thus

be *benchmarked* using this metric to ensure their reliability for any datasets with similar or lower randomization complexities. Finally, we open discussions on the potential impact of the benchmarks in the domain of MAC randomization.

4.1 Identifying frameworks unreliability

We follow a four-step methodology for establishing and addressing the reliance on current address association frameworks:

1. **Detecting the unreliability:** First, we detect and verify the unreliability of current association frameworks when tested with different contextual environments. We carefully select frameworks to evaluate in this thesis along with the input datasets to represent various real-world passive sniffing scenarios. The datasets contain varying degrees (*complexities*) of MAC address changes in small intervals of time, potentially challenging the association accuracy of chosen frameworks. (cf. Section 4.1.1)

2. **Reasoning the unreliability:** Second, we investigate the reason for the varying performance of association frameworks. We find the factors which cause the heterogeneity in input datasets causing the variance and degradation of performances. The probe-request generation in terms of timings and content in the dataset causes the difference in *complexities* that an association framework faces. (cf. Section 4.1.3)

3. **Introducing the benchmarks:** Third, we find a generic metric that could capture the effects of all the identified factors in the previous step. We observe that indeed, the *complexities* induced by an input probe-request trace can be modeled and used as *benchmarks* when evaluating an association framework. (cf. Section 4.3)

4. **Showcasing the effectiveness of benchmarks:** Finally, we showcase the potency of benchmarks in capturing the effectiveness of a framework. We show that the accuracy behavior of frameworks across input datasets follows the trends shown by the corresponding benchmarks. Benchmarks allow frameworks' *reliability* verification providing insights into their association behavior (cf. Section 4.3.2).

4.1.1 Case-Studies

We consider two literature works as case studies for testing the resilience of address association frameworks, i.e., (i) `Infocom2021` [2] and (ii) `WiSec16` [3], which are validated upon a set of heterogeneous datasets: *HongKong* (cf. Chapter 2.5) and probe-request collection inside a laboratory. The case

studies cover all association strategies: usage of sequence numbers (SEQ), information elements (IE), packet-reception timings, and RSSI - based signatures.

`Infocom2021` **case-study:** Using diverse scenarios, we evaluate two major components of the signature introduced by authors in [2]: the IE and the SEQ. We do not consider the third component: RSSI, as it requires multiple geographically-known sniffers around each transmitting device while contributing very less (drops up to 10%) to the effectiveness of the signature (Fig. 13, [2]). The effectiveness metric utilized in `Infocom2021` is *discrimination accuracy*. Authors define discrimination accuracy as the ratio of the correct association, estimated from 1000 randomly selected probes (P_i) and their previous probes in the *Period* ($[t_i - \tau, t_i]$). They vary the period between (0, 600s).

`WiSec16` **case-study:** Similarly, we proceed with the performance analysis of the timing-based signature introduced in [3], under different scenarios. They propose distance metrics based on timing (inter-frame arrival time) and use them together with an incremental learning algorithm to group probes. It claims up to 75% partially randomized traces collected in a laboratory environment.

Lack of comparability: `Infocom2021` uses probe-requests captured inside a shopping mall for evaluation by deploying multiple sensors. It consists of around 5000 unique MACs per hour. However, only the devices transmitting their true (physical) MAC address are considered for evaluating the framework. On the other hand, authors evaluate `WiSec16` on a dataset collected inside a laboratory consisting of probe-requests with non-randomized addresses. They partially randomized their dataset (100 out of 550 captured devices) by manually changing MAC addresses in four bursts (cf. Section 4.2.1) of probe-requests. Both frameworks analyze distinct datasets, and for a meaningful comparison between the two case studies, it is essential to evaluate them using the same input probe-request dataset.

4.1.2 Need for benchmarks

We need a *ground truth* of randomized MAC addresses emitted from the same device to compute the association accuracy of the chosen frameworks . Hence, we use the public anonymized *Sapienza* trace [1], consisting of about 11 million probe-requests, passively collected in eight different contextual scenarios. We choose datasets from five scenarios: *trainstation*, *themall*, *vatican 1*, and, *vatican 2*. The first two showcase highly frequented public spaces,

[1] `https://crawdad.org/sapienza/probe-requests/20130910`

while the remaining denote dense outdoor environments. The dataset consists of the true MAC addresses of devices. We randomize the dataset by changing the addresses every 4 bursts while considering various percentages of captured MAC addresses and keeping the *ground truth* of original addresses. Next, we look at the performance of both frameworks with respect to varied scenarios of this dataset.

Regarding `Infocom2021`, we can observe in Fig. 4.1a and 4.1b that discrimination accuracy of IE and SEQ-based signatures vary significantly across input datasets. For instance, we can see that in relatively more static environments (*train-station*) the performances are good for small periods. However, when the environment starts to be more dynamic the performances fall considerably. The accuracy of IE in *vatican2* drops to 23 percent while it goes close to zero when considering probes in the period of 600s. The reported discrimination accuracy for the same period, in `Infocom2021` is 44% and 19%, for IE and SEQ-based signatures, respectively. This shows that the parameter period highly impacts the results in all scenarios.

Considering `WiSec16`, we notice in Fig. 4.1c that the proposed association framework performs variably to the difference in scenarios and the degree of MAC randomization present in the collected probe-requests trace. Also, the achieved accuracy is significantly lower than the claimed best performance (75%) in fully randomized datasets.

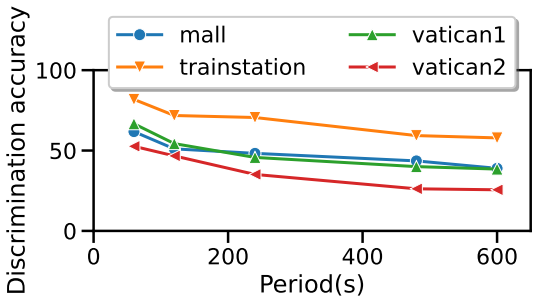
Awareness requirement: We observe that the performance of the studied association frameworks is considerably variable. We argue this variability induces unreliability in the obtained accuracy. Such observation leads us to the essential need for phenomena inference in input datasets, i.e., contextual inference of scenarios.

4.1.3 Causes of unreliability

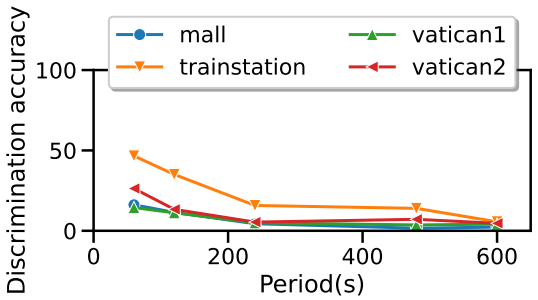
We identify a major issue in the validation of current association frameworks: **a lack of *characterization* of the *heterogeneity*** in terms of the burst's and the randomization's behavior (cf. Sec. 4.2.1) in utilized probe-request datasets. The *model* and the *user-usage patterns* of a device introduces this heterogeneity that influences the generation of probe-request packets (cf. Sec. 4.2). The *model* of devices denote the class of devices along with its manufacturer (e.g., smart watches, mobile phones with Android, iOS) while the *usage patterns* of users refer to the *mode* of the device when emitting a probe-request (e.g., screen ON/OFF).

The reason for the variation in the probe-request sending rates and the MAC randomization is that every manufacturer independently decides these behaviors to ensure a good trade-off between the network experience for users and the device's performance (e.g., battery life).

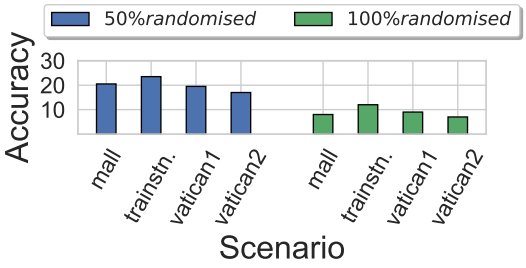
The main issues we identify are as follows:



(a) *Information elements (IE)*



(b) *Sequence numbers (SEQ)*



(c)

Figure 4.1: Case studies: (a-b) `Infocom2021` [2], (c) `WiSec16` [3].

1. The literature does not use the same dataset (to ensure *homogeneity*) or any kind of *benchmark* to show the trustworthiness of the obtained framework's performance (cf. Sec. 4.3).

2. There is a general lack of *ground truth* of randomized MAC addresses with respect to the sending device. The first issue causes looking only at a *selective view* to the framework's performance. In contrast, the second issue leads to the usage of indirect accuracy metrics such as discrimination accuracy (as in `Infocom2021`).

3. There is a lack of analysis of the performance with respect to varying *intensity* of MAC address changes that a data-collection scenario can capture (cf. Sec. 4.3.2). The more the *intensity* of address swaps, the more difficult, in general, it is for a framework to associate MACs.

The need: We claim the need for a metric to *characterize* the device *heterogeneity* in terms of the burst's and randomization's behaviors. Hence, we introduce the *randomisation complexity* to bring the notion of *benchmarks* for validating the reliability of association frameworks.

4.2 Impact of heterogeneity

We investigate the variability of MAC randomization in WiFi probe-requests with respect to *heterogeneity* introduced by the pool of current mobile devices. The three identified factors making the emission of probe-requests heterogeneous and subsequently having an impact on the address randomization in WiFi are: i) the device's *model*, ii) the device's *mode*, and, iii) the transmission *channel*. While the *model* captures the device *heterogeneity* with respect to randomization behavior, we observe that the *mode* is the one that has the highest impact on the probe-request generation. Hence, in the following, we will focus mainly on the device *mode* for characterizing probe-requests.

The probe-requests are susceptible to fingerprinting due to: 1) the nature of their *bursts* and 2) the randomizing strategy of their MAC addresses. In the following, we evaluate probe-request susceptibility with respect to heterogeneity introduced by the device's *models*, *modes*, and, transmission *channels*.

4.2.1 Probe-request bursts

We can not analyze probe-requests behavior using *Sapienza* datasets used for the case studies (cf. Sec. 4.1.1) as we need a wide range of current mobile devices with additional information such as device's *mode* when transmitting. Therefore, we used the probe-request dataset [5]. This dataset has 22 popular device models in practice, which were sniffed when present in various device *modes* (see Tab. 2.2). This is the first open-source probe-requests dataset, *labelled* with the *ground truth* of randomized addresses. It contains 20-minute duration captures of known devices using a Raspberry Pi-based sniffer.

There are active-screen modes (A, PA, and WA) and inactive-screen modes (S, PS, and WS). In power-saving modes (PA and PS), the device additionally keeps the power-saving setting active, while in WA and WS modes, the device also has the Wi-Fi interface switched off. Each device configuration is observed in the three non-overlapping channels (1,6, and,

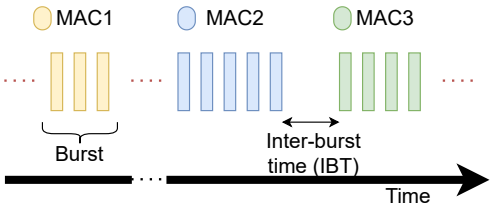


Figure 4.2: A device's randomised probe-requests.

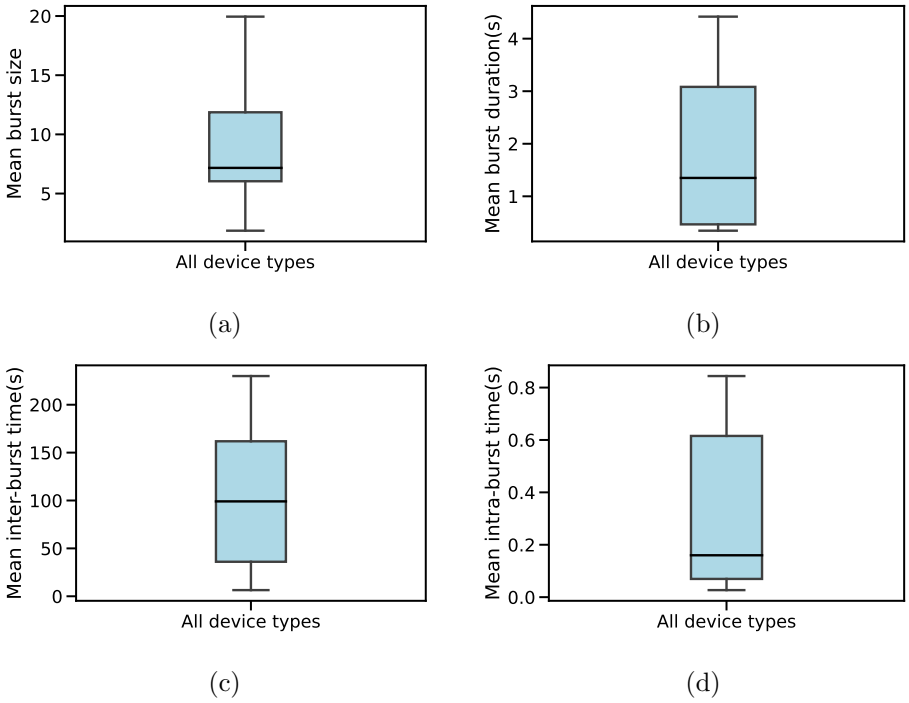


Figure 4.3: Analysis of probe-requests burst behaviors – i.e., size, duration, Inter-burst time, and Intra-burst time – under varying *Device model's*.

11) of the 2.4 GHz frequency band. The details of capture conditions for each device mode can be found in [5]. In the following, we discuss the metrics concerning Wi-Fi probe-request emissions that facilitate the association.

Fig. 4.2 illustrates the probe-requests sent from a device over time in the active scanning process. Devices send probes on every available channel to discover available networks in proximity. Each probing round consists of a *burst* of frames. The burst size is variable, depending mainly on the number of available channels. The period between successive bursts, inter-burst time (IBT), is variable to devise models and its operating *modes*.

In essence, the temporal behavior of WiFi probe-requests can be char-

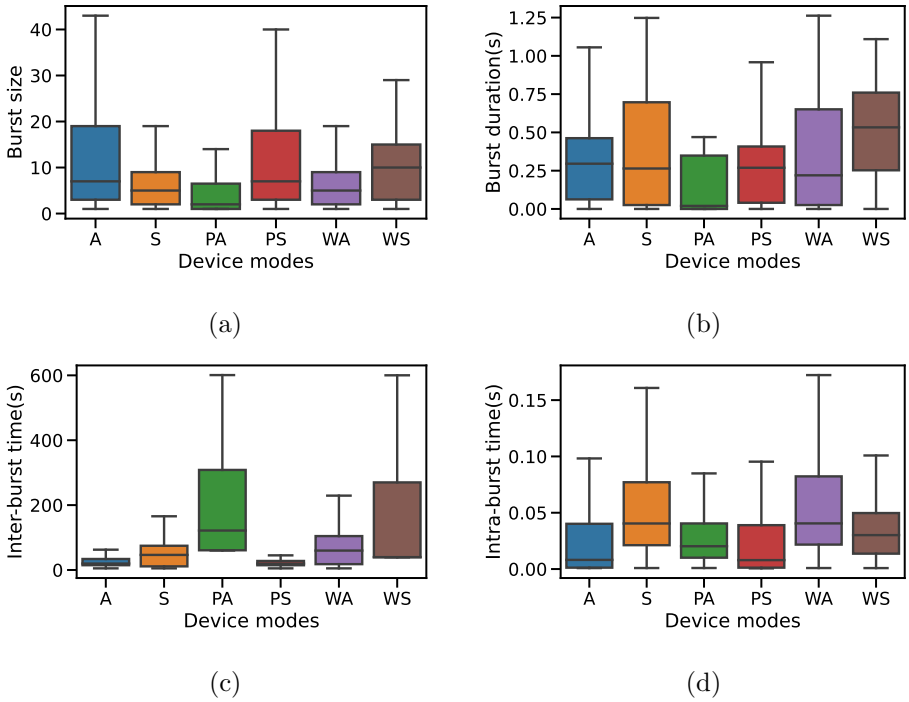


Figure 4.4: Analysis of probe-requests burst behaviors – i.e., size, duration, Inter-burst time, and Intra-burst time – under varying *Device modes*.

acterized by four metrics:

Bursts' size: The average number of frames in a burst when considering all device types (models) varies considerably with the mean value of 7 (see Fig. 4.3a). We further analyze a device's behavior per mode in Fig. 4.4a. We observe that devices send many frames in modes (A and PS). Intuitively, devices reduce the burst sizes to the minimum in the mode with active-screen and power saving mode (PA).

Bursts' duration: The average duration of a probe-request burst, across device types varies around $1s - 3s$, as shown in Fig. 4.3b. It is interesting to note in Fig. 4.4b that burst duration is relatively lower in active-screen modes (A and PA) than in static-screen modes (S and WS). This can be attributed to devices conserving energy when under constraints.

Inter-burst time (IBT): The inter-burst time is typical to devise manufacturers' factory configurations. Hence, across all metrics we consider to characterize probe-requests, it shows the most variation across the device types. Fig. 4.3c illustrates that IBT alters from a few seconds to more than 200 seconds. Moreover, we notice in Fig. 4.4c that IBT is the lowest in active-screen mode (A) while the highest when the power-saving mode is

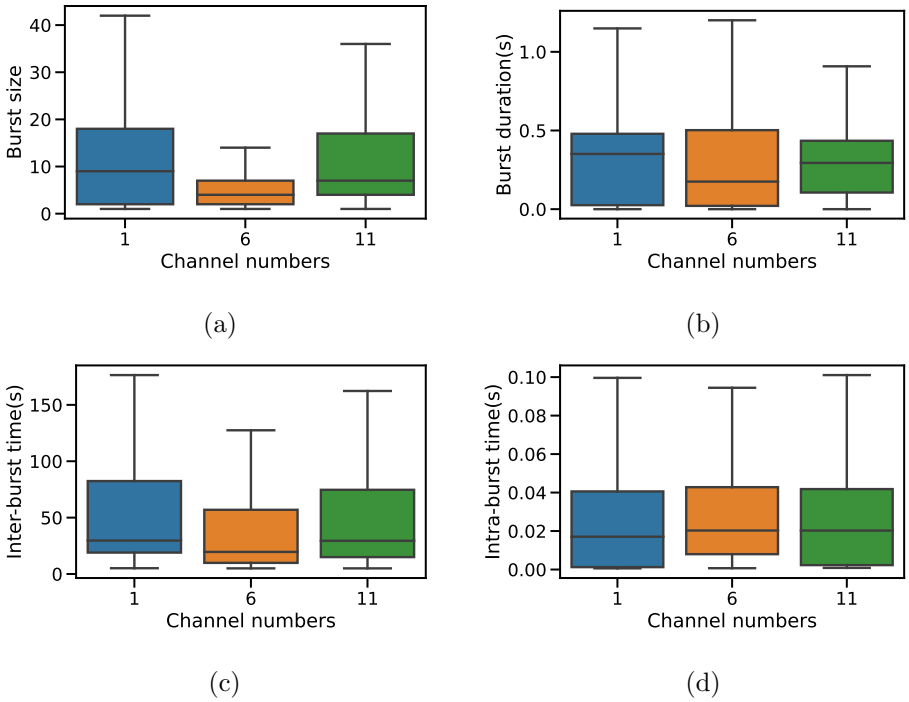


Figure 4.5: Analysis of probe-requests burst behaviors – i.e., size, duration, Inter-burst time, and Intra-burst time – under varying *Channels*.

additionally turned ON (PA).

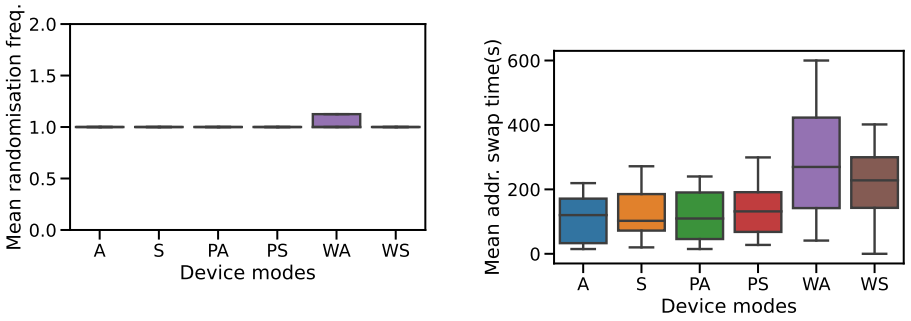
Intra-burst time: Finally, we investigate the time between frames in a single burst. We observe in Fig. 4.3d that it goes up to $0.8s$. When looking at the effect of device modes on the intra-burst time, we could notice in Fig. 4.4d that devices tend to send frames quickly in a burst in the active-screen mode (A) than in the static-screen mode (S).

We also observe a slight variation in the burst's size and duration as well as IBT and intra-burst time with respect to transmission channels (see Fig. 4.5a - 4.5d).

4.2.2 Randomization behavior of WiFi MAC addresses

In addition to the amount of probe-requests generated by WiFi devices, the temporal patterns of advertised MAC addresses also impact the accuracy of association frameworks. In WiFi, most of the current devices change their MAC identifiers after a period of time. We notice that 18 of 22 considered major device types in the dataset [5] do randomize their MAC addresses. Fig. 4.6a illustrates that most devices change their MAC address after a burst, irrespective of device modes. Moreover, the average address swap times are low and almost similar across device modes, except when the

WiFi interface is switched OFF (WA and WS), as shown in Fig. 4.6b.



(a) Mean randomization frequency (num. of bursts)

(b) Mean address swap time

Figure 4.6: Behaviour of MAC addresses

We have two key conclusions:

- **1st conclusion:** The device *heterogeneity* does play a major role in WiFi MAC address randomization. We could see that all broad types of signatures are based on metrics that vary with device *types* and *modes*. Signatures based on temporal information of probe-requests [3] rely upon the behavior of bursts, while those based on frame fields [2] depend upon device-specific frame fields. Finally, sequence numbers-based signatures [2,27] also subsequently depend upon the burst's size and IBT of individual devices, as it dictates the probability of sniffers recording a device's frames.

- **2nd conclusion:** MAC association frameworks need a representative evaluation. As we just discussed, we must *catch* device's population (models) as well as modes present at the time of dataset collection. We have to *ensure* that this dataset used for the evaluation of an address association framework should maintain the performance in *similar* scenarios. Hence, we need a metric that quantifies this *similarity* between data collection scenarios.

In the following section, we introduce *randomization complexity*, a metric that enables us to compare two datasets with respect to the *difficulty* that a framework has to face while associating contained MAC addresses. This metric hence allows the evaluation of various association frameworks to be comparable.

4.3 Benchmarks for association frameworks

We build upon our conclusions to introduce the formalization on *benchmarks* for the evaluation of MAC association frameworks. The proposed benchmarks are *generic* and are *representative* of the *complexity* in the used input dataset.

We believe such formalization: (i) Brings new ways of interpreting devices' randomization behavior, (ii) Opens paths for designing adaptive randomization techniques by the standard. For instance, devices can randomize more frequently in situations with sparse nearby device populations to maintain a high *complexity* for an adversary, and, (iii) And off-course allow association frameworks to be more robust and reliable.

4.3.1 Defining benchmarks

To associate observed randomized MAC addresses to respective sender devices, correct mapping of address pairs representing the same user has to be performed. The adversary has to choose among multiple possible MAC pairs seen during the same time interval. We utilize this intuition to define MAC association as the resolution of *conflicts*.

Conflicts ($\mathcal{C}$) refer to changing MAC addresses in a given time period. They are either caused by devices that stop emitting during the MAC changing process or due to their entry/exit from the sniffing range. We refer to this time period as *conflict period* (T_c). We illustrate conflict periods in Fig. 4.7. Dotted lines in different colors represent different appearing and disappearing MAC addresses sent by devices, in a T_c.

Any address association framework in essence has to resolve these conflicts and perform correct assignment between the disappearing and appearing MAC from individual devices. Each MAC address M is a bunch of probe-requests with a start (M^{start}) and a stop (M^{stop}). For instance, a disappearing MAC, M_j is said to be in conflict (illustrated in Fig. 4.7) with an appearing MAC, M_k if:

$$\mathcal{C} : M_k, M_j \mapsto (i-1)T_c < M_j^{stop}, M_k^{start} \leq iT_c \tag{4.1}$$

We obtain benchmarks from the following three steps:

1. Determining the *conflict period*: We denote the set of modes as $\mathcal{M}$, the set of device models $\mathcal{D}$, and the transmitting channels (frequency bands) as $\mathcal{F}$. The conflict period (T_c) should be such that we let the association framework consider possible associations. We observe in Sec. 4.2.2 that a device is likely to change its MAC after a burst. Hence, we must consider T_c to be at least of the inter-burst time (IBT).

First, we vary the device types (d_j) and channels (f) while keeping the mode fixed to obtain the min., max. and, avg. values of IBT per

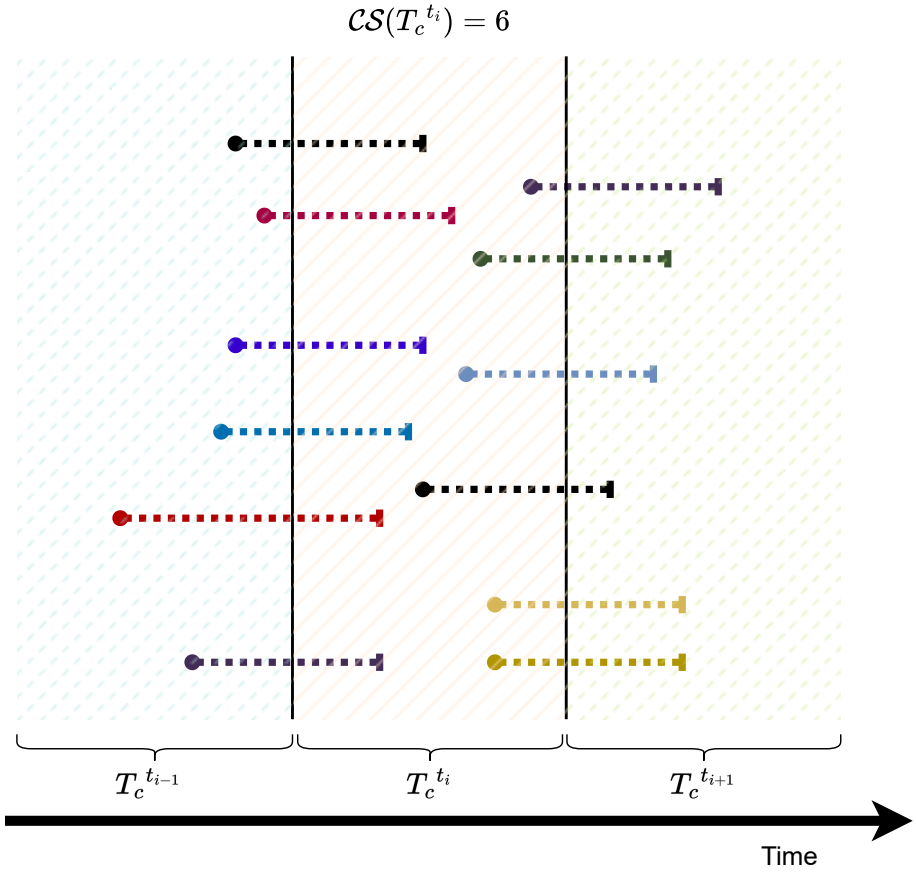


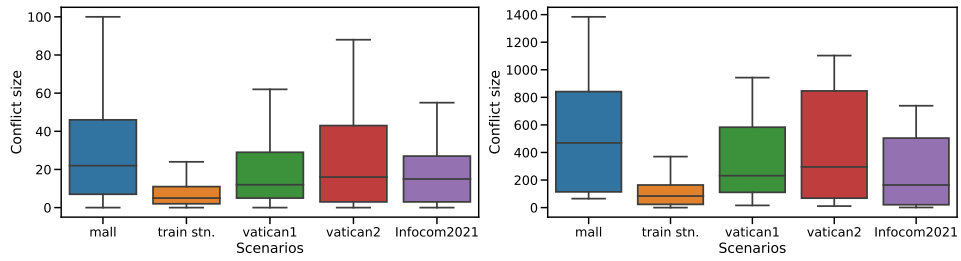
Figure 4.7: An illustration of conflict periods (T_c)

mode (IBT_m) [Eq. 4.2]. Then, we vary modes and take the corresponding min., max. and, avg. of the union set, to obtain the T_c's minimum, maximum, and, average values [Eq. 4.3]. We find the $T_c^{(min.,avg.,max.)}$ to be $(5.00s, 95.72s, 365.42s)$ using the *labelled* dataset [5].

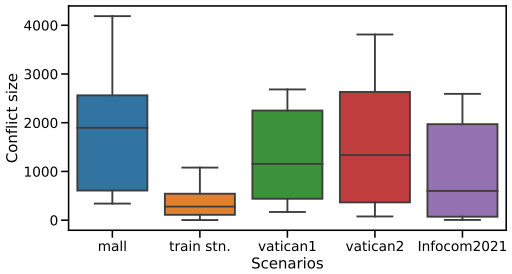
$$IBT_m{}^{(min.,avg.,max.)} = (min., avg., max.) \bigcup_{d_j \in \mathcal{D}, f \in \mathcal{F}} IBT_{(d_j,f)_m} \tag{4.2}$$

$$T_c^{(min.,avg.,max.)} = (min., avg., max.) \bigcup_{m \in \mathcal{M}} IBT_m{}^{(min.,avg.,max.)} \tag{4.3}$$

2. Obtaining the *conflict size*: We introduce a global metric: *conflict size* ($\mathcal{CS}$), which is capable of catching the "complexity" of any input network trace. We define conflict size as half the number of address trails (M^{start}'s and M^{stop}'s), in a conflict period (T_c). For instance, as shown in Fig. 4.7, the conflict size ($\mathcal{CS}$) is 6 for $T_c^{t_i}$. Higher $\mathcal{CS}$ makes it difficult for frameworks to *resolve* conflicts.



(a) Min. randomisation complexity (b) Avg. randomisation complexity



(c) Max. randomisation complexity

Figure 4.8: Randomisation complexities

3. Inferring the *randomization complexity*: We define the corresponding minimum, maximum, and, average: randomization complexities ($\mathcal{RC}^{(min.,avg.,max.)}$) as the set of conflict sizes ($\mathcal{CS}(T_c^{(min.,avg.,max.)})$) when considering corresponding conflict periods (t_i), overall slots ($\mathcal{S}$) in the considered dataset [Eq. 4.4].

$$\mathcal{RC}^{(min.,avg.,max.)} = \bigcup_{i \in \mathcal{S}} \mathcal{CS}(T_c^{(min.,avg.,max.)})^{t_i} \tag{4.4}$$

The randomization complexities act as a *benchmark* for the lower, upper, and, average performance limits of address association frameworks. The calculated values of $T_c^{(min.,avg.,max.)}$ in this thesis can be used to obtain randomization complexities of any input dataset by just following steps (2) and (3) above. New frameworks in literature, can accompany their frameworks with our benchmarks to ensure their *reliability* for any new scenarios with similar or lower complexities.

4.3.2 Showcasing benchmarks

In Fig. 4.8, we report the min., max., and avg. values in complexity of `Infocom2021` dataset and of the *Sapienza* datasets. We notice that the randomization complexity and the framework's performance are inversely

proportional, as expected. We observe that scenarios: *vatican2* and *mall* have relatively higher complexities than the other ones.

When considering `Infocom2021`, we can observe in Fig. 4.1a and 4.1b that the discrimination accuracy obtained by both signature metrics: IE and SEQ, degrade with increasing randomization complexities. The scenario *trainstation* for instance, obtains the best relative performance as it exhibits the lowest randomization complexity in Fig. 4.8. We observe a similar trend of the high and low accuracy in `WiSec16`, for the scenarios: *trainstation* and *vatican2* as shown in Fig. 4.1c. These two scenarios show relatively lower and greater randomization complexities respectively, as expected.

This certifies our *benchmarks* of *adequately* catching the *complexity* of the dataset. Both frameworks are expected to have the same association accuracy when evaluated with another input dataset with similar randomization complexity.

4.4 Conclusion

MAC association frameworks in the literature have demonstrated the ability to associate randomized MAC addresses with specific devices, which raises concerns about the effectiveness of privacy preservation. In this chapter, we question and verify the reliability of MAC association frameworks concerning the datasets used for their validation. The reliability of an address association framework is defined as its ability to achieve consistent accuracy across varying contexts where the input probe-requests datasets were collected. Through our analysis, we have observed a significant discrepancy in the performances of these frameworks when confronted with different contextual environments. We propose a novel metric that effectively captures the degree of randomization in evaluated datasets. This metric can be used to benchmark existing and new frameworks, ensuring their reliability for datasets with similar or lower randomization complexities.

Benchmarks play a crucial role in pinpointing weaknesses and gaps in the existing literature, enabling the design of a new association framework that effectively selects and combines signatures. In the following chapter, we present our novel MAC association framework, called `Bleach`, which achieves notably high accuracy across various input probe-request trace collections, bridging the reliability gap in the current literature. We also discuss the benchmarks introduced in this chapter concerning our new framework.

Chapter 5

Bleach framework

Contents

Having identified and addressed the unreliable nature of current MAC association frameworks, we introduce a new framework that obtains high association accuracy for a large range of input datasets. The effectiveness and generality of utilized signatures must be carefully investigated and validated to ensure reliability. The address association framework should be able to correctly associate randomized addresses even in scenarios where there is a large and varying number of simultaneous MAC changes/arrival observed in the input probe-request trace.

In this chapter, we first propose the `Bleach` framework to overcome the limitations of current frameworks that we highlighted in the last chapter. `Bleach` utilizes effective time and frame content-based signatures and employs a novel MAC association algorithm based upon a logistic regression predictor. We finally evaluate our framework , present our results, and give

a generic understanding of MAC address-association accuracy. We benchmark the framework with respect to the input datasets (cf. Chapter 4) and show that we obtain better performance than the literature for varied contextual scenarios.

5.1 Bleach framework

The framework Bleach takes probe-request trace with randomized MAC addresses as input and yields a dictionary ($\mathcal{A}$) of randomized addresses (M_j) associated with particular devices (U_n). $\mathcal{A}$ can be represented as:

$$\mathcal{A} = \{U_1(M_i, M_j, ..., M_k), \ldots, U_n(M_a, M_b, ..., M_z)\}$$

It consists of **four** major steps as shown in Figure 5.1.

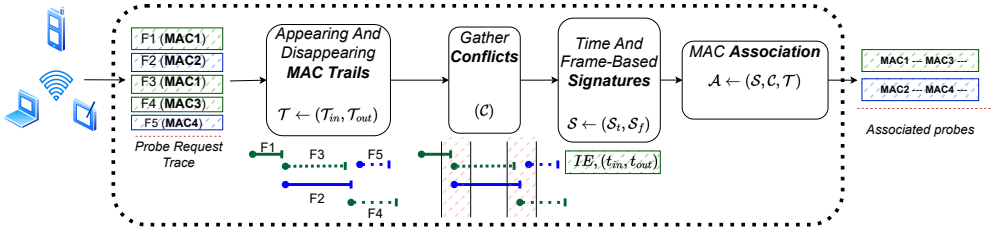


Figure 5.1: Bleach framework.

1. We transform the input probe-request trace into a set of MAC address trails. Each MAC trail can be viewed as an instance of the appearance or the disappearance of a MAC address in the sniffing zone. This reduces the problem of MAC association to that of correctly associating each disappearing MAC trail from a device with an appearing trail from the same device. We detail the process of MAC trail generation in Section 5.2.

2. We separate the trails into disjoint subsets comprising *conflicts* ($\mathcal{C}$) (cf. Sec. 5.3). The conflict denotes the set from which a disappearing MAC trail could be possibly associated, with any of the appearing MAC trails present in the dataset, within a period ($T_c^{\tau_i}$) from the end of the disappearing trail. We identify this period as the conflict period. The right value of the conflict period allows us to consider all potential associations while making the decision of linking the MAC address trail pairs.

   Conflicts are either caused by devices that change their MAC addresses or by their entry/exit from the sniffing range. Any address association framework, in essence, has to resolve conflicts to perform correct

assignments between the disappearing and appearing MAC from individual devices. After obtaining conflicts of MAC address changes and a generic formulation of the MAC association problem, we take a step further toward the association itself. We need to obtain effective signatures for resolving conflicting MAC address trails.

3. We define and extract the time and frame-based signatures ($\mathcal{S}_t, \mathcal{S}_f$) from the collected MAC trails (cf. Sec. 5.4). We consider two types of signatures in this paper: i) *time-based signatures*, which utilize the information from the temporal behavior of received probe-request frames, and ii) *frame-based signatures*, which use the control field information present in the captured frame itself to form effective signatures that have the potential of discriminating a device from the rest of the population.

4. We introduce a novel MAC association algorithm capable of resolving the conflicts observed in the input dataset accurately. It uses extracted signatures ($\mathcal{S}$) to fingerprint and differentiates randomized MACs in each conflict duration in order to finally associate them (cf. Sec. 5.5)

The following sections detail each of the above-mentioned steps of the `Bleach` framework.

5.2 Step 1: Extracting MAC trails

We divide the input dataset into MAC address trails, tr^j. A MAC address trail (cf. Sec. 5.3 and 5.5) comprises a group of probe-requests sent from a device with a particular MAC. For each MAC address (M_j) seen in the dataset, we extract two trails from it, as illustrated in Figure 5.2. One denotes the start of the M_j which we label as an appearing trail (tr^j_{in}), while the other showcases the end of the advertisement of M_j, which we name as a disappearing MAC trail (tr^j_{out}).

Trails of both natures though contain the same bursts (b_n) of probe-requests emitted by the device, with the MAC address as M_j as described in Equation 5.1. Each burst contains varying number of probe-requests (p_m), i.e. $b_n = \{p_1, p_2, ..., p_m\}$.

$$tr^j_{in}, tr^j_{out} = \{b_1, b_2, ..., b_n\} \tag{5.1}$$

We consider the appearing MAC trail for association at timestamp $(tr^j_{in})^{start}$, while we consider the disappearing trail for subsequent association at the timestamp $(tr^j_{out})^{stop}$ as shown in Figure 5.2.

This distinction in the nature of trails eases the formulation of the address association problem by simplifying it into the correct matching of each disappearing MAC trail (with address M_j) to an appearing MAC trail (with

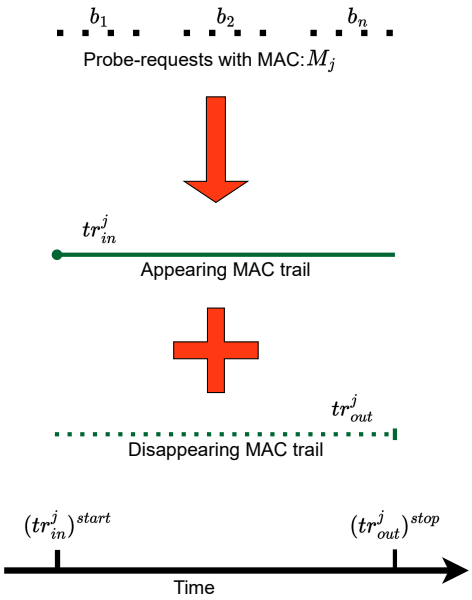


Figure 5.2: Breaking probe-request sequences (with address M_j) into MAC trails

address M_k). Each trail additionally has its characteristic features describing its temporal characteristics (transmission duration, frequency of probes e.t.c.), its nature, and subsequently, the information about composing the signatures from probe-request groups. We denote the set of appearing trails in the dataset as $\mathcal{T}_{in}$ and the set of disappearing trails as $\mathcal{T}_{out}$.

5.3 Step 2: Obtaining conflicts

After the preliminary step of our framework `Bleach`, we have a set of appearing and disappearing MAC trails from the input dataset. In the second step, `Bleach` identifies MAC association as the resolution of *conflicts* (also hinted in Chapter 4.3.1). A preliminary idea of MAC conflicts in the context of BLE is also discussed by [1]. We redefine it comprehensively with respect to WiFi probe-requests. Next, we describe the characteristics of conflicts and the methods to obtain them.

MAC *conflicts*: For each disappearing MAC trail (tr^j_{out}), we denote a time period ($T_c^{\tau_i}$) starting from the end of tr^j_{out}, called as *conflict periods* ($T_c^{\tau_i}$). We illustrate conflict periods in Figure5.3 where dotted lines in different colors represent different appearing and disappearing MAC address trails of devices in a $T_c^{\tau_i}$.

Formally, we define a *conflict* (cf. Figure 5.3) between a disappearing MAC trail, tr^j_{out} and an appearing MAC, tr^k_{in}, if the two trails satisfy the

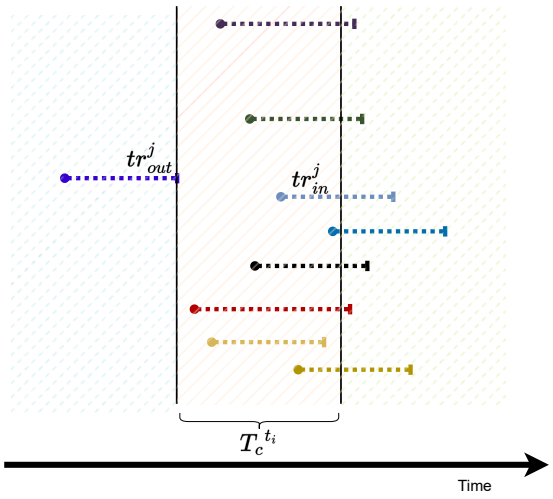


Figure 5.3: An illustration of conflict periods ($T_c^{\tau_i}$)

condition mentioned in Equation 5.2.

$$\mathcal{C} : tr^k_{in}, tr^j_{out} \mapsto (T_c^{\tau_i})^{begin} < (tr^j_{out})^{stop}, (tr^k_{in})^{start} \leq (T_c^{\tau_i})^{end} \tag{5.2}$$

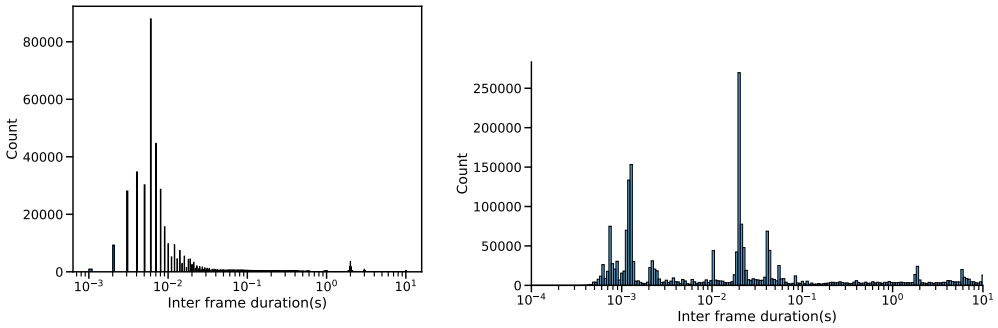
Here $(tr^k_{in})^{start}$ and $(tr^j_{out})^{stop}$ are the start and stop timestamps of the trails tr^k_{in} and tr^j_{out}. $(T_c^{\tau_i})^{begin}$ and $(T_c^{\tau_i})^{end}$ are the beginning and end timestamps of a particular conflict period $T_c^{\tau_i}$.

As we already know, each MAC trail (tr^j) consists of bursts of probe-requests. In order to isolate individual bursts from respective devices, we investigate burst-related parameters. Isolating and investigating individual and adjacent bursts is critical in choosing the right value of conflict period, $T_c^{\tau_i}$ as MAC addresses from a single device only change on a new burst of probe-requests. A small value of $T_c^{\tau_i}$ will cause `Bleach` to miss a potential correct association of a disappearing MAC trail to the appearing one as the new burst will start after the chosen $T_c^{\tau_i}$. A very large value would mean considering unnecessary associations, as it is highly unrealistic for the duration between two consecutive bursts from a device to be too big. These unnecessary associations lead to higher time complexity of the framework.

Choosing burst parameters: We identify two burst-related parameters: i) Burst duration (t_b) and ii) Conflict period ($T_c^{\tau_i}$). Knowing the burst duration is critical as each of the trails of MAC addresses (tr^j) in Figure 5.3 represents a sequence of bursts, with each burst containing two or more frames. Isolating bursts eventually help in developing signatures too (cf. Sec. 5.4.1.1). On the other hand, $T_c^{\tau_i}$ allows isolating conflicts that `Bleach` has to consider in associating a particular disappearing MAC address that has potentially been randomized.

We look at the histogram of inter-frame duration (IFS), observed in

captured frames of *HongKong* and *Sapienza* datasets. IFS is the time difference between two consecutive probe-requests sent by a particular device, as observed by the sniffer.



(a) (b)

Figure 5.4: Inter-frame duration(s) (IFS) in: Accumulation of all devices in a) HongKong dataset, b) Sapienza datasets

In Figure 5.4a, we observe two clear peaks in IFS bin counts. The first peak is in the order of milliseconds, while the second small peak is in the order of seconds. Similarly, in Figure 5.4b, which is the accumulation of all scenarios in Sapienza datasets, we observe two distinct peaks in the order of milliseconds and a few small peaks in the order of seconds. This is expected as probe-requests are sent in bursts by devices across various channels in the hope of getting a response from the nearby access point.

Indeed, IFS inside bursts are expected to be small as they happen consecutively, while the IFS denoting a new burst from the same device denotes a new probing round, which generally happens after a certain period of time (a few seconds). Hence, the frames within a duration of less than 1 second are likely to be part of a single burst, as suggested by the major peaks. All the frames with IFS lying between the period of 1 to 10s consist of the inter-bursts times (IBT), as shown by smaller peaks.

Henceforth, we choose the burst duration (t_b) to be 1s. The conflict period (T_c) is set to be 10s, as this allows the MAC address changes (that happen with a new burst from a device) to be noticed inside a conflict ($\mathcal{C}$)

5.4 Step 3: Obtaining signatures

Signatures ($\mathcal{S}$) are deductions from exhibited characteristics of a device or entity, which allows isolating it from the rest of the population. We propose and use two signatures extracted from captured probe-requests to associate randomized MAC addresses from a device.

In the following, we first present our choice of signatures for associating

randomized WiFi MAC addresses inside `Bleach` framework. Then, we proceed to present details for computing the chosen signatures. Finally, we end the section by justifying the choice of considered signatures.

5.4.1 Chosen signatures

We choose i) **Time-based** signatures and ii) **Frame-based** signatures for our association framework. Time-based signatures utilize the timing-related information obtained from the frame reception by a sniffer from respective devices. These signatures are effective choices as they are generic and independent of the device type.

We combine the time-based signatures with the frame-based signatures. Frame-based signatures supplement the cases where the timing information from the frames is not representative of a device due to fewer probes or the high variability in timing information per user device. Next, we discuss the choice and effectiveness of these two signatures in detail.

5.4.1.1 Time-based signature

We already illustrate the behavior of IFS in Figure 5.4 when analyzing probe-request bursts. The properties of a burst could be extracted that are unique for an observed device in the dataset. We choose the timing information: mean inter-frame time (IFS) across individual probe-request bursts as the time-based signature ($\mathcal{S}_t$) for the device advertising a particular MAC address. Mean IFS is the average interval between subsequent probe-request frames received from a device inside a burst while considering all the bursts from that device. The idea is that the frequency of sending probes during an active scan of networks is likely to differ across devices while remaining unique for the same device. Hence,

$$\mathcal{S}_t = \mu^{IFS}$$

For the calculation of μ^{IFS} in a MAC trail, we take the mean of IFS values inside a burst while considering all observed probe-request bursts in the trail.

5.4.1.2 Frame-based signature

For frame-based signatures, we investigate the information elements (IE) [Section 9.4.2.1, [73]] contained inside a probe-request frame. This field depicts the abilities of the sending device, which is used for its negotiation with the access point. There are multiple IE fields referred to by their Element IDs which range from 0 to 255 [73]. We take a look at around 500,000 frames from the HongKong dataset and investigate specific capabilities advertised by the probe-requests as a part of IE.

The inclusion of Information Elements (IEs) within the probe request is not obligatory, but they are necessary for specifying the supported functionalities of the device. Each device could send all the IEs or only a subset of them, depending upon the context where the WiFi device is situated, the manufacturer, etc. We take the maximum occurring elements of IE in the probes that we investigate. The top 8 most probable metrics that are likely to be consistent in terms of presence are shown in Tab. 5.1. Hence, we select frame-based signature as:

$$\mathcal{S}_f = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$$

We investigate the potential of this signature in being discriminative in Section 5.4.3.

Name	IE element	Percent occurrence
e_1	SSID	100
e_2	Supported Rates & BSS Membership	100
e_3	Extended Supported Rates	99.51
e_4	HT Capabilities	82.60
e_5	Vendor Specific	62.56
e_6	Extended Capabilities	54.52
e_7	Interworking	13.5
e_8	VHT Capabilities	2.43

Table 5.1: Most frequent IE elements

5.4.2 Computing MAC trail signatures

After obtaining the formulations for time and frame-based signatures, we proceed to finally give details for computing them for each MAC address trail in the input dataset (cf. Algorithm 2). In Alg. 2, we illustrate the application of the first step of `Bleach` too for completeness.

We first isolate/group probe-requests per MAC (M_j) to collect all the individual probe-requests bursts to advertise that address. Grouping into bursts takes into account the burst duration (t_b) that we calculated earlier (cf. Sec. 5.3). For each burst group with MAC M_j, we add an instance of appearing and disappearing MAC trails in ($\mathcal{T}_{in}$) and ($\mathcal{T}_{out}$) respectively.

For each MAC trail in tr_{in} and tr_{out}, we randomly select a representative frame for that trail (f_{in} and f_{out}). We use f_{in} and f_{out} for calculating the frame-based signatures ($\mathcal{S}_f^{f_{in}}$ and $\mathcal{S}_f^{f_{out}}$) of the considered MAC trail.

We finally obtain trail signatures ($\mathcal{S}[tr_{in}]$ and $\mathcal{S}[tr_{out}]$) as a tuple comprising of frame-based signatures and the mean inter-frame space (μ^{IFS}) of the considered appearing (tr_{in}) and disappearing trail (tr_{out}) respectively.

Algorithm 2 Computing signatures

procedure ComputeSignatures($t_b, \mathcal{S}_f$) ▷ input variables
  $\mathcal{B} \leftarrow \phi$ // Dictionary of probe bursts
  $\mathcal{S} \leftarrow \phi$ // Dictionary of signatures
  $\mathcal{T}_{in} \leftarrow \phi$ // Appearing MAC trail
  $\mathcal{T}_{out} \leftarrow \phi$ // Disappearing MAC trail
  for $M_j \leftarrow \Sigma$ **do**
    $\mathcal{P} \leftarrow GroupProbes(\Sigma, M_j, t_b)$
    $\mathcal{T}_{in}, \mathcal{T}_{out} \leftarrow TrailMACs(\mathcal{P})$
    $\mathcal{B}[M_j] \leftarrow \mathcal{P}$
  for $tr_{in}, tr_{out} \leftarrow \mathcal{T}_{in}, \mathcal{T}_{out}$ **do**
    $f_{in}, f_{out} \leftarrow RandSamples(tr_{in}), RandSamples(tr_{out}))$
    $\mathcal{S}[tr_{in}] \leftarrow (\mathcal{S}_f^{f_{in}}, \mu^{IFS}{}_{tr_{in}})$
    $\mathcal{S}[tr_{out}] \leftarrow (\mathcal{S}_f^{f_{out}}, \mu^{IFS}{}_{tr_{out}})$
  return $\mathcal{S}$

5.4.3 Evaluating chosen signatures

We have to formulate the *effectiveness* of a signature to ensure that the association is likely to be the correct one. The two factors that we identify as generic indicators for a signature's performance are: i) *Consistency* and ii) *Discriminating power*.

Consistency measures the ability of a signature to be uniform for a single entity, across multiple instances of itself in the population. In our case, the population is the set of WiFi devices emitting probe-requests while a single entity is a particular WiFi device. The multiple instances are multiple probe-requests/probe-request bursts with randomized MACs from the same device. The intuition is that the signature should not be volatile for a single device itself in the first place, and should be able to ideally associate all MACs from a device. Hence, a high *consistency* value is essential for an effective signature.

Once a signature validates *consistency* per device, the second factor we should complement it with is the *discriminating power*. It implies that the signature values should be variable across devices in the dataset. Ideally, the larger the size of the range from which the device's signature exhibit its values, the higher the chances of it to be correctly associating randomized MAC addresses among those in the population. Multiple devices with similar signature values are likely to lower the accuracy with which a signature correctly associates addresses.

Time-based signatures: We illustrate the *consistency* of the time-based signatures. We first compute the signatures for each burst of probe-requests by an individual device with MAC, M_j in the collected trace. We normalize

the signature values between 0 and 1. Finally, the *consistency* in time-based signatures, $\mathcal{CS}$ for M_j is defined as:

$$\mathcal{CS}^{M_j} = 1 - \sigma(\frac{\mathcal{S}_t}{maximum(\mathcal{S}_t)}) \tag{5.3}$$

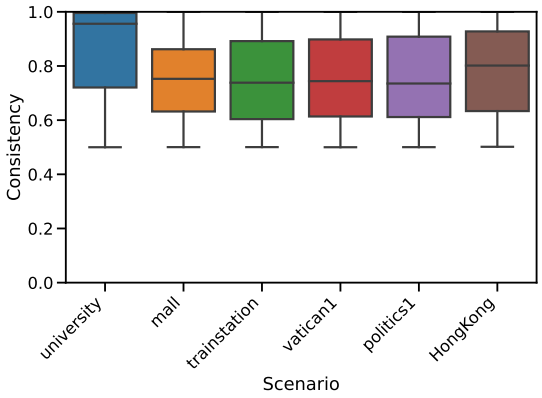


Figure 5.5: *Consistency* in time-based signatures

We look at the range of consistencies shown by observed MACs in multiple datasets. Figure 5.5 shows the results for the *consistency* of chosen time-based signatures. We could observe that $\mathcal{S}_t$ demonstrates high *consistency* in each of the scenarios. On average, the *consistency* is greater than 75%, and up to 100% for MAC addresses in the datasets. The stability of $\mathcal{S}_t$ across bursts from the same datasets is essential for it to be considered as an effective signature. We observe that for all scenarios, we achieve a high consistency, enforcing the stability of $\mathcal{S}_t$.

To finalize the mean IFS as the time-based signature, we also check its *discrimination power*. We have a look at the difference between mean IFS for each pair of MAC address pairs observed in various datasets. We observe in Figure 5.6 that difference in mean IFS takes a wide range of values in the interval (0, 0.2) seconds. This ensures the high discrimination power of $\mathcal{S}_t$ as a signature. Finally, the last observation is that the Mean IFS inside a burst is device-specific and similar across various datasets.

The mean IFS has a high *consistency* with respect to a particular device while is variable over a large range of values when considering different devices. This affirms the ability of the signature to discriminate the MAC from a device from the rest of the population.

Frame-based signatures: To compare multi-dimentional frame-based signatures ($\mathcal{S}_f$), we define a similarity metric ($\mathcal{Z}$) which demonstrates and validates its *Consistency* and the *Discriminating power*. For two MAC addresses emitted from devices A and B and their respective frame-based sig-

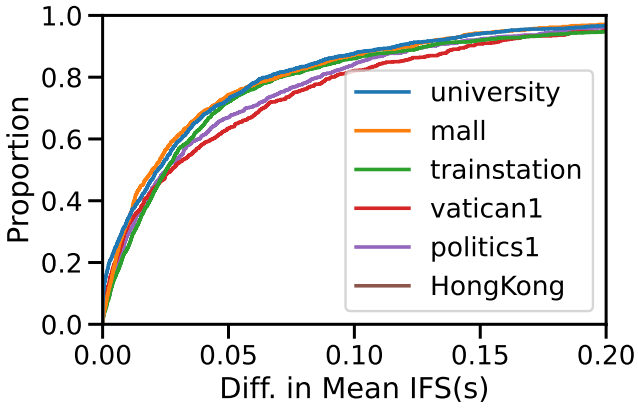


Figure 5.6: Difference in mean IFS.

natures, $\mathcal{S}_f^A$ and $\mathcal{S}_f^B$ the similarity, $\mathcal{Z}$ is:

$$Z(\mathcal{S}_f^A, \mathcal{S}_f^B) = \sum_{i=1}^{8} isEqual(\mathcal{S}_f^A[i], \mathcal{S}_f^B[i]) \tag{5.4}$$

The function $isEqual$ checks if the corresponding elements of either signature are equal and are not absent (ϕ). If this is satisfied, it returns 1, else 0. Intuitively, $Z(\mathcal{S}_f^A, \mathcal{S}_f^B)$ indicates the extent of similar elements transmitted by both devices.

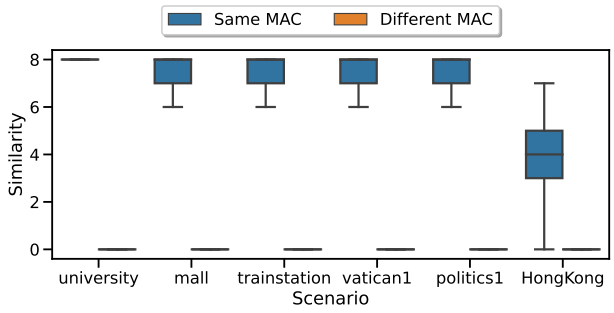


Figure 5.7: Similarity between frame-based signatures.

We investigate the similarity across a large number probe-requests pairs transmitting the same and different MAC addresses while considering frames transmitting their real MACs in the Sapienza and HongKong datasets. We look at the distribution of $Z(\mathcal{S}_f^A, \mathcal{S}_f^B)$ for both the cases in Figure 5.7. We observe that the similarity is very high for probes from the same device (MAC), while it is practically zero for different MACs. HongKong dataset

has relatively diverse values for the same MACs as Sapienza scenarios due to the absence of certain IE fields in some of the frames. The absence leads to the highest attainable value of similarity as lower than 8 for some frames.

There is a considerable gap in similarities between a potential true and false association by the signature, demonstrating the high *Discriminating power* of $\mathcal{S}_f$. The higher the gap, the easier it is for the signature to distinguish between the true and the false associations. Moreover, signatures from the same MAC, or in this case, the device, shows a high degree of similarity. This also showcases the high *consistency* of $\mathcal{S}_f$, hence validating its effectiveness.

5.5 Step 4: MAC Association

We utilize Algorithm 4 to associate randomized probe-request addresses. It takes as input the set of appearing and disappearing trails along with the obtained collection of signatures ($\mathcal{S}$). Algorithm 4 yields a dictionary of MAC pairs ($\mathcal{A}$) , denoting the associated randomized addresses. The association relies on the correctness of predictions for accurate associations when considering pairs of disappearing and appearing MAC trails.

We start with no associated MAC addresses. We sort the appearing and disappearing trails in time so that we could match each disappearing trail with a corresponding appearing trail if its the same advertising device ($\mathcal{T}_{in}{}', \mathcal{T}_{out}{}'$). We also keep track of associated appearing trails that are already paired in order to avoid comparing them again while resolving another conflict ($associated$).

Logistic regression predictor: We opted to utilize a logistic regression model for predicting the degree to which a potential MAC trail pair represents a correct association. To train this model, we combined frame and time signatures as features. The first feature, denoted as $f1$, is computed by determining the similarity between the representative frames of the conflicting MAC address trail pair using Equation 5.4. We perform this process for each possible pair of disappearing and appearing MAC trails observed in the training dataset ($\mathcal{S}_f^{f_{in}}, \mathcal{S}_f^{f_{out}}$). The second feature, denoted as $f2$, is derived by calculating the absolute difference between the mean Inter-Frame Spacing (IFS) periods observed in the trail pair.

The logistic regression predictor is likely to be efficient as we only have a couple of features with the two classes (true and false associations) distinctly different due to the high discriminative power of both features (cf. Section 5.4.3).

Resolving randomized MACs: We examine each disappearing trail (tr_{out}) one by one from the set of trails that have been previously sorted in chronological order ($\mathcal{T}_{out}^{'}$). To ensure the significance of the signatures, we filter out trails that are too short by considering only those with at least 4 frames

Algorithm 4 Address association

procedure AddressAssociation($\mathcal{S}, \mathcal{T}_{in}, \mathcal{T}_{out}$) ▷ input variables
  $\mathcal{A} \leftarrow \phi$
  $\mathcal{T}_{in}{}', \mathcal{T}_{out}{}' \leftarrow TimeSort(\mathcal{T}_{in}), TimeSort(\mathcal{T}_{out})$
  $associated \leftarrow [False] x length(\mathcal{T}_{in}')$

  $f1 = Similarity(\mathcal{S}_f^{f_{in}}, \mathcal{S}_f^{f_{out}})$
  $f2 = |\mu^{IFS}{}_{tr_{in}} - \mu^{IFS}{}_{tr_{out}}|$
  $\mathcal{L} \leftarrow LogisticRegression((f1, f2))$

  for $tr_{out} \leftarrow \mathcal{T}_{out}{}'$ **do**
    if $tr_{out}.length > MIN\_TRAIL\_LENGTH$ **then**
      $\mathcal{C} \leftarrow Conflicts(\mathcal{T}_{in}{}', tr_{out}, T_c^{\tau_i})$
      $\mathcal{V} \leftarrow \phi$
      for $ctrail \leftarrow \mathcal{C}$ **do**
        $fvect \leftarrow LogisticFeatures(tr_{out}, ctrail)$
        $\mathcal{V} \leftarrow PredictionProb(\mathcal{L}, fvect)$
      $\mathcal{V}' \leftarrow Sort(\mathcal{V})$
      for $ctrail \leftarrow \mathcal{C}$ **do**
        $dseq \leftarrow SeqNumGap(tr_{out}, ctrail)$
        if $dseq < SEQ\_TH$ & associated[ctrail] !=True **then**
          $\mathcal{A} \leftarrow (tr_{out}, ctrail)$
          $associated[ctrail] = True$
          $ExitTheLoop()$
  return $\mathcal{A}$

(MIN_TRAIL_LENGTH). Subsequently, we identify the set of appearing trails that conflict ($\mathcal{C}$) during the duration ($T_c^{\tau_i}$) following the disappearance of the considered MAC trail, tr_{out}.

For each conflict trail, we obtain the corresponding feature vector ($fvect$) to derive the prediction probabilities from the trained logistic regression model, $\mathcal{L}$. This yields a probability vector of the size of conflicts ($\mathcal{V}$), indicating the likelihood of the MAC trail pairs being transmitted from the same device. We then sort this vector in descending order of probabilities to select the best feasible match.

While it is plausible that some associations might involve a new device in the sniffing zone rather than a randomized MAC from a previously seen device, we present a methodology to address this issue. For each conflicting upcoming trail ($ctrail$), we calculate the gap in sequence numbers ($dseq$) between this trail and the disappearing trail under consideration. To avoid erroneous comparisons, we also establish a threshold for this sequence num-

ber gap (SEQ_TH), which plays a crucial role in the association algorithm.

To determine the appropriate threshold, we examine the gap in sequence numbers among all the MAC trails observed in various datasets, as depicted in Figure 5.8. We find that approximately 85% of the trails have a sequence number gap of less than 64. We opt to set the value of SEQ_TH to 64.

Regarding larger gaps between 64 and 4095, the proportion of sequence numbers in Figure 5.8 gradually and uniformly increases between 64 and 4095, possibly indicating the re-entry of the device into the sniffing zone after missing a significant number of consecutive bursts. Moreover, numerous new devices emerge in the dataset, with their first frame having a sequence number randomly distributed within the range [0, 4095]. Hence, we ignore sequence number gaps from 64 onwards in `Bleach`.

At last, we can proceed with the final step of the MAC association algorithm, which involves linking a newly detected randomized MAC trail to a previously seen one. If we encounter a conflicting appearing trail (considered based on their prediction probabilities) that meets the sequence number threshold and hasn't been associated before, we label this MAC as associated and exit the loop to continue with the next disappearing MAC. If none of the conflicting appearing MACs meet the SEQ_TH criterion, we assume it to be a disappearing MAC, representing the last trail observed by that device in the sniffing zone.

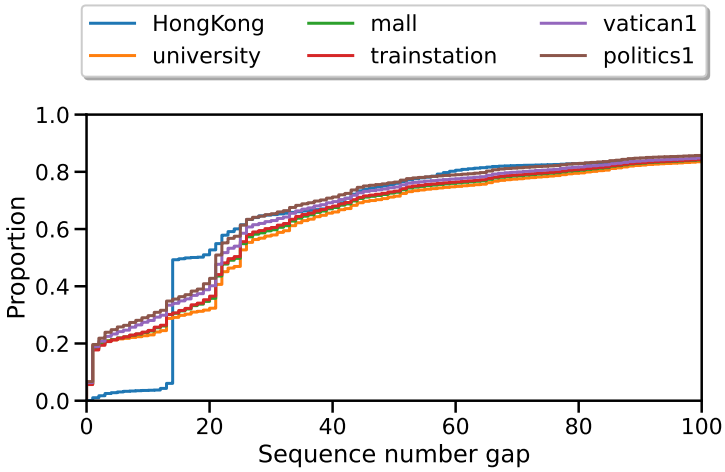


Figure 5.8: Sequence number gap between MAC trails.

5.6 Evaluation

In this section, we present the evaluation methodology utilized for `Bleach` framework before presenting the evaluation of its effectiveness in associating randomized WiFi MAC addresses.

5.6.1 Evaluation methodology

We first investigate the efficiency of chosen signatures that were used as features to train the logistic regression classifier. Then, we proceed to assess the MAC association capabilities of `Bleach`.

To evaluate the `Bleach`'s association performance, we use a variety of datasets. The first dataset is where we have a *ground-truth* of MACs from the same device. These datasets are part of the Sapienza collection and comprise scenarios like *university*, *mall*, *trainstation*, *vatican1*, and *politics1* (cf. Chapter 2.5). After validating the framework `Bleach` with ground-truth datasets, we utilize the HongKong dataset, which consists of capturing randomized MACs of devices in a shopping Mall using a large number of sniffers. This dataset is dense and contains both devices that transmit their true (non-randomized) MAC addresses and randomized MACs. We associate the randomized MAC of HongKong dataset, thus generalizing the performance of `Bleach` to the cases with no *ground-truth* of random MAC addresses from the same sender.

5.6.2 Performance of signatures

Since the base of our association framework is the logistic regression classifier trained with features comprising of the time and the frame-based signature, the first step of the evaluation process is to evaluate the performances of such signatures.

Training/test datasets: We train the logistic regression model over the two features, using the Sapienza datasets due to the access of *gound-truth*. For obtaining the *ground-truth*, we manually randomize the Sapienza datasets by grouping the MAC addresses per device into a sequence of bursts using the burst duration (t_b). We assign new unique identifiers to a device after every 4 bursts. We opt for the same number of bursts per MAC address as in literature [3] to keep a *ground-truth* of appearing and disappearing trails in the dataset. We isolate positive (true association) and negative (false association) MAC pairs to eventually train the logistic regression model ($\mathcal{L}$).

We train the model on *university* and *mall* scenarios and observe the accuracy of the classifier on test sets comprising of the remaining three datasets: *trainstation*, *vatican1*, and, *politics1*. IE fields and the mean IFS in the frame and time-based signatures are device-specific and hence are not heavily dependent on the choice of training scenarios. We chose 50k random

MAC trails from each dataset for the test. The accuracy depicts the model's effectiveness in correctly separating the true and false associations among the respective disappearing and appearing MAC trails.
Evaluation metrics: We use three metrics to look at the performance on the test set: i) Precision, ii) Recall, and iii) F1-score. Precision is the ratio between the True Positives and all the positives, while Recall shows the proportion of actual positives that were identified correctly. F1-score is the Harmonic mean of the Precision and Recall.

Case	Precision	Recall	F1-score	Dataset
False association (negative)	0.79	0.65	0.71	trainstation
	0.99	0.67	0.80	vatican1
	0.91	0.61	0.73	politics1
True association (positive)	0.70	0.82	0.76	trainstation
	0.75	0.99	0.86	vatican1
	0.70	0.94	0.81	politics1

Table 5.2: Performance of signatures

Evaluation results: We observe in Tab. 5.2 that we achieve an F1-score up to 86% with a minimum of 71%. This certifies relatively high Precision and Recall achieved by our signature-based logistic regression classifier, both in false and true associations. It shows that the model produces a lower number of false positives and negatives, demonstrating its effectiveness. The accuracy of association across a dataset could vary depending on the number of MAC addresses in a conflict that Alg. 4 has to resolve. We next discuss this in detail.

5.6.3 Datasets with ground-truth

In Tab. 5.3, we illustrate the accuracy of association obtained in different contextual scenarios of Sapienza datasets . We define accuracy as the percent of correct association of the disappearing trail with respect to the total number of disappearing trails that `Bleach` considered for the address resolution. Considering different scenarios helps the framework to be robust against i) a variety of mobile devices with specific temporal behavior of probe-request bursts, ii) high densities of mobile devices around the sniffer, and iii) diverse address randomization strategies by the manufacturer.

We observe that the accuracy of association is variable across datasets, demonstrating the heterogeneity that we expect each of them to possess. In *university* scenario, we resolve close to 99% of randomized address trails, while the train station to exhibits a high accuracy of close to 95 %. Even, the highly dense outdoor setting of Vatican city square (*vatican1*) achieves a modest accuracy of around 75%. Finally, the majorly indoor scenario of *mall* and political meeting hall obtains relatively low accuracy of around 61

Scenario	Accuracy	Scenario	Accuracy
university	99.14	trainstation	94.82%
mall	60.89	vatican1	74.80%
politics1	69.14		

Table 5.3: Accuracy in Sapienza datasets.

and 69%, respectively. We next explore and reason the performance variability for `Bleach` and, in general, any association framework in detail.

Interpreting association accuracy: In the following, we aim to understand the heterogeneity of datasets as an input to MAC association frameworks, which cause the fluctuation in performance in time and across scenarios. As we propose and illustrate in Sec. 5.3, MAC association can be abstracted into the resolution of address conflicts $\mathcal{C}$. $\mathcal{C}$ showcase all possible appearing MAC trails that could be associated with disappearing ones during various conflict periods $T_c^{\tau_i}$ of the input dataset. The size of conflicts, $|\mathcal{C}(T_c^{\tau_i})|$ in the dataset captures the *complexity* that an association framework has to face.

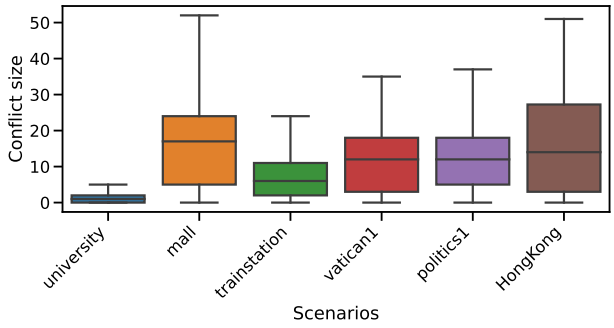


Figure 5.9: Conflict sizes of datasets.

$|\mathcal{C}(T_c^{\tau_i})|$ acts as a generic metric that captures various phenomena that could potentially affect the performance of address association like: i) Inter-arrival times of probe-requests, ii) Mobility patterns of users across the capturing sniffers, iii) Heterogeneity of hardware (mobile devices), and, iv) State of devices transmitting probe-requests (like idle screen, WiFi switched off, power-saving mode on, number of known access points) [74].

While lower inter-arrival times of frames at sniffer are likely to inflate the conflict size, short-term stay of the mobile device or repeated entry-exit in the sniffing zone will make the $|\mathcal{C}(T_c^{\tau_i})|$ high and volatile. This induces errors in the association as resolution means successfully isolating correct MAC

trails among a large number of possible pairs. Similarly, various datasets could have differences in the kinds of mobile devices and their state during the probe-request collection. These factors affect the frequency and pattern of transmitted probes, leading to variable conflict sizes faced by the resolution framework (cf. Section 4.2). Instead of looking at individual phenomena, conflict sizes act as a common metric to compare and *benchmark* the performance of our framework.

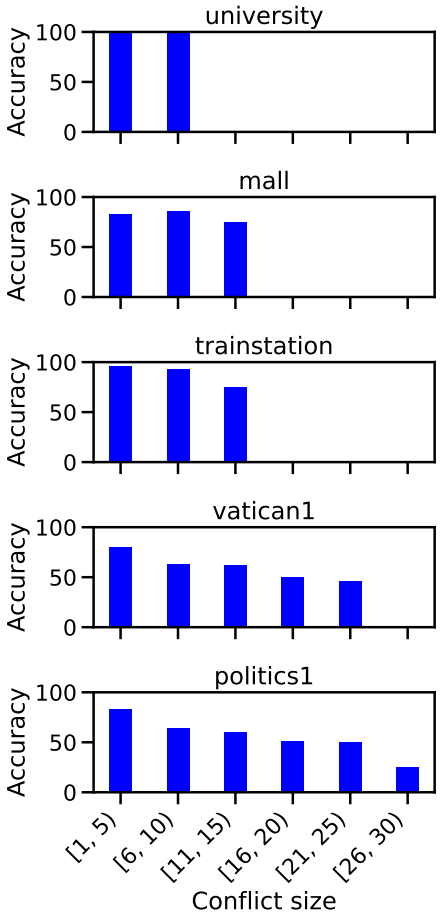


Figure 5.10: Association accuracy in different conflict sizes bins.

Consequently, we look at the performance of `Bleach` with respect to $|\mathcal{C}(T_c^{\tau_i})|$ seen in various input datasets. In Figure 5.9, we observe the distribution of conflict sizes resolved in various scenarios. *University* and *trainstation* have relatively lower value of $|\mathcal{C}(T_c^{\tau_i})|$, which should transform in to better accuracy of association. Indeed, Tab. 5.3 validates the claim as we achieve overall accuracy of 99.14% and 94.82%, respectively. *Vatican1* and *politics1* have mid-range conflict sizes resulting in slightly lower but good

accuracy. In contrast, highly dense shopping mall scenarios like *mall*, *HK dataset 1*, and *HK dataset 2* face considerably high conflict sizes for address resolution resulting in lower accuracy among the input datasets.

Next, we investigate the variability in association accuracy inside a single scenario across time. The hypothesis is that even with higher overall conflict sizes, there might be periods with low $|\mathcal{C}(T_c^{\tau_i})|$, which could be exploited by the adversary to resolve randomized addresses of target user devices. We indeed observe in Figure 5.10 that all scenarios generally have periods with low conflict sizes that yield better accuracy. While scenarios like *university* and *trainstation* perform reasonably well in all low $|\mathcal{C}(T_c^{\tau_i})|$, *vatican1* and *politics1* see a wide range of high conflict sizes causing a depletion in achieved correct MAC associations.

This characterization acts as a benchmark for `Bleach` in any new input datasets to the framework with similar or higher expected values of $|\mathcal{C}(T_c^{\tau_i})|$. It ensures the reliability of our framework, unlike other existing frameworks in the literature, which perform variably in different contextual scenarios in proprietary datasets.

5.6.4 Datasets without ground-truth

For datasets with no *ground-truth* (here HongKong dataset), we propose an alternate metric that denotes the correct association of the MAC addresses. The proposed metric is the sojourn time of a particular device around the sniffer zone. In the case of MAC randomization, the sojourn time is the sum of the sojourn times of all associated randomized MACs plus the time gaps between the associated MAC trails. More specifically, the device's sojourn time is the difference between the timestamps of the first frame of the first associated MAC trail and the last frame of the last associated MAC trail.

In the case of randomized MACs, which are not associated, the sojourn times correspond to the lifetimes of each random MAC address that the device advertises. While in the case of true or non-randomized MAC addresses, the sojourn time is the time for which the device was seen in the sniffing zone.

Hypothesis: We propose the hypothesis that for a large number of users observed by the sniffers, the distribution of the sojourn times of correctly associated MAC addresses and the true MACs advertised by users should demonstrate similar behavior in a scenario during a given period of time. To recall, the true MACs are the physical MAC addresses of devices that remain static across all sent probe-requests. The consistent nature of human mobility during that short period, and the uniform randomization nature of the device's MAC address for the large population, ensures that the sojourn times of devices are independent of MAC randomization.

Observations: In Figure 5.11, we present the probability densities of

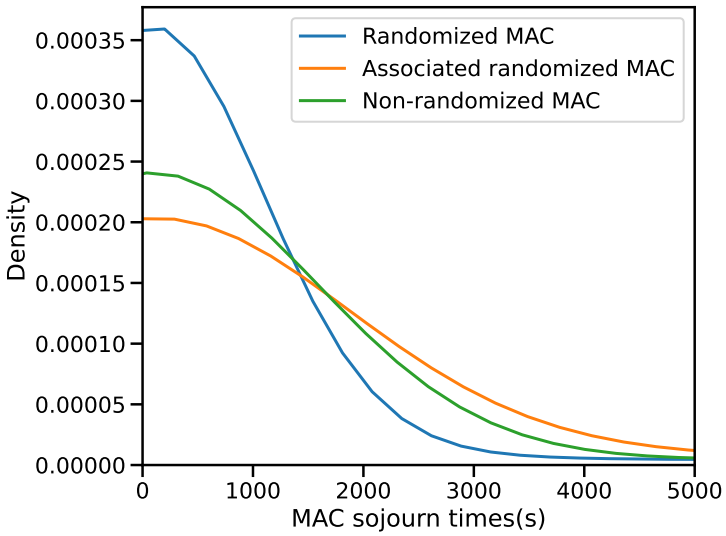


Figure 5.11: MAC sojourn times before and after association.

sojourn times observed when considering probe-requests from HongKong dataset that advertise non-randomized, randomized, and associated MAC addresses. Here, we consider around 9000 randomized and non-randomized MACs. We observe that randomized MAC addresses have quite lower sojourn times than the other two, as expected. Devices change MAC addresses frequently, lowering the time for which one of its random MAC was seen in the sniffing zone. Next, to validate the effectiveness of MAC association in `Bleach`, we look at the closeness between the sojourn time of devices of non-randomized MAC addresses of a device and the associated randomized ones. We notice that the sojourn times of devices after association and those of non-randomized ones are indeed very similar in their distributions. Perfect overlap is not possible because of the limits of the association algorithms in highly dense (in terms of probe-requests) and mobile scenarios like shopping malls (cf. Figure 5.9).

5.7 Conclusion

MAC address randomization is used by modern WiFi devices, where randomly generated virtual MAC addresses are used in probe-requests, instead of true MAC addresses. We find out that current address association frameworks underperform and are unreliable with respect to new input datasets. We henceforth present `Bleach`, a framework capable of associating randomized probe-requests advertised in the observation zone. We implement `Bleach` and used extensive datasets in different contextual scenarios which

shows that `Bleach` is robust and greatly outperforms the state-of-the-art works in terms of accuracy.

In the next chapter, we build upon the WiFi MAC association to show that users' trajectories can successfully be obtained by just utilizing the RSSI values of the transmitted probe-requests. Knowing the associated MAC addresses from a target device and the possible locations from which the probe-requests were subsequently sent, we reveal the reconstructed trajectory of the user. This poses a serious privacy concern for user devices communicating on WiFi in public spaces.

Chapter 6

Inferring users' bounded trajectories from WiFi

Contents

Having associated randomized WiFi MAC addresses through `Bleach` in the previous chapter, we proceed to reveal user trajectories through non-intrusive methods. Non-intrusive methods refer to approaches that do not require the use of GPS, installed applications, or any active participation

from the user. Instead, these methods rely solely on passive measurements/sniffing of public packets and signal characteristics. Passive sniffing-based trajectory inference has proven to be challenging, primarily due to significant errors in user localization. The prevalent localization metric based on received signal strength (RSSI) often suffers from inaccuracies that have been shown in the existing literature.

Henceforth, we perform a thorough assessment of RSSI-based trajectory inference in Section 6.1. Recognizing the severity and non-stationary behavior of localization errors, we introduce the notion of *bounds*. In this chapter, we propose `Allies`, a novel framework that introduces the concept of a user's bounded trajectory, leveraging the signal strength of observed WiFi probe-requests from the user.

To deduce bounds of trajectories, we perform a novel characterization of errors in RSSI-based radial-distance estimation between the user and the sniffer (cf. Section 6.4) that make their modeling possible. We then leverage this error characterization and approximated radial distances to estimate the bounds associated with a user position. We can infer a user's bounded trajectory by considering the spatiotemporal bounds of user positions over time (cf. Section 6.5). Our approach ensures that the bounds enclose a user position in both space and time with a high level of confidence and a low margin of error (cf. Section 6.7) [75].

This chapter demonstrates the effectiveness of the `Allies` framework in mitigating localization errors and providing reliable insights into user trajectories. By addressing the challenges posed by non-intrusive methods and localization inaccuracies, `Allies` offers a promising solution for understanding and tracking user movements in various passive sniffing scenarios.

6.1 Assessing RSSI-based trajectory inference

The trajectory $Traj_j$ of user j can be described as shown in Equation 6.1.

$$Traj_j = \{(x_j^0, y_j^0, t_j^0), (x_j^1, y_j^1, t_j^1), ..., (x_j^p, y_j^p, t_j^p), ..., (x_j^N, y_j^N, t_j^N)\} \quad (6.1)$$

It is the set of tuples consisting of X, Y coordinates and timestamps, with elements ordered in time. Each of the elements in $Traj_j$ denotes the obtained location of a user at a particular time. It is a sequence of spatial coordinates or positions that indicate the changing locations over a specific period. Hence, this section investigates location inference from RSSI as a basis for trajectory inference. RSSI is the only information that we have that we can use to infer the distances separating the off-the-shelf sniffers from the device.

To investigate raw RSSI credibility, we look at the measured RSSI of collected probe-requests. To figure out its potential for 2D spatial estimation of users' location, we build upon literature solutions [76] to the *radial-distance*

estimation (i.e., distance in meters separating users from sniffers) from packets' RSSI. The methodology employed is as follows.

We use the public anonymized *Sapienza* dataset [70], describing a large number of WiFi probe-requests passively collected in eight different contextual scenarios (cf. Section 2.5). We choose datasets from five scenarios: *vatican1*, *vatican2*, *university*, *trainstation*, and *mall*. The three first scenarios denote dense outdoor environments, and the last two showcase ample public spaces highly frequented. There are other notable distinctions among the data with respect to attributes such as user density, sojourn times, etc. The datasets contain the timestamps of received probe-requests and RSSI values observed by the capturing sniffer. There is no MAC randomization in *Sapienza* datasets which allows the inference of per-user behavior in terms of distances from sniffers.

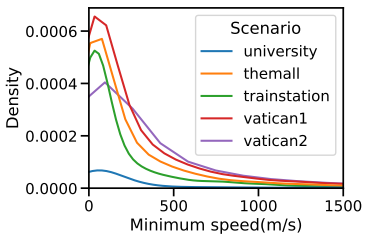
Next, we proceed with estimating radial distances from the recorded RSSI values from the probe-requests and look at the correctness of obtained location inferences. Results validate our assumption on the inaccuracy of RSSI to give accurate user trajectories which leads us to model the underlying errors in distance-estimation and eventually propose effective *bounded trajectories*.

6.1.1 Estimating radial distances

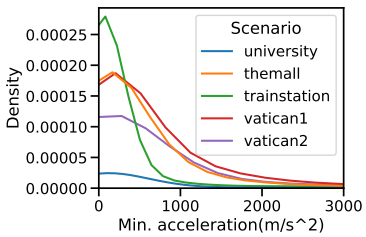
Using the raw RSSI values described in the aforementioned datasets, we estimate radial distances through the log-distance path-loss model [76] (cf. Equation 6.3) configured with standard parameters and varying path-loss exponent γ. The γ controls the "severity" of the losses in datasets environments where probes were collected and keep changing with space and time. We vary γ over a range of practical limits as we do not know the actual extent of losses beforehand. The chosen γ signifies the average value of the exponent in a particular scenario.

As for the *Sapienza* datasets, no *ground truth* with actual device-sniffer radial distances is available for large-scale, passively collected datasets. In addition, information on the orientation direction of mobile users is not available, which hardens the distance estimation's accuracy. Hence, to verify the correctness of the computed radial distances, we first obtain the corresponding range values of (i) instantaneous *speeds* –, indicating users' changing behaviors in estimated radial distances over time – and (ii) *accelerations* –, indicating the geographical dynamism of users' mobility, per time window of 50ms. We then compare the resulting minimum values of speed and acceleration with the corresponding realistic values of human mobility given in [77]: $1m/s$ for walking, $2.3m/s$ for walk-run mixture, and $3m/s$ for running.

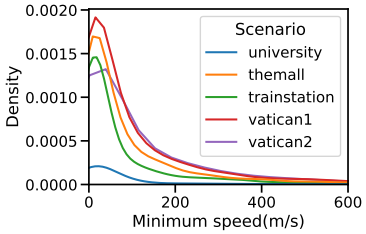
Figure 6.1a, 6.1c, and 6.1e reveal the range of minimum speeds issued from the radial distances estimated from datasets' raw RSSI values, consid-



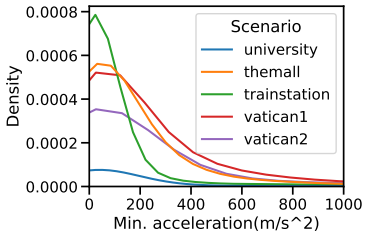
(a) Low losses ($\gamma = 2.5$).



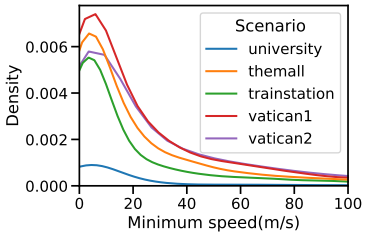
(b) Low losses ($\gamma = 2.5$).



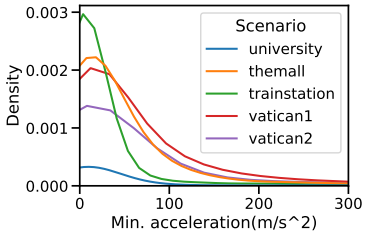
(c) Moderate losses ($\gamma = 3.0$).



(d) Moderate losses ($\gamma = 3.0$).



(e) High losses ($\gamma = 4.0$).



(f) High losses ($\gamma = 4.0$).

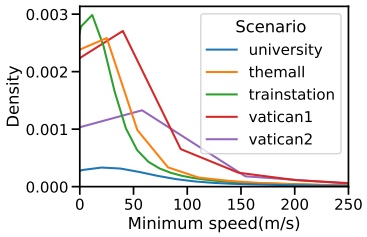
Figure 6.1: min. speed (a), (c), (e) and min. acceleration. (b), (d), (f) from raw RSSI values.

ering path-loss severity of $\gamma = [2.5, 3.0, 4.0]$. Similarly, Figure 6.1b, 6.1d, and 6.1f show the range of minimum accelerations. The density values give the proportion of the instantaneous speed and acceleration values that we observe in the considered dataset. We note that even in the highest severity context, i.e., $\gamma = 4.0$, the minimum speed and acceleration values reach up to $80m/s$ and $200m/s^2$ (Figure 6.1e and 6.1f). Those are unrealistic human values of speed and acceleration, indicating the presence of errors in the radial-distance estimations associated with the datasets' raw RSSI values.

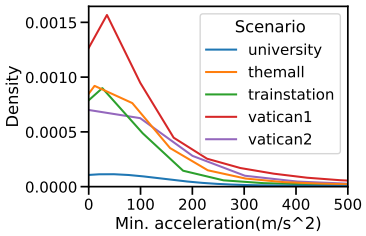
6.1.2 Can literature handle errors in distance estimation?

The primary cause of distance-estimation errors is the significant fluctuations in RSSI values over short periods of time. To reduce fluctuations, various filters have been suggested in the literature to smooth noisy raw

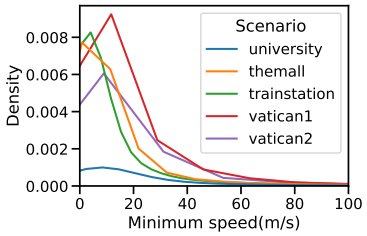
RSSI values that cause distance-estimation errors [53,65,78–81]. The works discuss and evaluate various smoothing algorithms for the RSSI observed in passive sniffing. Smoothing approaches include feedback, moving-average, or median filters, among others.



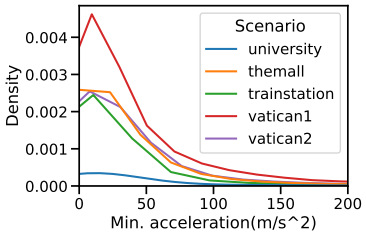
(a) Low losses ($\gamma = 2.5$).



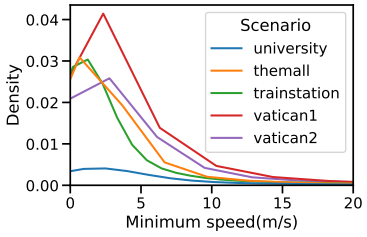
(b) Low losses ($\gamma = 2.5$).



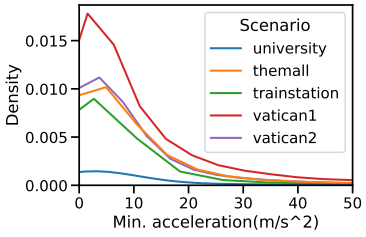
(c) Moderate losses ($\gamma = 3.0$).



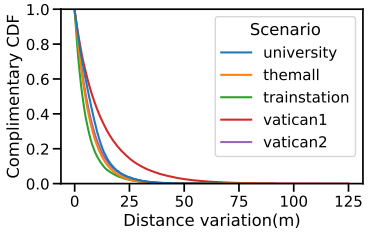
(d) Moderate losses ($\gamma = 3.0$).



(e) High losses ($\gamma = 4.0$).



(f) High losses ($\gamma = 4.0$).



(g) Distance variation (m)

Figure 6.2: min. speed ((a),(c), (e)), min. acceleration ((b), (d), (f)) from smoothed RSSI values, and (g) Distance estimations computed from EWMA [4] smoothed RSSI values, with smoothing factor α=0.1.

We choose the Exponentially Weighted Moving Average (EWMA) filter [4] with a smoothing factor of $\alpha = 0.1$, which gives exponentially decreasing weights for already observed raw RSSI values. The α value ranges between 0 and 1, and selecting a value closer to zero results in a stronger smoothing effect, making the system less sensitive to recent fluctuations in RSSI. EWMA is considered the state-of-the-art model and is more effective than other classical moving average filters. Finally, as for the raw values, we estimate the radial distances related to the smoothed RSSI values and obtain mobile users' corresponding smoothed speeds and acceleration.

Figure 6.2a, 6.2c, and 6.2e reveal the range of minimum smoothed speeds for various scenarios and varying loss severity (γ). Similarly, Figure 6.2b, 6.2d, and, 6.2f show the corresponding range of minimum accelerations.

With the highest severity value of $\gamma = 4.0$, the minimum smoothed speed, and corresponding acceleration reach out to around $15m/s$ and $40m/s^2$ (Figure 6.2e and 6.2f). Although still not corresponding to realistic human speed and acceleration, smoothed results bring a considerable improvement with respect to non-smoothed values computed from raw RSSI measurements. Still, most obtained results for smoothed speeds and corresponding accelerations of users are unrealistic.

In the next, we discuss the potency of location inference when utilizing RSSI along with literature methods. We discuss the current challenges in RSSI exploitation which leads us to a new framework proposed in Section 6.2.

6.1.3 Discussing the effectiveness of location inference

Unrealistic induced minimum speed and acceleration values show that radial-distance estimation is incorrect. The incorrect estimation is attributed to the noisy RSSI observations in dense outdoor scenarios. The errors vary with scenarios and are non-stationary (change with time for different users). Furthermore, when localizing a user in 2-dimensions, distance-estimation errors from multiple references will further deteriorate the estimated coordinates accuracy of the users positioning. In addition, errors also contribute to the choice of γ in the path-loss estimation, which makes the precise inference of users' locations impossible through RSSI observations.

Although reducing the RSSI's fluctuations and the unrealism of derived speeds and acceleration, radial distances computed from smoothed RSSI values (using EWMA filter with $\alpha = 0.1$) are still inaccurate, presenting smoothing-induced errors that can reach up to 50m (cf. Figure 6.2g) in dense scenarios. Such observations stress *the difficulties in using RSSI values to estimate users' locations and trajectories accurately.*

The following section demonstrates how the introduced framework: `Allies`, addresses these challenges. Moreover, we adopt a unique approach to utilize

RSSI (Received Signal Strength Indicator), showing that although errors may be present, they can be effectively modeled to infer trajectories with a level of precision that allows for extracting valuable information.

6.2 Allies overview

Consider the context of a passive and non-intrusive wireless measurement where no precise location of a user is possible to be collected in the measurement. As previously mentioned, we rely on RSSI measurements captured from public frames, which will be later used for location estimation. RSSI values are highly unstable per se due to environmental variability as seen in the last chapter. Variations in the strength of the reception of electromagnetic waves are due to factors like path-losses and multi-path fading. Besides, there are errors in distance estimation from the observed RSSI. Hence, (x_j^p, y_j^p, t_j^p) in Equation 6.1 is just an estimate of the actual user co-ordinates.

This inevitably raises the question of the certainty of the estimated user trajectory, T_j. Any method that we use to specify the locations of a user must be accompanied by a limit on the deviation from every (x_j^p, y_j^p, t_j^p) in $Traj_j$, in order to make sense of the trajectory. To find this limit, we need to solve major challenges, knowing the behavior of errors in the location estimation while considering human mobility. We also need to know the evolution of errors incurred in distance estimation in time from multiple sniffers, as well as the maximum guarantee that we can possibly provide, given the measurement conditions. We introspect these open questions in the introduced framework: `Allies` and find a limit to the obtained locations in $Traj_j$. We call this limit on user locations in small time intervals as *bounds of locations*.

Bounds of locations: We define *bounds of locations* $(\mathcal{L}_{t_n})$ as a set of timestamped points describing the possible locations of a user during the interval $t_n = [t0 + \delta t * (n-1), t0 + \delta t * n]$ as:

$$\mathcal{L}_{t_n} = [(t1, x1, y1), (t2, x2, y2), ..., (tn, xn, yn)]$$

Here, δt is the size of time intervals in which we divide the duration of an observed user. The idea is that the if we know the exhaustive list of probable locations, the area encircling the respective locations across various δts will yield the *bounded trajectory*.

Combining all time intervals during the period we observe a user, results in what we call a *bounded trajectory*.

Bounded trajectory: This trajectory $\mathscr{B}_{user_j}$ is defined as the sequence of successive convex hulls[1] ordered in time, spanning the trajectory. We decide

[1] `https://mathworld.wolfram.com/ConvexHull.html`

to divide the user-trajectory into multiple convex hulls to capture the turns (changes in direction) that a user takes over time. Formally, a bounded trajectory, $\mathscr{B}_{user_j}$, of user j, is defined as:

$$\mathscr{B}_{user_j} = \{\mathcal{CH}_{j1}, \mathcal{CH}_{j2}, \mathcal{CH}_{j3}, ..., \mathcal{CH}_{jk}, ..., \mathcal{CH}_{jM}\}$$

where:

$$\mathcal{CH}_{jk} = \{\mathcal{L}_{t_1}, \mathcal{L}_{t_2}, ..., \mathcal{L}_{t_N}\}$$

where $\mathcal{CH}_{jk}$ represents the kth member of the sequence of convex hulls of the trajectory of the jth user. Hull contains coordinates with timestamps. Hence, $\mathcal{CH}_{jk}$ gives the "bounded area" for finding the jth user with a sense of "time."

Still, the following question remains: *What is an ideal bound of a user trajectory?* Larger bounds of users' locations in terms of covered geographical area intuitively ensure *inclusiveness*, as, on average, the probability of finding users inside the specified wide bounds is high. However, an ideal user's bound of location must ensure *correctness* – i.e., must include the real precise user's location – and be *narrow* – i.e., must limit the width of possible locations inside the bound. Unfortunately, given the highly unstable nature of RSSI measurements, achieving such ideal bounds of locations is a challenging task.

`Allies` addresses the above challenges and comprises the following three steps:

1. **Generating *observation Sets*:** We illustrate the first step in Figure 6.3. The aim of this step is to obtain *observation Set* containing "views" of different sniffers for a particular user in a small time interval (δt). An entry in an observation set is a group of 3-tuples (*timestamp*, *RSSI*, *sniffer id*) for all probe-requests that we see in the input dataset. We detail more on the definition in Section 6.3.

2. **Error Characterization:** We characterize the radial-distance errors from observation sets as illustrated in Figure 6.4. For the first time in literature, we identify that estimation errors can be viewed as a sum of two additives error components. The first component denotes the mean error behavior incurred in radial distances estimations, which we call *span error*. The second component captures the fluctuations in distance estimations originating from the changing RSSI values of a user in short time intervals, which we call *environment error*. We quantify these fluctuations and estimate their distribution for each observation set in the user trajectory.

3. **Generating *bounded Trajectories*:** The objective of the last step is to obtain *bounded* trajectories of users observed in the sniffing zone, as illustrated in Figure 6.5. First, we do a Gaussian fit to *environment*

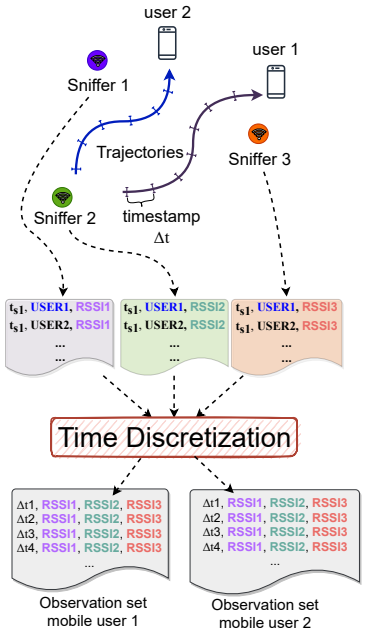


Figure 6.3: Step 1: Generating observation sets

error distributions. Then, to obtain the total estimation error, we sample multiple times from the resulting fitted distribution and add the error samples to the corresponding *span error*. Next, we add the obtained total errors to users' radial-distances and feed them to our multilateration-based location estimator. Depending on the number of samples drawn from the fitted environment error, we obtain a set of user positions named *bounds of locations*. Finally, we aggregate these *bounds of locations* per observation set to get the *bounded trajectories* of a user. We detail the second step in Sec. 6.4 and the last step in Sec. 6.5.

For evaluation of `Allies`, we need a full-fledged *ground-truth* of mobile user-locations, which is almost impossible in passive monitoring using WiFi sniffers. No large public dataset in literature has the *ground-truth*, with the new data-collection facing additional challenges such as government regulations, scaled sniffer deployment, and high user mobility. We address this huge issue concerning user-localization works by introducing *WiSurve* simulation framework in Section 6.6. `WiSurve` provides controllability in the setup of the user mobility and sniffers deployment. *WiSurve* enables obtaining real-world, passive sniffing environments for WiFi which eventually makes possible the evaluation of our proposed *bounds*. We also complement it with two real-world collections named *Campus dataset 1* and *Campus*

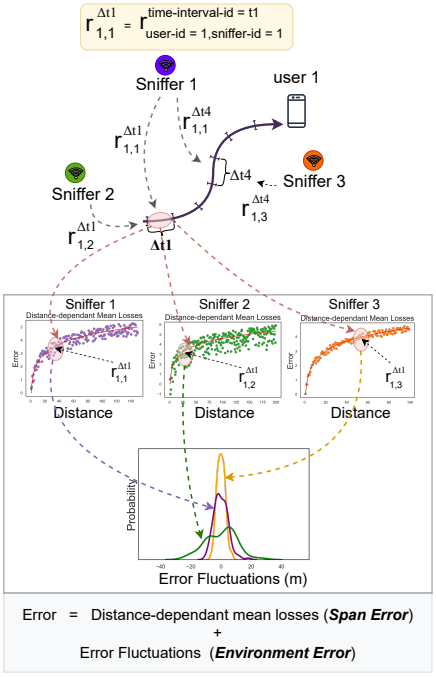


Figure 6.4: Step 2: Error Characterization

dataset 2 that we use for calibration purposes.

After already briefing the steps of `Allies` and having *WiSurve datasets* for the justification of the framework's design choices, we next proceed to detail each step of our framework.

6.3 Step 1: Generating *Observation sets*

We define observation set $Ob\_Set^{t_n}$ in a particular time interval $t_n = [t0 + \delta t * (n-1), t0 + \delta t * n]$ as shown in Equation 6.2.

$$Ob\_Set^{t_n} = \{(time_i, RSSI_i, sniffer ID_i)\} \tag{6.2}$$

To tackle the variation of localization errors incurred by environment changes associated with users' mobility, we divide the duration for observing a user into time intervals, noted as δt.

We meticulously select the value of δt to meet several requirements. Firstly, to achieve accurate 2-dimensional localization and effectively represent variations in environmental errors, δt needs to be sufficiently large for:

1. Accommodating multiple reference points capturing a user's presence

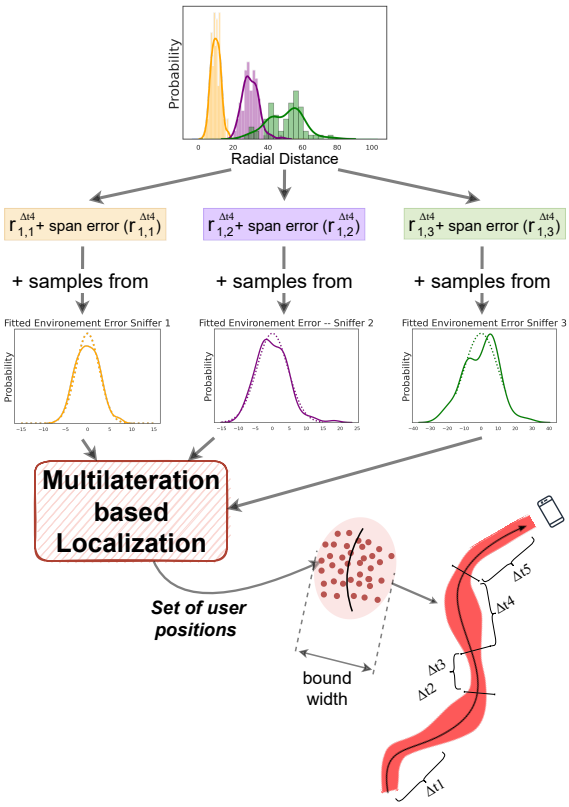


Figure 6.5: Step 3: Generating *bounded Trajectories*

2. Gathering an adequate number of probe-request samples within its time interval.

However, it is also essential to restrict the value of δt to avoid significant changes in the user's position. We satisfy this trade-off utilizing WiFi probe-request rates in the dataset and the best error characterization of *environment errors* (cf. Section 6.4.2) to set the value of δt. We empirically set the value of δt as 4 s.

6.4 Step 2: Characterizing estimation errors

We next characterize and approximate the errors in radial-distance estimation. To account for errors in radial-distance estimation, the following steps are taken:

1. RSSI values are converted into approximate distances through path loss models with optimal parameters. A *error calibration* methodology

is proposed to ensure the best choice of model inputs. (cf. Section 6.4.1)

2. Errors in distance estimation are analyzed and formulated using the optimal path-loss model. (cf. Section 6.4.2)

3. The errors are further separated into two components with distinct behavior, namely *span errors* and *environment errors*, which are utilized for error characterization. (cf. Section 6.4.3)

6.4.1 Converting RSSI to approximate radial-distances

We use a path-loss model to estimate the losses incurred in the wireless medium. As we can see in Eq. 6.3, `Allies` uses the state-of-the-art log-distance path loss model [76] to capture losses (PL) encountered for signals in densely populated areas.

$$PL = P_{Tx_{dBm}} - P_{Rx_{dBm}} = PL_0 + 10\gamma \log \frac{r_{ji}}{r_0} + \mathcal{X}_g \tag{6.3}$$

PL is the total path loss in decibels (dB) at a distance r_{ji} between the user j and the sniffer i. $P_{Tx_{dBm}}$ and $P_{Rx_{dBm}}$ are the transmitted and received power in dBm. PL_0 is the path-loss (dB) at reference distance, r_0. γ is the path loss exponent that depends on the propagation characteristics of the received signal. Finally, X_g is a normal random variable that takes into account the losses (due to obstructions caused by buildings, pedestrians, etc.) from shadow-fading in outdoor scenarios.

We set the reference distance r_0 to be 1m for which we know the path-loss PL_0. Using Eq. 6.3, we then compute the approximate radial distances (r_{ji}) between users and sniffers by using RSSI values directly observed from real WiFi probe-requests, as in Eq. 6.4.

$$r_{ji} = 10^{\frac{P_{Tx_{dBm}} - P_{Rx_{dBm}} - PL_0 - \mathcal{X}_g}{10\gamma}} \tag{6.4}$$

Choice of path-loss model parameters: To obtain the radial distance r_{ji}, we need to estimate the optimal values of γ and $\mathcal{X}_g$, as in Eq. 6.4. The other inputs to the equation are either standard: $P_{Tx_{dBm}}$ (set to 23 dBm) and PL_0 (set to 46.67 dB), or known: $P_{Rx_{dBm}}$ (observed RSSI).

Values of γ vary from around 2 in line-of-sight (LoS) environment to 3.5 in dense urban scenarios [82]. Hence, we set the range of γ values, i.e., R_γ, to be $[2.0, 4.0]$, corresponding to the range of values of path-loss exponents considered in optimal parameter search.

To consider shadow fading, we model $\mathcal{X}_g$ as the random variable with a Gaussian distribution [76] with zero mean (μ) and standard deviation (σ) in decibels. The σ varies from close to 0 in free space to around 5 [83]. We set the search-space for the standard deviation of the random variable, i.e., $R_{\mathcal{X}_g}$, to be $[0, 5]$.

Optimal parameters in real-world scenarios vary with the chosen deployment areas. We have to calibrate the parameters in the area where we *set-up* sniffers. Next, we propose an one-time calibration step to obtain γ_{opt} and $\mathcal{X}_{g_{opt}}$. We further demonstrate the calibration methodology using specific datasets collected in real and simulated environments.

Calibrating loss-parameters in the deployment area: We propose a calibration methodology that considers a real sniffers deployment, as described in Alg. 5. Furthermore, we design the calibration such that we can obtain optimal path-loss parameters for real (*Campus dataset 1* and *Campus dataset 2*) environments but could be extended to simulated environments (*WiSurve dataset*) too.

For *Campus dataset 1* and *Campus dataset 2*, during the collection procedure inside a busy university campus' open spaces, a controlled WiFi probe-request generation device and a nearby passive sniffer were placed at fixed and known positions to have the knowledge of source-to-sniffer distances. The generated probe-requests did not randomize their MACs allowing us to recognize packets from the generator at the receiving sniffer.

The source (i.e., S_{probes}) of WiFi probe-requests is kept at a fixed location. Sniffers are placed at known regular distances (i.e., $D_{sniffers}$) from the source, and probe-requests are sent for a particular duration at each of the distances (i.e., *sink_dist*).

After having the calibration's per-distance traces, we then find optimal path-loss parameters. We generate a number (i.e., $Size$) of uniformly-spaced samples (i.e., S_γ and $S_{\mathcal{X}_g}$) from the R_γ and $R_{\mathcal{X}_g}$ (cf. line 10). For each combination (i.e., $S_{\gamma,\mathcal{X}_g}$) of drawn samples, we calculate the *distance-estimation error*. We define the error as the average of the differences between the distances calculated from the RSSI values (see Eq. 6.4) and corresponding ground-truth distances (i.e., *sink_dist*) (cf. line 15). Finally, we obtain optimal parameters $(\gamma_{opt}, \mathcal{X}_{g_{opt}})$ by finding the combination minimizing the distance-estimation error (i.e., $AllErrors$) (cf. line 17).

6.4.2 Formalising the distance-estimation error

Having the estimation and calibration, we finally can obtain user distances using Equation 6.4, which has embedded estimation errors. In the following, we form two hypotheses for the behavior of distance-estimation errors in outdoor door scenarios. The "intuitions" behind the two hypotheses were obtained from our study of error causes and our observations while calibrating the path-loss model.

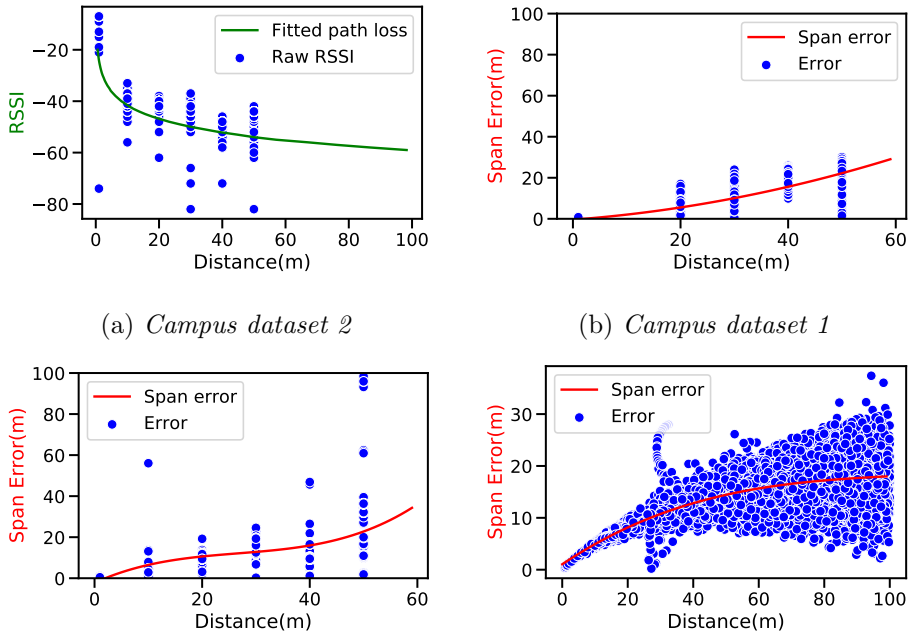
- **Hypothesis 1**: *The average estimation error increases with an increase in the radial-distance.*

  The average error depends on fitting the optimal path-loss model with the distribution of the RSSI propagation in the deployment area. We

Algorithm 5 Calibration & Optimal path-loss parameters.

procedure GETOPTMODELPARAMS($D_{sniffers}, S_{probes}, R_\gamma, R_{\mathcal{X}_g}, Size$)
  $Traces = []$ ▷ Initialising trace collection
  SetPacketGenerator(S_{probes})
  for $sink\_dist \leftarrow Enumerate(D_{sniffers})$ **do**. ▷ Calibration trace collection
    DeploySniffers($sink\_dist$)
    StartPacketGenerator()
    $Traces$.Add(CollectRssiTrace($sink\_dist$))

  $AllErrors = []$ ▷ Average distance-estimation errors
  S_γ = GenUniSamples($R_\gamma, Size$) ▷ Finding optimal parameters
  $S_{\mathcal{X}_g}$ = GenUniSamples($R_{\mathcal{X}_g}, Size$)
  for $S_{\gamma,\mathcal{X}_g} \leftarrow JointSamples(S_\gamma, S_{\mathcal{X}_g})$ **do**
    $Error = 0$
    for $tr \leftarrow Enumerate(Traces)$ **do**
      $Error = Error$+[CalcDistFromRSSI $(tr, Sample_{\gamma,\mathcal{X}_g})$ - $sink\_dist_{tr}$]
    $AllErrors$.Add((Mean($Error$), $S_{\gamma,\mathcal{X}_g}$))
  $\gamma_{opt}, \mathcal{X}_{g_{opt}}$ = FindMinimum($AllErrors, (\gamma, \mathcal{X}_g)$)
return (γ_{opt}, $\mathcal{X}_{g_{opt}}$)



(a) *Campus dataset 2* (b) *Campus dataset 1*

(c) *Campus dataset 2* (d) *WiSurve dataset*

Figure 6.6: (a) Calibration of path-loss parameters. (b)-(d): Errors in radial-distance estimation.

can see the illustrated optimal path-loss fit, which we obtain using *Campus dataset 2* in Figure 6.6a. As signal degradation varies considerably after an initial threshold distance, the optimal path-loss fit must react to more "severe" RSSI fluctuations at increasing distances. This should cause an increase in the average estimation error.

- **Hypothesis 2**: *The error fluctuations in an observation set (δt) grow with increasing radial-distances. Their distribution can be modeled in small intervals of time.*

  The shadowing and muti-path fading increase with the signal propagation distance. This should result in a higher range of RSSI values that we observe at a given distance, leading to an increasing variability of estimation errors. Moreover, environment errors result from the interaction of multiple individual error "sources". We assume individual errors from these "sources" to be independent and identically distributed. Also, with user-mobility, the error "sources" associated with a user are likely to change due to physical environment changes. Given a large number of error "sources" that cause RSSI fluctuations in δt, the distribution of the resulting sum (i.e., environment error) should converge to the normal distribution [84].

Hypotheses validation: To attest our hypotheses, we extensively investigate using two case studies and three datasets, the behavior of radial-distance errors when approximated with the log-distance path-loss model with optimum parameters (cf. Section 6.4.1) using two case studies.

In the *first case study*, we extract RSSI values from two real-world passive sniffing datasets: *Campus dataset 1* and *Campus dataset 2*. Then, we infer the related radial-distance errors. These datasets represent a unique static user that a unique sniffer captures. This static and simple case evaluates errors' behaviors when fixed and discrete distances between the user and the sniffer are considered. Figure 6.6b and 6.6c show the obtained results where we visualize the behavior of errors in distance estimation.

In the *second case study*, we exploit large-scale *WiSurve dataset* to introduce a mobile scenario with varying distances between multiple users and sniffers. Here, we have 100 users that move with a walking speed of $1m/s$ in the sniffing zone (cf. Section 6.6.1) while emitting probe-requests. Here, Figure 6.6d depicts the behavior of estimation errors in this dynamic large-scale deployment case study.

From the results obtained from the two previous case studies, we can validate our hypothesis through the following observations. The first observation, related to **Hypothesis 1**, is that we note a consistent trend followed by the average error with respect to the radial-distance, as shown in Figure 6.6b, and 6.6c, and, 6.6d. As stated in **Hypothesis 1**, we do see that estimation error grows with increasing radial-distances. We further show that this

error-growing trend can be predicted through a polynomial approximation.

The second observation, related to the **Hypothesis 2**, can be spotted in Figure 6.6d. We can notice that the fluctuations in the distance-estimation error increase with larger distances. Moreover, when comparing the magnitude of fluctuations in Figure 6.6b, 6.6c, and 6.6d, we observe that they do change with varying environments. In the next section, we detail their estimation in each δt and their ability to be modeled by a distribution in Section 6.5.3.

6.4.3 Characterizing distance-estimation errors

Having formulated and validated the hypotheses, we now take the next step into characterizing them, which eventually helps in predicting distance-estimation errors. We divide the error into components based on the behavior stated in the two hypotheses. We call the first error component as *Span error*, which relies mainly on the *span* of the user from the sniffer. We name the second component as *Environment error*, as it depends upon the environment for signal propagation in the user's vicinity. *Correctly characterizing and then combining these two components allows us to bind the user between two radial-distances where it is likely to be present.* We estimate the two components of the radial-distance estimation errors using *WiSurve dataset.*

Estimating the span error: We estimate the *span error* after calibrating loss-parameters (cf. Section 6.4.1). Due to the relative stationarity of the sniffing zone (in terms of positioning and density of obstacles like buildings, bridges, etc.) with respect to the obtained optimal path-loss model, we consider that the mean behavior of errors (span error) does not vary noticeably with time.

It is the *environment error* that represents the fluctuations in RSSI in short periods of time due to changes in channel conditions (movement of cars, fellow pedestrians, e.t.c). We see this behavior in Figure 6.6d when observing the total error in distance-estimation across the course of user-trajectories. As shown in the figure, we can get the span error for a user, given its approximate radial-distance (span) using the procedure *GetSpanError* of Alg. 6.

The procedure utilizes calibration traces obtained during calibration, such as *Campus dataset 1* and *Campus dataset 2* (refer to Section 6.4.1). For each element in the RSSI trace (tr), we compute the error ($Errors$) by taking the difference between the distance obtained from the RSSI using the optimal path-loss model estimated in Section 6.4.1 and the corresponding ground-truth distances ($sink\_dist_{tr}$) in the input calibration trace (line 4).

Finally, we estimate the average error, i.e., the *span error*, by using a polynomial fit of order ($Order$) using ridge regression (cf. line 5) [85].

Algorithm 6 Error characterization

1: **procedure** GETSPANERROR(γ_{opt}, $\mathcal{X}_{g_{opt}}$, $Traces$, $Order$)
2: $Errors = []$ ▷ Initialising distance-estimation errors
3: **for** $tr \leftarrow Enumerate(Traces)$ **do**
4: $Errors$.Add((CalcDistFromRSSI $(tr, \gamma_{opt}, \mathcal{X}_{g_{opt}})$ - $Sink\_dist_{tr}$, $Sink\_dist_{tr}$))
5: $Span\_error$ = GetPolynomialFit(Errors, Order)
6: **return** $Span\_error$
7:
8: **procedure** GETENVERROR($CollectedTraces$, DT)
9: $Env\_error = \{\}$. ▷ Initialising environment errors
10: $Users$ = GetAllUsers($CollectedTraces$)
11: **for** $user \leftarrow Enumerate(Users)$ **do**
12: $User\_tr$ = GetUsertrace($CollectedTraces$)
13: Obs_sets = DiscretizeTrace($User\_tr, DT$)
14: **for** $ob\_set \leftarrow Enumerate(Obs\_sets)$ **do**
15: $Obs\_err = \{\}$. ▷ Initialising observation set
16: **for** $obs \leftarrow Enumerate(ob\_set)$ **do**
17: Obs_err[SniferID(obs)].Add (DistRSSI $(obs, \gamma_{opt}, \mathcal{X}_{g_{opt}})$)
18: **for** $s_id \leftarrow Enumerate(Keys(Obs_err))$ **do**
19: av_dist = Mean($Obs\_err[s\_id]$)
20: $Obs\_err[s\_id]$ = SubtAll($Obs\_err[s\_id]$, av_dist)
21: $Env\_error[user]$.Add(Obs_err)
22: **return** (Obs_sets, Env_error)

We find that polynomials of order 3 are the best fit to characterize *span errors*. Higher orders fail to catch the rate of change of error with distance in outdoor environments.

Estimating the environment error: In terms of the total error in the estimation of the distance between a user and a sniffer, environment error can be interpreted as a *"complement"* to the span error in the corresponding observation set. The environment error varies with each observation set in the user-trajectory. Hence, we define this error for each user across all observation sets per sniffer. We obtain environment errors for individual sniffers and, due to their different spatial positioning, we observe different environment fluctuations in an observation set.

To obtain the environment error, we first obtain per-sniffer traces for each user that we capture in the trace using the procedure *GetEnvError* of Alg. 6. Then, we discretize the per-user trace in time to get the *observation sets* of size DT (cf. line 13). For each observation set, we calculate the distance between the user and sniffer (cf. line 17). Finally, we subtract the mean error per sniffer from the user-distances observed in an observation set (cf. line 20). This "residue" captures the fluctuation of estimation errors in an observation set.

6.5 Step 3: Obtaining bounded trajectories

Until now, we have formulated and estimated errors in radial-distance estimation in RSSI. In the following, we utilize the estimated approximate radial-distances for inferring user locations. First, we formalize the *localization error* and deduce the optimal user location per time interval, by solving an optimization problem. Then, we propose a novel method to leverage radial-distance errors to infer the bounds of users' locations (cf. Figure 6.5).

6.5.1 Formalizing the localization error

Since the environment error and user location are likely to change with every δt due to propagation conditions and mobility changes, we decide to consider the formulation of localization error per observation set, using a multi-lateration technique. We suppose for the k'th observation set Obs_k^j; we have a number N_s of sniffers receiving probe-requests of a user, j. We use *multi-lateration*, a technique utilizing the intersection of circles, for the positioning of a user based on measured distances between users and multiple known reference points (sniffers) [86]. We represent the references' locations in two-dimensions as $S_i = (x_s, y_s)$. We define the estimated position of a user j as $T_j = (x_u, y_u)$. We define the radial-distance between user j and sniffer i in k'th observation set as R_{ji}^k:

$$R_{ji}^k = d(T_j, S_i) = \sqrt{\sum_{m=1}^{2} \left(T_j^m - S_i^m\right)^2} \tag{6.5}$$

We define the measured distance from the probe-request RSSI's between user j and sniffer i in the observer set as r_{ji}^k. Finally, we state the localization error as $LocErr_j^k$:

$$LocErr_j^k = \sqrt{\sum_{i=1}^{N_s} \left(R_{ji}^k - r_{ji}^k\right)^2} \tag{6.6}$$

6.5.2 Getting the optimal user-location

To find the optimal user-location in an observation set, we need to minimize the localization error. We have to solve a non-linear least squares problem for minimizing $LocErr_j^k$. We formulate the optimum location T_j^* of user j for the observer set k in Eq. 6.7.

$$T_j^* = \arg\min_{T_j} LocErr_j^k \tag{6.7}$$

Literature brings several approaches to solving Eq. 6.7 for optimal user-location [87–93]. One option is a closed-form solution, which generally cannot be considered when circles in multi-lateration do not intersect at one

common point [87–89]. However, numerical methods provide a stronger argument for the quest for an optimum solution. They give the *best approximate* solution, even in the condition of not-intersecting circles [90–93]. Numerical methods provide various search-based optimization algorithms such as the gradient descent, the Newton-Raphson, the simulated annealing, and the genetic algorithm that provide candidate solutions [94]. We use simulated annealing (SA) to obtain the global minimum with sufficient iterations [95].

6.5.3 Finding bounded trajectories

To obtain the bounds of users' locations, we utilize the formulated radial-distance errors while finding the optimal user-location (Alg. 7) . For each observation set $\delta t'$, we initially compute r_user for a user j using Equation 6.8. Here, r_user represents the set of approximate distances r_{ji} between the user j and each sniffer i that detects it (see line 6).

$$r\_user = \{r_{ji}\} \forall i \in \delta t' \tag{6.8}$$

Next, we proceed to **model the distribution of environment errors**. We discuss the behavior of the *environment error* in Section 6.4 within small intervals (δt). However, one potential issue we may encounter is the limitation on the length of δt, which could result in an insufficient number of probe-request samples in the dataset for accurately characterizing the error distribution.

To tackle the scarcity of samples, we find out that the actual distribution ($\mu = 0, \sigma = \sigma_{N\_distr}$) of environment errors can be estimated through best Normal fit (cf. line 12). This distribution estimation allows us to sample and add extra localization errors in δt. The sampling is done multiple times ($Depth$) to obtain the bounds of user locations in the considered observation set. We validate the effectiveness of Normal fit in Section 6.6.2.

To offer a **probabilistic guarantee** that encompasses the actual user locations within bounds, we introduce the concept of $Depth$ in Equation 6.9. This parameter accounts for both the level of confidence ($\mathbf{Z}$) and the *margin of error* (MOE) [75] in defining the bounds.

$$Depth = \frac{\mathbf{Z}^2 \sigma^2_{N\_distr}}{MOE^2} \tag{6.9}$$

The value of σ_{N_distr} varies with every δt. We fix the level of confidence to 95 % and the margin of error to 10%.

Lastly, we proceed to **derive the bounds of user-locations**. As demonstrated in Algorithm 7 and discussed in Section 6.4.2, we calculate the span error ($SpanE$) and environment error ($EnvE$) per sniffer for each observation set. The sum of $SpanE$ and samples drawn from the fitted Normal distribution (Dtr) yields the total error ($MesE$).

This enables us to deduce the possible user distances (r'_{user}) by adding the total error ($MesE$) to the approximate radial distances (r_{user}). To obtain the target location of the user (Opt_target) from the possible user-sniffer distances, we employ Simulated Annealing (SA) (see line 15). The resulting set of target coordinates (Opt_target) represents the bounds of user locations in each observation set (see line 17).

Algorithm 7 Generating localisation bounds

```
1: procedure GenMobiltyBounds(User_trace, Obs_sets, Init_pt)   ▷ input variables
2:     Bounds = []                                  ▷ Initialising bounds of user-locations
3:     for user ← Enumerate(User_trace) do
4:         User_bounds = []                                  ▷ Per-user localisation bounds
5:         for obs ← Enumerate(Obs_sets) do
6:             r_user = DistPerSniff(obs)
7:             Depth = GetNumSamples(σ_N_distr)
8:             for k ← 1 to Depth do
9:                 for sid ← Enumerate(obs) do
10:                    EnvE[sid] = EnvErrObs(obs, sid)
11:                    SpanE[sid] = SpanEr(μ(r_user[sid]))
12:                    Dtr = NormalFit(EnvE[sid])
13:                    MesE[sid] = SpanE[sid] + GetS(Dtr)
14:                r'_user = r_user + MesE
15:                Opt_target = SolOptLoc(Init_pt, r'_user)
16:                User_bounds.ADD(Opt_target)
17:        Bounds.ADD(user_bounds)
18: return Bounds
```

With the set of obtained optimal-target points, we construct the **bounded trajectory** by creating a sequence of convex hulls, as detailed in Section 6.2.

Subsequently, we proceed to evaluate the performance of the `Allies` framework. For evaluating the effectiveness of the framework, we first introduce, showcase the generation, and validate the utilized `WiSurve` datasets. Next, we detail the `WiSurve` framework and various parameters related to the generation of large-scale probe-request datasets.

6.6 `WiSurve` framework

The `WiSurve` dataset serves as a valuable resource for evaluating the performance of the Allies framework in large-scale passive sniffing scenarios with *ground-truth* information of user-positions. By leveraging this dataset, we can assess the accuracy and effectiveness of Allies in realistic WiFi environments, providing valuable insights into user trajectory inference and localization accuracy.

In this chapter, we rely on three real-world datasets as well as realistic synthetic datasets obtained from the `WiSurve` framework as shown in Table

6.1. The first two real-world datasets are non-public and fully anonymized datasets named *Campus dataset 1* and *Campus dataset 2* while the third one is *Sapienza* dataset (cf. Section 2.5). While real-datasets are used for the calibration (cf. Sec. 6.4.1) and the validation in real-world conditions, the WiSurve datasets allow us to have a comprehensive evaluation of `Allies` with the ground-truth. `WiSurve` provides controllability in the setup's wireless environment, user mobility, and sniffers deployment.

Name	Type	Scenario	Mobility	Ground truth
Campus dataset 1	Real-world	Outdoor	No	Yes
Campus dataset 2	Real-world	Outdoor	No	Yes
Sapienza dataset	Real-world	Mixed (indoor and outdoor)	Yes	No
WiSurve dataset	Emulated	Outdoor	Yes	Yes

Table 6.1: Used datasets

In the following, we give details related to the generation and the validation of large-scale `WiSurve` dataset, which is need for the large-scale evaluation of `Allies`.

6.6.1 Generating datasets

The nature of bounded trajectories in *Allies* depends upon the *quality* of probe-requests data we capture. The *quality* is expressed by: (i) the density of probe-requests captured by the deployed sniffers, (ii) the number of missed packets, on average, and (iii) the amount of time for which a sniffer perceives a user in an *observation set*.

The density of captured probe-requests increases with the number of deployed sniffers. Furthermore, the average number of missed packets increases with the increase in user density since sniffers are limited to listening one channel at a time. Finally, the time spent by a user in the sniffer view is a function of her mobility behavior. High mobile users will likely be seen for a shorter time by sniffers. Considering such issues, we generate datasets using *WiSurve* by varying the above metrics, namely: (i) the number of sniffers, (ii) the number of users, and (iii) the speed of users.

In Table 6.2, we detail the deployment parameters with the considered range of values and the default value. When varying a specific metric, we fix the values of the other metrics with the default values. We consider a passive-sniffing zone of size $600m$ x $600m$. We generate pedestrian trajectories with varied individual characteristics using SUMO [96], a traffic simulation tool. We generate trajectories with users performing a random walk in a Manhattan area tessellated in grids of size $20m$ x $20m$. To ensure connectivity, we keep the grid size smaller than the WiFi range inside `WiSurve`, which is more than $50m$. We deploy sniffers on randomly-chosen vertices

while APs are randomly placed in the overall zone. A user-trajectory is comprised of all the probe-requests sent by the device advertising a particular MAC address, across the simulation time.

We set devices to use 802.11n (2.4 GHz) WiFi standard with a simulation duration of $40min$. For accurate emulation of real-world outdoor environments, we employed a real-world data-based outdoor path-loss model [97] that represents a typical suburban setting in China (Shanghai), completed with various outdoor features such as buildings, bridges, vehicles e.t.c., that could impact signal transmission. As in Table 6.2, we vary the number of mobile users and sniffers in the simulation over a range of values. We also vary user speeds from walking ($1m/s$) to running ($2.6m/s$). We go for long traces, rather than multiple runs of small duration, to catch the influence of users' wireless encounters in time. Each trajectory in the trace is at least of length $300m$.

Parameter	Value	Default	Range	Parameter	Value	Default	Range
Zone	600x600m	-	-	Standard	802.11n	-	-
Grid size	20m	-	-	Path-loss	Kun [97]	-	-
N(sniffers)	-	80	25-100	User speed	-	$1m/s$	1-2.6m/s
N(APs)	25	-	-	Duration	40 mins	-	-
N(users)	-	150	50-200				

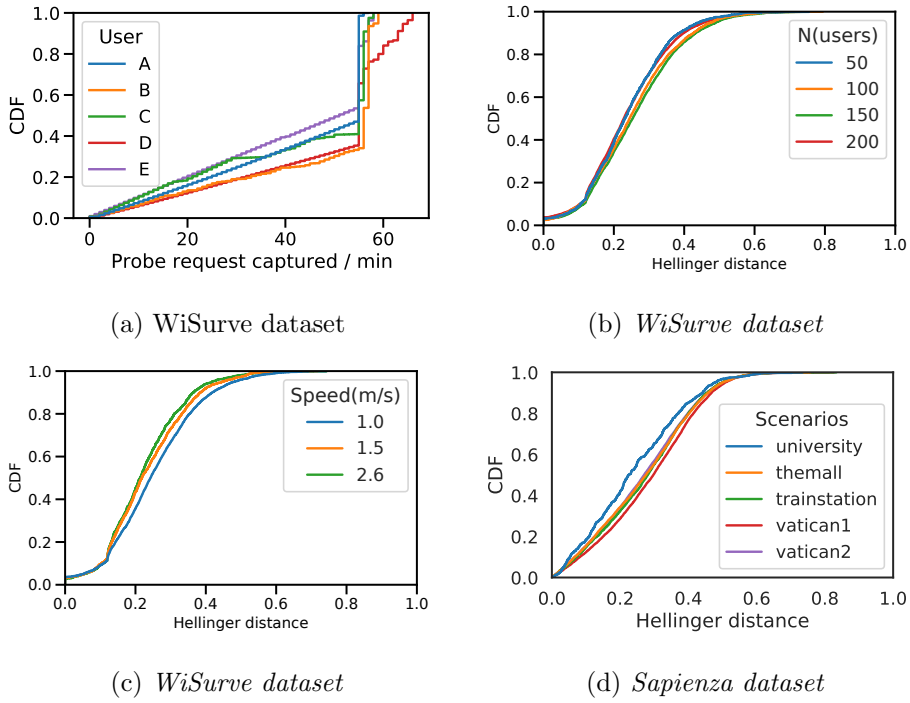
Table 6.2: `WiSurve`'s deployment parameters.

6.6.2 Validating datasets

Once we generate the dataset, we evaluate the realistic nature of the `WiSurve` dataset by comparing it with real datasets. The degree of realism is assessed according to the behavior of prove-request generation and the obtained environment error distribution.

Probe-request generation: We first ensure that there is no bias introduced in the simulation regarding probe-requests frequency emitted by user devices and the error behavior. We look at probe-requests captured per *min* in *WiSurve dataset*, shown in Figure 6.7a to certify this first aspect. We observe that the probe rate follows the same trend measured in various real mobile devices as in [74], i.e., around 50 probe-requests per *min* on average. Regarding the bias in error behavior, we already show in Section 6.4 that the behavior of span and environment errors using *WiSurve dataset* is similar to those obtained from the real-world ones.

Environment error distribution: The second validation concerns the assessment of the Normal-fit environment error distributions (**Hypothesis 2**). In this case, we calculate the Hellinger distance [98] between the actual and the Normal-fit environment error distributions in an observation set. We



(a) WiSurve dataset

(b) *WiSurve dataset*

(c) *WiSurve dataset*

(d) *Sapienza dataset*

Figure 6.7: (a): Probe-request rate for five random users. (b)-(d): Using Normal-fit on environment errors.

empirically compare the "similarity" of distributions over varying Hellinger distances and a large number of test cases and observe the distributions to be "similar" for Hellinger distances less than 0.4. As discussed in Section 6.5.3, we model and fit a distribution to the environment errors due to the possibility of lacking enough samples in an observation set. Small deviations from the fitted Normal distribution are expected.

In the evaluation, we first look at the real-world environment errors observed in the five heterogeneous scenarios of *Sapienza datasets* (cf. Section 6.1.2). We see in Figure 6.7d that the Hellinger distance is below 0.4 for around 80% of observation sets in *vatican 1*, *vatican 2*, and *university* outdoor scenarios. Then, we extend our evaluation to the large-scale *WiSurve dataset*. We observe in Figure 6.7b and 6.7c that despite increasing the user density and the speed, the distance between actual and fitted distributions is less than 0.4 for about 90% of the observation sets. It attests that for both real and simulated datasets, the environment errors are effectively emulated by Normal-fit, echoing the realistic nature of `WiSurve`.

Having obtained and validated large-scale passive sniffed WiFi probe-request traces, we proceed to the evaluation of *bounded* trajectories.

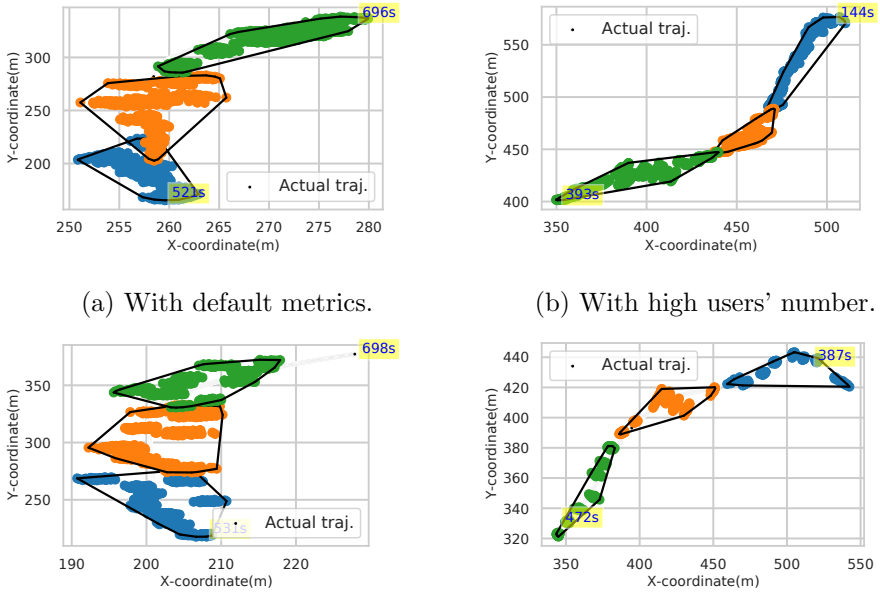
6.7 Assessing *bounded* trajectories

Finally, after having a large-scale passive sniffed probe-request dataset through `WiSurve`, we proceed to assess the obtained *bounded* trajectories. In this section, we first illustrate our concept of *bounded* user-trajectories. Then, we proceed to evaluate our results against its *inclusiveness* and *effectiveness*.

The general observation is that `Allies` does give inclusive bounds even in worse wireless medium conditions and, at the same time, maintains a high degree of effectiveness.

6.7.1 Illustrating bounded trajectories

Figure 6.8a illustrates the bounded user-trajectories in the default scenario (cf. Table 6.2). We observe that convex hulls completely bound the actual trajectory of the user for its entire duration in the sniffing zone (521s to 696s). Figure 6.8c certifies the successful identification of bounds of the user trajectory even for a reduced number of sniffers (25) when having a higher number of missed packets.



(a) With default metrics.

(b) With high users' number.

(c) With low sniffers' number.

(d) With high users' speed.

Figure 6.8: Bounded trajectories' illustration.

Although overshadowed by the points that convex hulls enclose, Figure 6.8 shows that the actual (*ground-truth*) trajectories inside the sniffing zone are fully captured within the revealed bounds. We also verify our framework against an increased number of users (200) (Figure 6.8b) and for higher

speeds ($2.6m/s$) (Figure 6.8d).

6.7.2 Quality of inferred bounds

We state the quality of bounded trajectory is expressed through two metrics (i) *correctness* and (ii) *width*. *Correctness* brings the intuition of "inclusiveness," where a user's bound of locations includes its real precise locations. The *width* captures how narrow the bounded trajectory is to limit the extent of possible user locations.

6.7.2.1 Bounds' correctness

We infer the "inclusiveness" of bounds as the length (in terms of the period of time) of the actual trajectory enclosed by the convex hulls ($\mathcal{CH}_{jk}$) of a user j. Given time intervals for which a user was observed during its ideal trajectory, we define "ideal inclusiveness" as the length of the actual trajectory enclosed between the minimum and maximum timestamps of $\mathcal{CH}_{jk}$. Note that the "ideal inclusiveness" acts as *ground-truth* for "inclusiveness". Finally, we compute the *Correctness* as the distance between the distributions of resulting bounds' "inclusiveness" and "ideal inclusiveness".

Figure 6.9a, 6.9c, and 6.9e investigate the length of actual trajectories inside the bounded ones (per hull), which depict "inclusiveness". In Figure 6.9a and 6.9c, the distributions of enclosed trajectory lengths are practically the same irrespective to varying sniffers and users in the zone. This validates the resilience of our framework against sniffer and user densities. Figure 6.9e demonstrates that the enclosed length of the actual trajectories decreases when increasing user speeds, as highly mobile users stay in the sniffers' range for shorter periods.

We find the Hellinger distance between distributions of *inclusiveness* and *ideal inclusiveness* for the quantitative analysis of the *correctness*. We get distances below 0.22 for all the scenarios; showing that the corresponding distribution pairs are very similar. This validates the *correctness* of our framework's bounded trajectories.

6.7.2.2 Bounds' width

To ensure the "utility" of the bounded trajectories, it's resulting *width* has to be relatively narrow. We define the *width* of the bounded trajectory of a user j, as:

$$Width\left(\mathscr{B}_{user_j}\right) = \frac{\sum_{k=1}^{M}\left(ActualTrajDist(\mathcal{CH}_{jk})\right)}{M} \tag{6.10}$$

where $ActualTrajDist$ calculates the distance between each of the enclosed points of the hull to the corresponding closest-in-time point on the actual trajectory. It gives us a measure of how "spread" are the hulls' enclosed

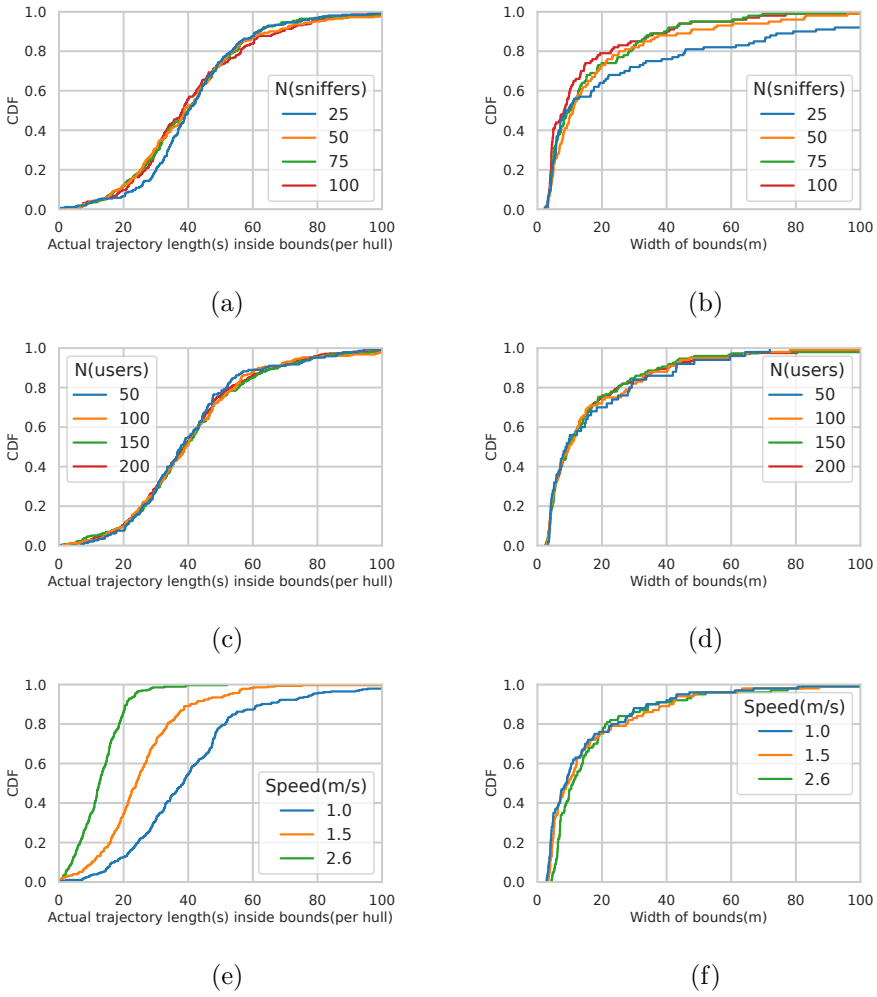


Figure 6.9: (a), (c), (e): *Inclusiveness* per hull. (b), (d), (f): Bounds' width.

points, i.e., bounded trajectory around the actual trajectory. We average out the "spread" over all the hulls of a user j to obtain the *width*.

Figure 6.9b, 6.9d, and 6.9f report the *width* of bounded trajectories, for scenarios with varying number of deployed sniffers, user-densities, and user speeds. We simultaneously have a look at the average total localization (distance-estimation) error for a user (cf. Alg. 7, line 13) in Fig. 6.10a, 6.10b, and 6.10c. It gives us an understanding of the behavior of *width* with changing parameters.

Fig. 6.10a shows a significant impact of the number of sniffers on the localization errors and, henceforth, the width. With the small number of sniffers in the deployment zone, we observe fewer user-probe requests in

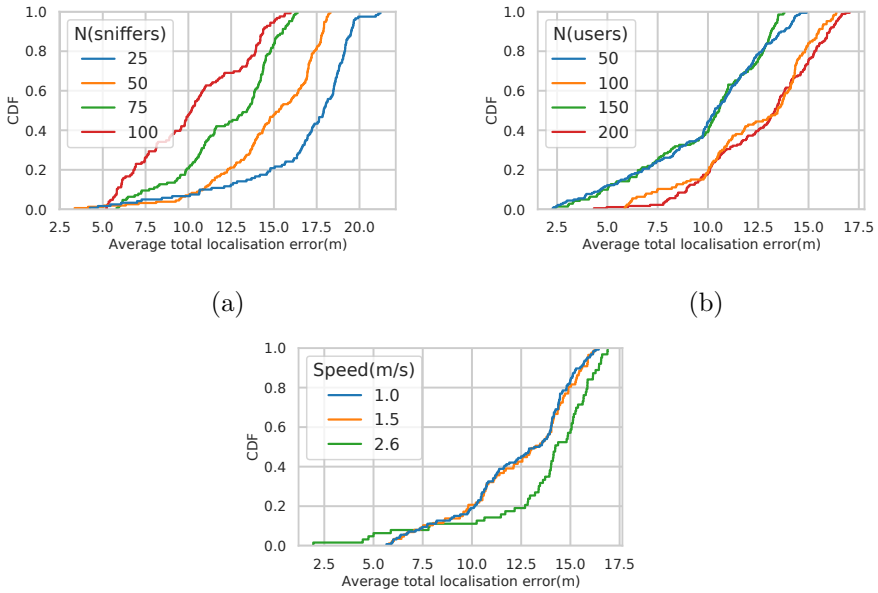


Figure 6.10: Average total localization error.

an observation set and a more significant span error on average. This leads to higher errors and increased width of bounds. Nevertheless, Figure 6.9b shows the resilience of bounds' width to the variation of average total distance-estimation errors as bounds' widths are almost the same for more than 25 sniffers.

Increasing the number of users and speed increases the magnitude of localization errors due to the higher number of missed probe-requests and lower observation sets, respectively. Figure 6.9d and 6.9f show that this increase in error amounts has a negligible decrease in the width. In essence, Figure 6.9b demonstrates that the *width* of bounded trajectories is less than $10m$ for 70% of users, when considering enough sniffers (i.e., 50 or more). This *width* is 8.3% of the maximum WiFi range ($140m$) seen in utilized large-scale *WiSurve dataset*. These results certify the "utility" of bounded trajectories.

6.8 Conclusion

Estimating trajectories of mobile users through non-intrusive measures remains an open problem. This leverages RSSI as a localization metric and *transforms* related errors in distance-estimation into an "insightful metric". We show distance-estimation errors are a combination of two distinct com-

ponents, i.e., span and environment errors. We then use this classification to find the bounds of a user location in short time intervals, which are then exploited to reveal the bounds of the user's trajectory. We observe that `Allies`' bounded trajectories do exhibit high *inclusiveness* and a small *width* of less than $10m$ for 70% of bounds. `Allies` can strengthen a wider range of privacy-compliant services, applications, and socioeconomic activities leveraging human mobility.

We believe that `Allies` framework can produce output directly usable by pervasive computing services. Furthermore, obtaining the user's *bounded* trajectory supports a range of applications such as human contact tracing, recommendation systems, human mobility analysis, and urban planning.

Chapter 7

Contributions conclusion and future perspectives

Contents

This thesis tackles privacy concerns related to the broadcast of public packets by WiFi and BLE devices, which expose users' MAC addresses, leading to tracking and behavioral profiling. The research presented in this thesis contributes to the field of privacy and security in wireless networks, shedding light on potential threats and proposing solutions to enhance user privacy in the context of MAC address associations.

7.1 Contributions summary

There were three objectives of the thesis, related to the three research questions that we pose in Section 1.1:

Objective 1: The first objective of the thesis was to *investigate the level of privacy risks associated with the existing design of public wireless frames, specifically WiFi probe-requests and BLE beacons.*

We delved into the examination of potential vulnerabilities, with a specific focus on MAC address association. The association of randomized MAC addresses in beacons and probe-requests to their sender devices poses a significant threat to user privacy. Through a thorough exploration of this problem, we have highlighted the urgent need for privacy-preserving public frames. During our investigation, we identified crucial flaws in the current

design of public frames that contribute to privacy concerns demanding immediate attention from the community.

By addressing these flaws and proposing potential solutions, our aim is to empower devices to effectively safeguard user privacy. The recommendations we have presented are rooted in the findings of our research.

In the existing literature, it is evident that the majority of association frameworks exploit the identified flaws to associate MAC addresses. However, before we proceed to identify additional potential vulnerabilities and propose a better-performing association framework, it is essential to assess the accuracy of current frameworks concerning varying input datasets.

We focused on WiFi probe-requests for MAC association because it presents a more challenging scenario for MAC association compared to BLE, primarily due to the high density of WiFi-enabled devices in public spaces. Moreover, it is worth noting that the MAC addresses of BLE devices have already been associated with very high accuracy.

Objective 2: The second objective was *to enquire about the already present WiFi MAC association frameworks in literature for their effectiveness and reliability.*

Our investigation into MAC association frameworks presented in the literature has raised concerns about the effectiveness of privacy preservation. These frameworks have demonstrated the capability to associate randomized MAC addresses with specific devices, potentially compromising user privacy.

We aimed to assess the reliability of these MAC association frameworks, particularly concerning the datasets used for their validation. We defined the reliability of a framework as its ability to maintain consistent accuracy across varying contexts where input probe-request datasets were collected. Our analysis revealed a significant discrepancy in the performances of these frameworks when faced with different contextual environments.

To address this issue, we introduced a novel metric that effectively measures the degree of randomization in evaluated datasets. This metric serves as a benchmark to assess the reliability of existing and new MAC association frameworks for datasets with similar or lower randomization complexities.

Subsequently, we proposed our own framework called `Bleach`, designed to associate randomized probe-requests advertised in the observation zone. The implementation of `Bleach` was rigorously tested using extensive datasets from various contextual scenarios. The results showed that `Bleach` is robust and significantly outperforms state-of-the-art works in terms of accuracy.

The findings of our research highlight the importance of considering the reliability of MAC association frameworks and their potential impact on user privacy. With `Bleach`, we present a promising solution that addresses these concerns and provides improved accuracy in associating randomized probe-requests. As wireless networks continue to evolve, preserving user privacy becomes increasingly crucial, and we believe that our work contributes

to the advancement of privacy-preserving technologies in this domain.

Objective 3: The last goal of the thesis, was to *infer user locations/trajectories from passively sniffed WiFi probe-request frames, utilizing our already developed effective MAC association framework,* `Bleach`.

Estimating the trajectories of mobile users through non-intrusive measures has been an ongoing challenge. This research tackles this problem by utilizing Received Signal Strength Indicator (RSSI) as a localization metric and transforming related errors in distance estimation into a novel insightful metric.

Through our study, we have identified that distance-estimation errors consist of two distinct components: span errors and environment errors. This classification has allowed us to determine the bounds of a user's location in short time intervals, which in turn, enables us to reveal the bounds of the user's trajectory.

Our proposed approach called `Allies`, has been evaluated and demonstrated promising results. The bounded trajectories generated by `Allies` exhibit a high degree of inclusiveness, indicating the accuracy in capturing the user's actual trajectory. Additionally, the small width of less than 10 meters for 70% of the bounds further validates the precision and reliability of our method.

The capabilities of `Allies` hold great potential for enhancing privacy-compliant services, applications, and socioeconomic activities that leverage human mobility. It contributes to addressing the challenge of estimating user trajectories while not being intrusive. The insights gained from this research can pave the way for improved location-based services, benefiting various domains that rely on human mobility information for decision-making and resource allocation.

7.2 Limitations

In this thesis, we indeed provide insights and showcase that privacy-related flaws in current public wireless frames can be exploited. However, there are a couple of limiting factors that we try to tackle our best through simulation frameworks: `SimBle` and `WiSurve`. Nonetheless, by acknowledging these limitations, we aim to provide a clear context for the scope and validity of our findings, while also highlighting the importance of further advancements in evaluating privacy-preserving techniques in wireless communication systems.

7.2.1 Privacy regulations in data collection

Collecting network traces through active methods that require user participation can be impractical for several reasons. First, it involves the cum-

bersome task of recruiting volunteers willing to participate in the data collection process. Additionally, specific applications need to be installed on the users' devices to enable the necessary data capture, which may deter many potential participants. These factors can lead to limited and biased data collection, making it challenging to obtain comprehensive and diverse datasets. To effectively address the challenges of this thesis, the availability of more real-world traces becomes critical.

Passive sniffing emerges as a promising alternative for trace collection in such scenarios. Passive sniffing involves listening to packets transmitted by devices without actively engaging with them. The collection process adheres to the principle of "legal capture" to ensure compliance with privacy and data protection laws in different countries. These regulations typically restrict the sniffing of packets without the proper authorization to safeguard user privacy and prevent unauthorized data interception . It requires collection advertisements for passing-by people, to give them the possibility to decide about their data being collected. As a result, passive sniffing on public channels too is difficult to achieve. This hinders the availability of large-scale traces needed to design and develop works in this thesis such as the benchmarks for MAC association and eventually the new association framework itself.

7.2.2 Lack of *ground-truth*

In this thesis, our research is constrained by two types of essential *ground-truth* information. The first type of ground-truth pertains to having precise knowledge of the history of randomized MAC addresses emitted by a device. This information plays a critical role in the evaluation of MAC association algorithms and device fingerprinting methods [1] [13] [12]. Without this ground-truth, accurately assessing the performance and efficacy of these proposed methods becomes challenging, potentially leading to biased or inaccurate conclusions.

The second type of ground-truth that we require for our research relates to obtaining real insight into the actual positions of users when they transmit probe-requests. It is very complicated to retrieve these ground-truth positions in real-world passive sniffing environments. This ground-truth is indispensable for evaluating the correctness and effectiveness of the "bounded trajectories" that we introduce in this thesis. Bounded trajectories play a crucial role in analyzing user movement patterns and tracking their positions over time. To properly validate the accuracy and reliability of these bounded trajectories, we need genuine user trajectories as ground-truth references.

7.3 Future perspectives

In this section, we discuss the potential research perspectives in the context of our analytical insights and technical contributions.

7.3.1 Short-term perspectives

1. **Implementing suggestions for confidentiality:** In this thesis, we suggest adding controlled noise to the information fields in the probe requests to reduce the effectiveness of fingerprinting attacks. We also recommend replacing entries in Preferred network lists (PNLs) advertised by the probe requests with pseudo-identifiers, which are already agreed upon by each client-AP pair. We could test and evaluate these suggestions in real hardware or in simulation, for ensuring confidentiality.

2. **New datasets for benchmarks:** We could demonstrate the complexities of our address association framework, `Bleach` over new data-collection scenarios and *benchmark* its performance. We could perform probe-request collection in dense public spaces, given the regulations are permitting.

3. **Studying the impact of conflicts on WiFi MAC association:** It would be interesting to investigate the rate of degradation of an association framework's performance with respect to randomization complexities (size of MAC conflicts) in the future when considering various classes of MAC association strategies. We could gradually increase the number of devices in the simulation or in controlled environments and see the evolution and fluctuations in the association accuracy over time.

4. **Realizing a real-world small-scale probe-request dataset:** The *ground-truth* for MAC addresses emitted from the same device is also lacking for passively collected datasets, which hinders the usage of a direct association accuracy metric. We need a data generation methodology to vary these devices' *modes* and *models* while preserving the identity of the sending device. We can linearly combine traces (in time) of different individual devices captured inside a Faraday cage (*labeled* dataset, cf. Section 2.5) while considering realistic packet losses and the sojourn time of devices. Using this methodology, we could obtain a real-world small-scale probe-request dataset with desired levels of heterogeneity. We could hence be able to realize *direct* accuracy metric and benchmarks, over custom scenarios, by varying the linear combinations.

7.3.2 Long-term perspectives

1. **A more robust and adaptive MAC randomization:** For a robust MAC randomization devices could be more collaborative and adaptive. Currently, in WiFi and BLE standards, the identifiers like MAC addresses change after a fixed interval regardless of the surroundings. It would be nice to have a model which adapts or predicts the timestamps of future changes, dynamically based on the surroundings that it listens to. For example, when fewer devices are around, the intuition is that devices should change their identifiers faster. This makes the tracking by the adversary more and more challenging. The model could either be analytical or artificial intelligence-based which considers based on the number of nearby public frames that a device perceives in recent time intervals.

2. **Integrating simulation frameworks with hardware provisions:** The simulation frameworks introduced in the thesis such as `WiSurve` and `SimBle` could be enriched with more hardware-related provisions to make the obtained traces more realistic. The key distribution for MAC randomization, for instance, could be done by using control messages (as done in BLE hardware) rather than the current pre-installation at the node. BLE stack could be enriched by the addition of different device pairing modes. As one of the aims is to emulate any real device, more and more vendor-specific information such as the time intervals between the subsequent MAC changes could be added.

3. **Combining BLE and WiFi for MAC association:** Currently the frameworks in literature as well as our proposal `Bleach`, focus on one of the wireless technologies (BLE and WiFi) for MAC address association. We could potentially combine the information from WiFi and BLE public frames to increase the success rate of association frameworks. Generally, a user might use BLE-based connected devices such as smartwatches and wireless headphones along with his mobile phone. The combined view of information from all devices both spatially in terms of receiving sniffers and temporally, could yield much more effective signatures. The resolution of conflicts for the MAC association is relatively easier as we could use the information if available from either technology to complement and reduce the size of effective conflicts, by discarding unlikely association pairs.

4. **Inferring contacts from *bounded* trajectories:** In this thesis, we stick to the formulation and the realization of *bounded* trajectories. Bounded trajectories can potentially solve the problem of passively inferring human contact tracing used for controlling and managing pandemics. Deploying sniffers for listening to emitted public packets by

devices in dense spaces could help us in obtaining the temporal and spatial overlap of bounded trajectories. The inferred contacts, assuming the MAC association, is likely to be more effective than current solutions such as applications actively sending beacons for contact-tracing [99, 100]. WiFi devices keep on sending probe-requests even when connected to an AP as they constantly look for better connections. This behavior aids the scale at which bounded trajectories could be retrieved from the area where passive sniffing is deployed.

Bibliography

[1] L. Jouans, A. C. Viana, N. Achir, and A. Fladenmuller, "Associating the randomized bluetooth mac addresses of a device," in *IEEE Annual Consumer Communications & Networking Conference, CCNC 2021, Las Vegas, NV, USA, January 9-12, 2021*, pp. 1–6, 2021.

[2] J. Tan and S.-H. Gary Chan, "Efficient association of wi-fi probe requests under mac address randomization," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pp. 1–10, 2021.

[3] C. Matte, M. Cunche, F. Rousseau, and M. Vanhoef, "Defeating mac address randomization through timing attacks," WiSec '16, (New York, NY, USA), Association for Computing Machinery, 2016.

[4] J. S. Hunter, "The exponentially weighted moving average," *Journal of quality technology*, vol. 18, no. 4, pp. 203–210, 1986.

[5] L. Pintor and L. Atzori, "A dataset of labelled device wi-fi probe requests for mac address de-randomization," *Computer Networks*, vol. 205, p. 108783, 2022.

[6] C. Matte, M. Cunche, F. Rousseau, and M. Vanhoef, "Defeating mac address randomization through timing attacks," in *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pp. 15–20, 2016.

[7] M. V. Barbera, A. Epasto, A. Mei, V. C. Perta, and J. Stefa, "Signals from the crowd: uncovering social relationships through smartphone probes," in *Proceedings of the 2013 conference on Internet measurement conference*, pp. 265–276, 2013.

[8] S. Jamil, S. Khan, A. Basalamah, and A. Lbath, "Classifying smartphone screen on/off state based on wifi probe patterns," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, pp. 301–304, 2016.

[9] C. Kolias, L. Copi, F. Zhang, and A. Stavrou, "Breaking ble beacons for fun but mostly profit," in *Proceedings of the 10th European Workshop on Systems Security*, pp. 1–6, 2017.

[10] J. Martin, T. Mayberry, C. Donahue, L. Foppe, L. Brown, C. Riggins, E. C. Rye, and D. Brown, "A study of mac address randomization in mobile devices and when it fails," *Proceedings on Privacy Enhancing Technologies*, vol. 2017, no. 4, pp. 365–383, 2017.

[11] M. Uras, R. Cossu, E. Ferrara, O. Bagdasar, A. Liotta, and L. Atzori, "Wifi probes sniffing: an artificial intelligence based approach for mac addresses de-randomization," in *2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pp. 1–6, IEEE, 2020.

[12] M. Cominelli, F. Gringoli, P. Patras, M. Lind, and G. Noubir, "Even black cats cannot stay hidden in the dark: Full-band de-anonymization of bluetooth classic devices," in *2020 IEEE Symposium on Security and Privacy (SP)*, pp. 534–548, 2020.

[13] J. Martin, D. Alpuche, K. Bodeman, L. Brown, E. Fenske, L. Foppe, T. Mayberry, E. Rye, B. Sipes, and S. Teplov, "Handoff all your privacy–a review of apple's bluetooth low energy continuity protocol," *PoPETs*, vol. 2019, no. 4, pp. 34–53, 2019.

[14] Grantz, Kyra H. and Meredith, Hannah R. and Cummings, Derek A. T. and Metcalf, C. Jessica E. and Grenfell, Bryan T. and Giles, John R. and Mehta, Shruti and Solomon, Sunil and Labrique, Alain and Kishore, Nishant and Buckee, Caroline O. and Wesolowski, Amy, "The use of mobile phone data to inform analysis of COVID-19 pandemic epidemiology," *Nature Communications*, vol. 11, no. 1, p. 4961, 2020.

[15] A. Trivedi, C. Zakaria, R. Balan, A. Becker, G. Corey, and P. Shenoy, "Wifitrace: Network-based contact tracing for infectious diseases using passive wifi sensing," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 5, no. 1, pp. 1–26, 2021.

[16] L. X. B. L. H. Petter, "Predictability of population displacement after the 2010 haiti earthquake," *Proceedings of the National Academy of Sciences*, vol. 109, no. 29, pp. 11576–11581, 2012.

[17] L. Wu, L. Yang, Z. Huang, Y. Wang, Y. Chai, X. Peng, and Y. Liu, "Inferring demographics from human trajectories and geographical context," *Computers, Environment and Urban Systems*, vol. 77, p. 101368, 2019.

[18] P. Robyns, B. Bonné, P. Quax, and W. Lamotte, "Noncooperative 802.11 mac layer fingerprinting and tracking of mobile devices," *Security and Communication Networks*, vol. 2017, 2017.

[19] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *Proceedings IEEE INFOCOM 2000. Conference on computer communications. Nineteenth annual joint conference of the IEEE computer and communications societies (Cat. No. 00CH37064)*, vol. 2, pp. 775–784, Ieee, 2000.

[20] A. Awad, T. Frunzke, and F. Dressler, "Adaptive distance estimation and localization in wsn using rssi measures," in *10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007)*, pp. 471–478, 2007.

[21] S. Sen, J. Lee, K.-H. Kim, and P. Congdon, "Avoiding multipath to revive inbuilding wifi localization," in *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '13, (New York, NY, USA), p. 249–262, Association for Computing Machinery, 2013.

[22] R. M. Susanti, K. M. Adhinugraha, S. Alamri, L. Barolli, and D. Taniar, "Indoor trajectory reconstruction using mobile devices," in *2018 IEEE 32nd international conference on advanced information networking and applications (AINA)*, pp. 550–555, IEEE, 2018.

[23] B. Yang, L. Guo, R. Guo, M. Zhao, and T. Zhao, "A novel trilateration algorithm for rssi-based indoor localization," *IEEE Sensors Journal*, vol. 20, no. 14, pp. 8164–8172, 2020.

[24] F. Bao, S. Mazokha, and J. O. Hallstrom, "Mobintel: Passive outdoor localization via rssi and machine learning," in *2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 247–252, 2021.

[25] M. Cominelli, F. Gringoli, P. Patras, M. Lind, and G. Noubir, "Even black cats cannot stay hidden in the dark: Full-band de-anonymization of bluetooth classic devices," in *2020 IEEE Symposium on Security and Privacy (SP)*, pp. 534–548, 2020.

[26] X. Gu, W. Wu, X. Gu, Z. Ling, M. Yang, and A. Song, "Probe request based device identification attack and defense," *Sensors*, vol. 20, no. 16, p. 4620, 2020.

[27] F. Guo and T.-c. Chiueh, "Sequence number-based mac address spoof detection," in *International Workshop on Recent Advances in Intrusion Detection*, pp. 309–329, Springer, 2005.

[28] M. Vanhoef, C. Matte, M. Cunche, L. S. Cardoso, and F. Piessens, "Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms," in *Proceedings of the 11th ACM on Asia conference on computer and communications security*, pp. 413–424, 2016.

[29] M. Čavojskỳ, M. Uhlar, M. Ivanis, M. Molnar, and M. Drozda, "User trajectory extraction based on wifi scanning," in *2018 6th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, pp. 115–120, IEEE, 2018.

[30] M. Kotaru and S. Katti, "Position tracking for virtual reality using commodity wifi," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 68–78, 2017.

[31] J. He and W. Yang, "Imar: Multi-user continuous action recognition with wifi signals," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 6, sep 2022.

[32] K. Jaffrès-Runser, G. Jakllari, T. Peng, and V. Nitu, "Crowdsensing mobile content and context data: Lessons learned in the wild," in *PerCom Workshops*, pp. 311–315, IEEE, 2017.

[33] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD dataset cambridge/haggle (v. 2009-05-29)." Downloaded from `https://crawdad.org/cambridge/haggle/20090529`, May 2009.

[34] L. Pajevic, V. Fodor, and G. Karlsson, "Predicting the users' next location from wlan mobility data," in *2018 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, pp. 61–66, 2018.

[35] Y. Li, J. Barthelemy, S. Sun, P. Perez, and B. Moran, "A case study of wifi sniffing performance evaluation," *IEEE Access*, vol. 8, pp. 129224–129235, 2020.

[36] B. SIG, *Bluetooth SIG. 2010. Bluetooth Core Specification v4.0.*

[37] B. SIG, *Specification of the Bluetooth System, Core v4.1.* 2013-03-12.

[38] B. SIG, *Specification of the Bluetooth System, Core v5.2.* 2019-12-31.

[39] 802-2014 Standard.

[40] M. Gast, *802.11 wireless networks: the definitive guide.* " O'Reilly Media, Inc.", 2005.

[41] B. SIG, *Specification of the Bluetooth System, Core v5.1.* 2019-01-21.

[42] J. K. Becker, D. Li, and D. Starobinski, "Tracking anonymized bluetooth devices," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, no. 3, pp. 50–65, 2019.

[43] G. Celosia and M. Cunche, "Saving private addresses: an analysis of privacy issues in the bluetooth-low-energy advertising mechanism," in *MOBIQUITOUS*, pp. 444–453, 2019.

[44] J. Martin, E. Rye, and R. Beverly, "Decomposition of mac address structure for granular device inference," in *Proceedings of the 32nd Annual Conference on Computer Security Applications*, pp. 78–88, 2016.

[45] J. Pang, B. Greenstein, R. Gummadi, S. Seshan, and D. Wetherall, "802.11 user fingerprinting," in *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pp. 99–110, 2007.

[46] M. Cunche, M.-A. Kaafar, and R. Boreli, "Linking wireless devices using information contained in wi-fi probe requests," *Pervasive and Mobile Computing*, vol. 11, pp. 56–69, 2014.

[47] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, pp. 93–108, 2005.

[48] C. Arackaparambil, S. Bratus, A. Shubina, and D. Kotz, "On the reliability of wireless fingerprinting using clock skews," in *Proceedings of the third ACM conference on Wireless network security*, pp. 169–174, 2010.

[49] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J. V. Randwyk, and D. Sicker, "Passive data link layer 802.11 wireless device driver fingerprinting.," in *USENIX Security Symposium*, vol. 3, pp. 16–89, 2006.

[50] B. Bezawada, M. Bachani, J. Peterson, H. Shirazi, I. Ray, and I. Ray, "Behavioral fingerprinting of iot devices," in *Proceedings of the 2018 workshop on attacks and solutions in hardware security*, pp. 41–50, 2018.

[51] F. Guo and T.-c. Chiueh, "Sequence number-based mac address spoof detection," in *Recent Advances in Intrusion Detection: 8th International Symposium, RAID 2005, Seattle, WA, USA, September 7-9, 2005. Revised Papers 8*, pp. 309–329, Springer, 2006.

[52] G. Chandrasekaran, J.-A. Francisco, V. Ganapathy, M. Gruteser, and W. Trappe, "Detecting identity spoofs in ieee 802.11 e wireless networks," in *GLOBECOM 2009-2009 IEEE Global Telecommunications Conference*, pp. 1–6, IEEE, 2009.

[53] G. Deak, K. Curran, and J. Condell, "Evaluation of smoothing algorithms for a rssi-based device-free passive localisation," in *Image Processing and Communications Challenges 2*, pp. 469–476, Springer, 2010.

[54] G. Chen, A. Carneiro Viana, M. Fiore, and C. Sarraute, "Complete Trajectory Reconstruction from Sparse Mobile Phone Data," *EPJ Data Science*, Oct. 2019.

[55] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *nature*, vol. 453, no. 7196, pp. 779–782, 2008.

[56] Y. Zheng, X. Xie, W.-Y. Ma, *et al.*, "Geolife: A collaborative social networking service among user, location and trajectory.," *IEEE Data Eng. Bull.*, vol. 33, no. 2, pp. 32–39, 2010.

[57] G. Bernard, C. Faucher, and K. Bertet, "Towards reconstruction of human trajectories in indoor environments.," in *EKAW (Posters & Demos)*, pp. 37–40, 2018.

[58] A. Noulas, S. Scellato, N. Lathia, and C. Mascolo, "Mining user mobility features for next place prediction in location-based services," in *2012 IEEE 12th international conference on data mining*, pp. 1038–1043, IEEE, 2012.

[59] S. Shang, L. Chen, Z. Wei, C. S. Jensen, K. Zheng, and P. Kalnis, "Trajectory similarity join in spatial networks," *Proceedings of the VLDB Endowment*, vol. 10, no. 11, 2017.

[60] A. Bjerre-Nielsen, K. Minor, P. Sapieżyński, S. Lehmann, and D. D. Lassen, "Inferring transportation mode from smartphone sensors: Evaluating the potential of wi-fi and bluetooth," *PloS one*, vol. 15, no. 7, p. e0234003, 2020.

[61] C. Comito, D. Falcone, and D. Talia, "Mining human mobility patterns from social geo-tagged data," *Pervasive and Mobile Computing*, vol. 33, pp. 91–107, 2016.

[62] Y. Wang, Q. Ye, J. Cheng, and L. Wang, "Rssi-based bluetooth indoor localization," in *2015 11th International Conference on Mobile Ad-hoc and Sensor Networks (MSN)*, pp. 165–171, IEEE, 2015.

[63] W.-H. Kuo, Y.-S. Chen, G.-T. Jen, and T.-W. Lu, "An intelligent positioning approach: Rssi-based indoor and outdoor localization scheme in zigbee networks," in *2010 International Conference on Machine Learning and Cybernetics*, vol. 6, pp. 2754–2759, IEEE, 2010.

[64] E. Goldoni, L. Prando, A. Vizziello, P. Savazzi, and P. Gamba, "Experimental data set analysis of rssi-based indoor and outdoor localization in lora networks," *Internet Technology Letters*, vol. 2, no. 1, p. e75, 2019.

[65] S. J. Halder, P. Giri, and W. Kim, "Advanced smoothing approach of rssi and lqi for indoor localization system," *International Journal of Distributed Sensor Networks*, vol. 11, no. 5, p. 195297, 2015.

[66] X. Hou, T. Arslan, and J. Gu, "Indoor localization for bluetooth low energy using wavelet and smoothing filter," in *2017 International Conference on Localization and GNSS (ICL-GNSS)*, pp. 1–6, IEEE, 2017.

[67] J.-H. Kim, J.-H. Seong, Y.-S. Ha, and D.-H. Seo, "Improved adaptive smoothing filter for indoor localization using rssi," *Journal of the Korean Society of Marine Engineering*, vol. 39, no. 2, pp. 179–186, 2015.

[68] A. S. Paul and E. A. Wan, "Rssi-based indoor localization and tracking using sigma-point kalman smoothers," *IEEE Journal of Selected Topics in Signal Processing*, vol. 3, no. 5, pp. 860–873, 2009.

[69] H.-S. Ahn and W. Yu, "Environmental-adaptive rssi-based indoor localization," *IEEE Transactions on Automation Science and Engineering*, vol. 6, no. 4, pp. 626–633, 2009.

[70] M. V. Barbera, A. Epasto, A. Mei, S. Kosta, V. C. Perta, and J. Stefa, "CRAWDAD dataset sapienza/probe-requests (v. 2013-09-10)." Downloaded from `https://crawdad.org/sapienza/probe-requests/20130910`, Sept. 2013.

[71] Stijn Geysen, https://gitlab.com/Stijng/ns3-ble-module/-/tree/master/ble.

[72] E. McKinion and A. Lin, "Evaluation of security flaws in the current probe request design and proposed solutions," in *International Conference on Cyber Warfare and Security*, p. 529, Academic Conferences International Limited, 2017.

[73] "Ieee standard for information technology–telecommunications and information exchange between systems - local and metropolitan area networks–specific requirements - part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications - redline," *IEEE Std 802.11-2020 (Revision of IEEE Std 802.11-2016) - Redline*, pp. 1–7524, 2021.

[74] J. Freudiger, "How talkative is your mobile device? an experimental study of wi-fi probe requests," in *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pp. 1–6, 2015.

[75] K. Calder, "Statistical inference," *New York: Holt*, 1953.

[76] J. Andersen, T. Rappaport, and S. Yoshida, "Propagation measurements and models for wireless communications channels," *IEEE Communications Magazine*, vol. 33, no. 1, pp. 42–49, 1995.

[77] L. L. Long III and M. Srinivasan, "Walking, running, and resting under time, distance, and average speed constraints: optimality of walk–run–rest mixtures," *Journal of The Royal Society Interface*, vol. 10, no. 81, p. 20120980, 2013.

[78] B. Alexander, T. Ivan, and B. Denis, "Analysis of noisy signal restoration quality with exponential moving average filter," in *2016 International Siberian Conference on Control and Communications (SIBCON)*, pp. 1–4, 2016.

[79] T. I. Chowdhury, M. M. Rahman, S.-A. Parvez, A. K. M. M. Alam, A. Basher, A. Alam, and S. Rizwan, "A multi-step approach for rssi-based distance estimation using smartphones," in *2015 International Conference on Networking Systems and Security (NSysS)*, pp. 1–5, 2015.

[80] J. A. Jayakody, S. Lokuliyana, D. Chathurangi, D. Vithana, *et al.*, "Indoor positioning: Novel approach for bluetooth networks using rssi smoothing," *International Journal of Computer Applications*, vol. 137, no. 13, pp. 26–32, 2016.

[81] E.-E.-L. Lau and W.-Y. Chung, "Enhanced rssi-based real-time user location tracking system for indoor and outdoor environments," in *2007 International Conference on Convergence Information Technology (ICCIT 2007)*, pp. 1213–1218, 2007.

[82] J. Miranda, R. Abrishambaf, T. Gomes, P. Gonçalves, J. Cabral, A. Tavares, and J. Monteiro, "Path loss exponent analysis in wireless sensor networks: Experimental evaluation," in *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*, pp. 54–58, 2013.

[83] O. Y. Olajide and S. M. Yerima, "Channel path-loss measurement and modeling in wireless data network (ieee 802.11 n) using artificial neural network," *European Journal of Electrical Engineering and Computer Science*, vol. 4, no. 1, 2020.

[84] S. G. Kwak and J. H. Kim, "Central limit theorem: the cornerstone of modern statistics," *Korean journal of anesthesiology*, vol. 70, no. 2, p. 144, 2017.

[85] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[86] Y. Zhou, "An efficient least-squares trilateration algorithm for mobile robot localization," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3474–3479, 2009.

[87] F. Thomas and L. Ros, "Revisiting trilateration for robot localization," *IEEE Transactions on robotics*, vol. 21, no. 1, pp. 93–101, 2005.

[88] S. Rao, "Comments on efficient solution and performance analysis of 3-d position estimation by trilateration," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, no. 2, p. 681, 1998.

[89] I. Coope, "Reliable computation of the points of intersection of n spheres in $\hat{rn}$," *ANZIAM Journal*, vol. 42, pp. C461–C477, 2000.

[90] W. H. Foy, "Position-location solutions by taylor-series estimation," *IEEE transactions on aerospace and electronic systems*, no. 2, pp. 187–194, 1976.

[91] W. Navidi, W. S. Murphy Jr, and W. Hereman, "Statistical methods in surveying by trilateration," *Computational statistics & data analysis*, vol. 27, no. 2, pp. 209–227, 1998.

[92] W. Hu and W. H. Tang, "Automated least-squares adjustment of triangulation-trilateration figures," *Journal of surveying engineering*, vol. 127, no. 4, pp. 133–142, 2001.

[93] M. Pent, M. Spirito, and E. Turco, "Method for positioning gsm mobile stations using absolute time delay measurements," *Electronics Letters*, vol. 33, no. 24, pp. 2019–2020, 1997.

[94] J. C. Spall, *Introduction to stochastic search and optimization: estimation, simulation, and control*, vol. 65. John Wiley & Sons, 2005.

[95] D. Mitra, F. Romeo, and A. Sangiovanni-Vincentelli, "Convergence and finite-time behavior of simulated annealing," *Advances in applied probability*, vol. 18, no. 3, pp. 747–771, 1986.

[96] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility: an overview," in *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*, ThinkMind, 2011.

[97] S. Kun, W. Ping, and L. Yingze, "Path loss models for suburban scenario at 2.3ghz, 2.6ghz and 3.5ghz," in *2008 8th International Symposium on Antennas, Propagation and EM Theory*, pp. 438–441, 2008.

[98] J. Oosterhoff and W. R. van Zwet, "A note on contiguity and hellinger distance," in *Selected Works of Willem van Zwet*, pp. 63–72, Springer, 2012.

[99] M. Cunche, A. Boutet, C. Castelluccia, C. Lauradoux, and V. Roca, *On using Bluetooth-Low-Energy for contact tracing*. PhD thesis, Inria Grenoble Rhône-Alpes; INSA de Lyon, 2020.

[100] P. C. Ng, P. Spachos, and K. N. Plataniotis, "Covid-19 and your smartphone: Ble-based smart contact tracing," *IEEE Systems Journal*, vol. 15, no. 4, pp. 5367–5378, 2021.

[101] F. I. P. S. P. 197, "Announcing the advanced encryption standard (aes)," vol. 21, p. 51, 2001.

[102] Jason Lee, Encryptions.

[103] A. K. Mishra, A. Carneiro Viana, and N. Achir, "Introducing benchmarks for evaluating user-privacy vulnerability in WiFi," in *VTC2023-Spring - IEEE 97th Vehicular Technology Conference*, (Florence, Italy), June 2023.

[104] A. K. Mishra, A. Carneiro Viana, and N. Achir, "Do wifi probe-requests reveal your trajectory?," in *2023 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, 2023.

[105] A. K. Mishra, A. Carneiro Viana, N. Achir, and C. Palamidessi, "Public wireless packets anonymously hurt you," in *2021 IEEE 46th Conference on Local Computer Networks (LCN)*, pp. 649–652, 2021.

[106] A. K. Mishra, A. Carneiro Viana, and N. Achir, "SimBle: Comment générer des traces réelles Bluetooth conformes aux recommandations de préservation de la vie privée ?," in *ALGOTEL 2021 - 23èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, (La Rochelle, France), 2021.

[107] A. K. Mishra, A. C. Viana, and N. Achir, "Simble: Generating privacy preserving real-world ble traces with ground truth," *arXiv preprint arXiv:2101.11728*, 2021.

[108] A. k. Mishra, S. Ayoubi, G. Grassi, and R. Teixeira, "Nemfi: Record-and-replay to emulate wifi," *SIGCOMM Comput. Commun. Rev.*, vol. 51, p. 2–8, jul 2021.

Appendix A

SimBle: technical details

A.1 Address generation

The format of a Resolvable private address is shown in Figure A.1. The resolvable private address is generated with the IRK and a 24-bit number known as *prand*. We see that it could be mainly divided into two blocks of 24 bits each. The first block consists of a 24-bit hash introduced in [Alg. 8 line 7]. `Simble` incorporates the AES (Advanced Encryption Standard) support as it is recommended by the standard [38] for encrypting the plain-text data into ciphered block [101] [102] in the process of randomized MAC address generation.

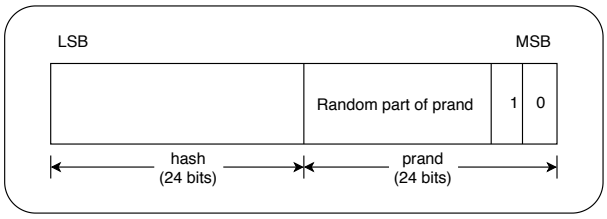


Figure A.1: Format of a Resolvable Private Address

The second block consists of *prand*. *Prand* in the case of Resolvable private address has two most significant bits as 1 and 0 as shown in Figure A.1. The random part of *prand* must consist of at least one bit as 0 and one bit as 1. We discover in detail the generation of the Resolvable private address by *PrivacyManager* in [Alg. 8].

Each of the nodes in `Simble` has an instance of *PrivacyManager* as illustrated earlier in Figure A.1. [Alg. 8] performs two major functions. GENERATE in [Alg. 8 line 1], takes as input the *IRK* and generates a resolvable private address for that node. While UPDATE [Alg. 8 line 1] take care of necessary calls to update a device's MAC address according to the user-specified BLE standard and device class that we are trying to emulate.

Whenever GENERATE is called we generate a 24 bits value with the two most significant bits as *10*. The rest of the bits are random and we use this value as *prand*, the trailing half a resolvable private address [Alg. 8 line 2]. This generated *prand* is then padded by 104 null bits such that

Algorithm 8 SimBle's Resolvable Private Address generation

1: **procedure** GENERATE(IRK) ▷ Input variable

▷ Prepare encryption inputs

2: $prand \leftarrow genPrand()$
3: $padding \leftarrow genPaddingBits(104)$
4: $plaintext \leftarrow Concatenate(padding, prand)$

▷ AES encryption

5: $aesobj \leftarrow AES(IRK)$
6: $ciphertext \leftarrow aesobj.getEncrypt(plaintext)$

▷ Getting MAC address

7: $prunedcipher \leftarrow getLeastSigBits(ciphertext, 24)$
8: $macstr \leftarrow Concatenate(prunedcipher, prand)$
9: $macaddr \leftarrow toMacHex(macstr)$

10: **return** ▷ Returns a Resolvable Private Address

11: **procedure** UPDATE($randInterval, swapDelay, IRK$) ▷ Input variables
12: $roundIndex = getCurrentRoundIndex()$
13: $macDevice =$ GENERATE(IRK)

▷ Check if this call is just after device initialization

14: **if** $roundIndex == 1$ **then** ▷ Calculate time offset for recursive callback
15: $nextUpOffset \leftarrow getURV(0, randInterval) + swapDelay$
16: **else**
17: $nextUpOffset \leftarrow randInterval + swapDelay$

▷ Schedule a callback after offset expires

18: $incRoundIndex()$
19: Schedule(UPDATE, nextUpOffset)

the most significant byte of the *prand* becomes the most significant byte of padding [Alg. 8 line 4]. We call this value *plaintext* as it is given as input for encryption. Then, we generate an instance of the AES algorithm initialized with the IRK of the current node [Alg. 8 line 5]. AES instance then encrypts the *plaintext* to generate 128 bits of *ciphertext* [Alg. 8 line 6].

We take 24 most significant bits of *ciphertext* [Alg. 8 line 7] and concatenate to the earlier generated *prand* to generate a string of 48 bits [Alg. 8 line 4]. The generated string is then finally formatted in IEEE 802.11 MAC address format to produce a resolvable private address [Alg. 8 line 9].

Once the randomized MAC address is generated, the next step is to change this address dynamically while respecting the standard. This is done by the UPDATE function of *PrivacyManager* which takes three arguments. One of them is *IRK*, the identity resolving key of the node, which we have already discussed. The other two arguments are device-dependent with the freedom to users for allocating any specific values. They are as follows:

- **randInterval:** This is the time after which a specific device generates a new resolvable private address. In BLE 4.1 standard [37], the most prevalent Bluetooth standard in current mobile devices, this interval is fixed to 15 minutes. However, in the most recent release, BLE 5.2 [38], the vendor is flexible to randomize the MAC address before the mark of 15 minutes. But standard recommends not to update the addresses too frequently as it might affect the paired devices' performance. It is due to an increase in the number of control messages that need to be exchanged after generating a new address. `Simble` takes the BLE standard and device class as input from the user at the initialization of nodes to calculate the respective *randInterval* value.

- **swapDelay:** It is introduced to emulate the behavior of the device in practice. We see from the experiments that devices take some time before they develop a new randomized address and advertise. This delay is caused due to resources used in address generation and in updating the current MAC level state. *swapDelay* could be device-specific. We empirically choose the value to be 10 times the *frequency of transmitting beacons*. We do this after measuring the value of this delay in experiments done on a large set of BLE devices broadcasting beacons.

On receiving the input arguments, UPDATE first checks the iteration index of this call and stores it as *roundIndex* [Alg. 8 line 12]. For calls to UPDATE, *roundIndex* has the value greater than or equal to 1. It distinguishes the two states in which a node can generate a new address. The first state(*roundIndex*=1) is when a node goes for obtaining a new address just after spawning inside the simulation. While the second state(*roundIndex*>1) is when the node requests an address after the expiration of the old one. GENERATE is called from UPDATE to assign the device a new resolvable private address [Alg. 8 line 13].

After assigning the randomized address, UPDATE calculates the duration for which this address would be valid. If the device has called UPDATE for the first round, then we calculate this duration by taking a random value

out of uniform random variable distribution in [0, *randInterval*] and adding the *swapDelay* to this value [Alg. 8 line 15].

We do this to respect the standard guidelines for setting the address expiration timers as discussed in Section 2.2.1. Else if the device has already changed its MAC address since spawning, then we assign the offset to be the sum of *randInterval* and *swapDelay* [Alg. 8 line 17].

Finally, we increase the *roundIndex* and schedule a recursive callback to UPDATE after the expiration of offset that we just calculated above [Alg. 8 line 19] in order to get resolvable private addresses during the simulation time.

A.2 Address resolution

Generation of MAC address is not sufficient for a BLE device. The receiving node must be able to "resolve" or associate the private address with the sending device's identity. A Resolvable private address may be resolved if the sending device's IRK is available to the receiver. If the address is resolved, the receiving device can associate this address with the peer device.

To support this privacy-preserving feature, we need to figure out solutions to two major questions inside a device; how to resolve a private address of a device? And, where do we need to check the validity of the private address in the packet being handled inside `Simble`?

The solution to the first question is given by RESOLVE [Alg. 9 line 1] while CHECKVALIDATION [Alg. 9 line 16] answers the second question that we arise above.

As briefly stated earlier, RESOLVE returns a tuple consisting of (*resolved*, *resIDAdd*). Here *resolved* states if the resolution attempt of the *privateAddress* was successful or not. If the private address is resolved then *resIDAdd* consists of the Identity Address of the node creating the private address, else it is an empty string in the returned pair.

Whenever a node receives a resolvable private address, the corresponding *PrivacyManager* calls RESOLVE with *privateAddress* and *irkIAddPairList* as input. While *privateAddress* is the sending device's randomized MAC address, *irkIAddPairList* is the locally maintained list of (*IRK*, *Identity Address*) pairs at the resolving node, as described in section 3.2.1.3.

RESOLVE first extracts *hash* and *prand* part of the private address [Alg. 9 line 3] as described earlier in Figure A.1. We pad 104 null bits to the extracted *senderPrand* such that the most significant byte of the *senderPrand* becomes the most significant byte of *plaintext*, which is the resulted byte array after padding.

Before considering a *privateAddress* to be resolved, the handling node checks the validity of the address. The valid private address refers to the address which was resolved using one of the *IRK's* in the list available at the

Algorithm 9 SimBle's Resolvable Private Address resolution

```
1: procedure RESOLVE(privateAddress,
   irkIAddPairList)                                  ▷ Input variable

                                  ▷ Extract hash and random part of privateAddress
2:     senderHash ← extractHash(privateAddress)
3:     senderPrand ← extractPrand(privateAddress)
4:     padding ← genPaddingBits(104)
5:     plaintext ← Concatenate(padding, senderPrand)
6:     resolved ← FALSE
7:     resIDAdd ← NULLSTR

                                  ▷ Check if Sender hash is valid
8:     for IRK, IDAdd in irkIAddPairList do
9:         aesobj ← AES(IRK)
10:        ciphertext ← aesobj.getEncrypt(plaintext)
11:        localHash ← getLeastSigBits(ciphertext, 24)
12:        resolved ← isEqual(localHash, senderHash)
13:        if resolved == TRUE then
14:            resIDAdd ← IDAdd

                                  ▷ Return resolved status & Identity Address
15:    return (PAIR(resolved, resIDAdd))

16: procedure CHECKVALIDATION

    ▷ Call RESOLVE to validate private address if any of the function
    calls below is triggered in Simble

17:    if
18:        BroadbandManager:LinkExists(),
   GetLinkManager(), GetLink()
19:        LinkController:CheckReceivedAckPacket()
   then
20:        RESOLVE(privateAddress, irkIAddPairList)
```

resolving node. To get this verification, we first take out a (*IRK*: *Identity Address*) pair from the *irkIAddPairList*. We generate an instance of the AES algorithm initialized with the IRK from the current pair [Alg. 9 line 9]. AES instance then encrypts the *plaintext* to generate 128 bits of *ciphertext* [Alg. 9 line 10]. We take 24 most significant bits of *ciphertext* to generate the

localHash. If the value of *localHash* matches the earlier extracted *senderHash* [Alg. 9 line 2] for any of the iterations, RESOLVE successfully returns the (TRUE, *Identity Address*) pair. Otherwise, resolution is considered a failure and RESOLVE returns the (FALSE, *" "*) pair.

After resolving a private address, we look into the framework of `Simble` to investigate the modules that need address resolution. We identify two modules that need to call *PrivacyManager*'s RESOLVE procedure: *BroadbandManager* and *LinkController* through CHECKVALIDATION [Alg. 9 line 18]. Whenever *BroadbandManager* receives a packet from the *NetDevice*, RESOLVE is recalled in two cases. The first is when it checks/tries to fetch the link. The second is when it requests the *LinkManager* to the destination node. We do this to ensure that the identity address resolved by the node suggested by the destination address matches the identity address of the existing link. Finally, CHECKVALIDATION also needs to check if the sender address of the correctly received packet by the *LinkController* could be resolved using one of the stored *IRK*'s at the receiver [Alg. 9 line 19].

For validation of `Simble`, it is fundamental to evaluate the functionalities of the introduced *PrivacyManager*. Therefore resolvable private address generation and resolution must be validated. Specifically, we must show that generated randomized addresses are very close to what real-world devices advertise. Also, we have to show that BLE data communication continues flawlessly between the paired devices even when they change their advertised MAC address. In this case, we assume that the devices have exchanged each other's *IRK* during initialization. All the MAC addresses shown in the thesis are hashed using SHA-256 and truncated to the first 8 bytes for illustration purposes.

Appendix B

List of Publications

Publications

P$_1$ A. K. Mishra, A. Carneiro Viana, and N. Achir, "Introducing benchmarks for evaluating user-privacy vulnerability in WiFi" 2023 IEEE 97th Vehicular Technology Conference: VTC-Spring, Florence, Italy 20-23 June 2023. [103]

P$_2$ A. K. Mishra, A. Carneiro Viana, and N. Achir, "Do WiFi Probe-Requests Reveal Your Trajectory?," 2023 IEEE Wireless Communications and Networking Conference (WCNC), Glasgow, United Kingdom, 2023, pp. 1-6. [104]

P$_3$ A. K. Mishra, A. Carneiro Viana, N. Achir and C. Palamidessi, "Public Wireless Packets Anonymously Hurt You," 2021 IEEE 46th Conference on Local Computer Networks (LCN) [Doctoral-track - Promising ideas], Edmonton, AB, Canada, 2021. [105]

P$_4$ A. K. Mishra, A. Carneiro Viana, N. Achir. SimBle: Comment générer des traces réelles Bluetooth conformes aux recommandations de préservation de la vie privée ?. ALGOTEL 2021 - 23èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications, 2021, La Rochelle, France. [106]

P$_5$ A. K. Mishra, A. Carneiro Viana, and N. Achir, "SimBle: Generating privacy preserving real-world BLE traces with ground truth." arXiv preprint arXiv:2101.11728 (2021). [107]

Under Submission

P$_6$ A. K. Mishra, A. Carneiro Viana, and N. Achir, "Bleach: From WiFi probe-request signatures to MAC association." (Journal)

P$_7$ A. K. Mishra, A. Carneiro Viana, and N. Achir, "Allies: Inferring users' bounded trajectories from WiFi passive sniffing." (Journal)

Outside the scope of this thesis

P$_8$ A. K. Mishra, S. Ayoubi, G. Grassi, and R. Teixeira. 2021. NemFi: record-and-replay to emulate WiFi. SIGCOMM Computer Communication Review 51, 3 (July 2021), 2–8. [108]

DOCTORAL SCHOOL

Titre : Révéler et exploiter les vulnérabilités liées à confidentialité dans les paquets sans fil publics des utilisateurs

Mots clés : Confidentialité, Réseaux sans fil, Paquets publics, Reconstruction de trajectoire, Association MAC

Résumé : La prolifération croissante des dispositifs réseau WiFi (Wireless Fidelity) et Bluetooth Low Energy (BLE) diffusant des paquets publics non chiffrés par voie hertzienne a suscité de plus en plus de préoccupations concernant la confidentialité des utilisateurs. Ces paquets publics se composent de trames de gestion, telles que les trames *probe-requests* et les trames *beacons*, nécessaires aux appareils pour découvrir les réseaux sans fil disponibles et améliorer l'expérience utilisateur. La révélation de l'adresse MAC d'un appareil à travers les paquets publics permet de suivre l'appareil et de réaliser un profilage comportemental. Les appareils modernes changent périodiquement leur adresse MAC annoncée, de manière aléatoire (randomisation). Néanmoins, des attaques contre la randomisation des adresses MAC ont été menées, démontrant que des adresses aléatoires provenant d'un appareil peuvent être associées avec aussi peu d'informations que les horodatages de leurs paquets publics.
Dans cette thèse, nous identifions les principales failles qui peuvent conduire à l'association des adresses MAC. Afin de mesurer la gravité des failles identifiées en examinant les performances des frameworks d'attaques d'association MAC existantes, nous avons besoin de traces d'écoute passive à grande échelle des paquets publics avec une "vérité terrain" concernant les adresses aléatoires générés par le même équipement. Finalement, nous mettons en évidence ces défauts et en proposons un outil de simulation pour générer des traces d'écoute passive WiFi et BLE à grande échelle. Nous révélons que la randomisation actuelle des appareils n'est pas suffisamment efficace et doit être révisée.
En plus de l'identification des failles clés, nous avons mené des études de cas afin d'en tirer des benchmarks pour évaluer les performances de tout framework d'association. Finalement, nous proposons et évaluons un nouveau framework d'association afin de déterminer ses performances attendues avec divers ensemble de traces en entrée.
Une fois l'association MAC obtenue, nous révélons l'inférence des emplacements des utilisateurs à partir des trames *probe-requests* captées passivement. Nous proposons un nouveau concept appelé "trajectoires bornées". Une trajectoire bornée désigne une zone où un utilisateur particulier est probablement présent dans le temps.

Title : Revealing and exploiting privacy vulnerabilities in users' public wireless packets

Keywords : Privacy, Wireless networks, Public packets, Trajectory reconstruction, MAC association

Abstract : The increasing proliferation of Wireless Fidelity (WiFi) and Bluetooth Low Energy (BLE) networked devices broadcasting over-the-air unencrypted public packets has raised growing concerns regarding users' privacy. Such public packets consist of management frames, like probe-requests and beacons, necessary for devices to discover available wireless networks and enhance user experience. Revealing the MAC address of a device through public packets allows adversaries to follow the device and do behavioral profiling. Modern devices periodically change/randomize their advertised MAC addresses. Nevertheless, attacks on MAC address randomization have been carried out, demonstrating that randomized addresses from a device can be associated with as little information as the timestamps of their advertised public packets.
In this thesis, we identify key flaws that lead to the MAC association. To measure the severity of identified flaws by looking at the performance of current MAC association attacks, we need large-scale traces of public packets with "ground truth" information regarding randomized addresses from the same device. We assert the flaws by employing our proposed simulation framework to generate large-scale WiFi and BLE passive sniffing traces. We reveal that current device randomization is ineffective and needs revision.
In addition to key flaws identifications, we conduct case studies and introduce benchmarks for evaluating the performance of any association framework. We show the need for a new and effective WiFi MAC association framework, and finally, we develop and benchmark a novel association framework to determine its expected performance with any new input probe-request dataset.
Once achieving effective MAC association, we reveal the inference of user locations from passively sniffed probe-requests. We propose a novel concept called "bounded trajectories." Bounded trajectory refers to an area where a particular user is probable to be present across time.

Institut Polytechnique de Paris
91120 Palaiseau, France