July 2020

The Limits of Location Privacy in Mobile Devices

Keen Yuun Sung

University of Massachusetts Amherst

THE LIMITS OF LOCATION PRIVACY IN MOBILE DEVICES

A Dissertation Presented

by

KEEN YUUN SUNG

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2020

College of Information and Computer Sciences

THE LIMITS OF LOCATION PRIVACY IN MOBILE DEVICES

A Dissertation Presented

by

KEEN YUUN SUNG

Approved as to style and content by:

DEDICATION

To Mom and Dad.

ACKNOWLEDGMENTS

I would like to thank my collaborators, Marc Liberatore, Erik Learned-Miller, Hamed Soroush, Joydeep Biswas, Mariya Zheleva, JianYi Huang, and Mark Corner, whom have not only made it possible for our research, but also provided invaluable advice and encouragement. Thank you to Dennis Goeckel for providing vital feedback as part of my committee, and thank you to Florin Dolcos for teaching me how to be a scientist as an undergraduate. And especially thank you to Brian Levine for being a collaborator, mentor, and advisor throughout these years, and for being so patient with me in my march towards graduation.

Many colleagues and friends stood by me during both rewarding and difficult times. Emma, Pat, Pinar, Luis, Jesse, Arun, and Shinyoung, thank you for making the journey less lonely. Rachee, thanks for being the voice of optimism, and getting me addicted to coffee. My roommates Larkin, Amanda, Kevin, Myungha, Li, and Joie were welcome company after a long day at the lab. Mike, Paul, Carey, Luka, and Jill, you were the ones who made Amherst a beautiful place to live.

Petros, Gabi, Claudia, Crystal, Tae, Bailey, and Kenzie — though we were apart through these years, we grew up together during this time, and I appreciate you all. David, I could always count on you for your frankness. Thanks for syncing up with my sleep schedule by moving to the west coast.

Thank you Elizabeth for being by my side from start to finish.

Finally, my parents have unconditionally loved and supported me morally and materially, and have consistently been willing listeners and advisors themselves. Thank you.

ABSTRACT

THE LIMITS OF LOCATION PRIVACY IN MOBILE DEVICES

MAY 2020

KEEN YUUN SUNG

B.Sc., UNIVERSITY OF ALBERTA

M.S., UNIVERSITY OF MASSACHUSETTS AMHERST

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Brian Neil Levine

Mobile phones are widely adopted by users across the world today. However, the privacy implications of persistent connectivity are not well understood. This dissertation focuses on one important concern of mobile phone users: location privacy.

I approach this problem from the perspective of three adversaries that users are exposed to via smartphone apps: the mobile advertiser, the app developer, and the cellular service provider. First, I quantify the proportion of mobile users who use location permissive apps and are able to be tracked through their advertising identifier, and demonstrate a mark and recapture attack that allows continued tracking of users who hide these identifiers. Ninety-five percent of the 1500 devices we tested were susceptible to this attack. We successfully identified 49% of unlabelled impressions from iOS devices, and 59% from Android, with a budget of only \$5 per day, per user. Next, I evaluate an attack wherein a remote server discovers a user's traveled path without permission, simply by analyzing the throughput of the connection to the user over time. In these experiments, a remote attacker can distinguish a user's route among four paths within a University campus with 77% accuracy, and among eight paths surrounding the campus with 83% accuracy. I then propose a protocol for anonymous cell phone usage, which obviates the need for users to trust telecoms with their location, and I evaluate its efficacy against a passive location profiling attack used to infer identity. According to these simulations, even one day is enough to identify one device from among over a hundred with greater than 50% accuracy. To mitigate location profiling attacks, users should change these identifiers every ten minutes and remain offline for 30 seconds, to reduce their identifiability by up to 45%. I conclude by summarizing the key issues in mobile location privacy today, immediate steps that can be taken to improve them, and the inherent privacy costs of remaining constantly connected.

TABLE OF CONTENTS

Page

ACKNOWLEDGMENTS . . . . **v**

ABSTRACT . . . . **vi**

LIST OF FIGURES . . . . **xii**

CHAPTER

1. INTRODUCTION **1**

1.1 Approach to studying privacy and utility 2
1.2 Overview of studies 3

1.2.1 Re-identifying mobile devices in advertising networks 3
1.2.2 Using throughput to deduce user location 4
1.2.3 Establishing a protocol for anonymous usage of a cellular network 4

1.3 Collaborators 5

2. RE-IDENTIFYING MOBILE DEVICES IN ADVERTISING NETWORKS **6**

2.1 Overview 6

2.1.1 Main results 8
2.1.2 Chapter outline 9

2.2 Background on mobile advertising 9

2.2.1 Information return 10
2.2.2 Device retargeting 11

2.3 Adversarial model 12

2.3.1 Self-service vantage point 13

2.4 Methodology 14

2.4.1 Marking implementation 14
2.4.2 Retargeting strategy 16

2.5 Evaluation 17

2.5.1 Data sets 18
2.5.2 Mark persistence 19
2.5.3 Retargeting 21

2.5.3.1 IP addresses 22
2.5.3.2 Geocoodinates 23
2.5.3.3 Additional filtering using device attributes 25

2.5.4 Cost 26

2.6 Case Studies 28

2.6.1 Case Study 1: Rediscovering Devices with One Impression 29
2.6.2 Case Study 2: Re-identification of VPN Users 30

2.6.2.1 Clearnet Identification using ad IDs 31
2.6.2.2 Clearnet Identification using Location 32

2.7 Discussion 33
2.8 Related work 35
2.9 Concluding Remarks 37

3. USING THROUGHPUT TO DEDUCE USER LOCATION 39

3.1 Overview 39

3.1.1 Main results 39
3.1.2 Chapter outline 40

3.2 Background on streaming 41
3.3 Attacker model 42
3.4 Approach 43
3.5 Data Exploration 44

3.5.1 Data Collection Methodology 45
3.5.2 Geographic Analysis 47

3.6 Classifiers for Mobile Throughput Traces 50

3.6.1 Model 51

3.6.2 HMM . . . 52
3.6.3 Sequence-based k-Nearest Neighbor Classifier . . . 54
3.6.4 Naive Bayes with a sequence of kernel density estimators . . . 54

3.7 Experimental Results . . . 56

3.7.1 Overview . . . 57
3.7.2 Differentiating Stationary and Mobile Users . . . 59
3.7.3 Determining a User's Path . . . 60
3.7.4 Effect of trace duration on accuracy . . . 62

3.8 Discussion . . . 63

3.8.1 Approaches to Enhancing Privacy . . . 63
3.8.2 Limitations . . . 64

3.9 Related work . . . 64
3.10 Concluding remarks . . . 67

4. ESTABLISHING A PROTOCOL FOR ANONYMOUS USAGE OF A CELLULAR NETWORK . . . 69

4.1 Overview . . . 69

4.1.1 Main results . . . 70
4.1.2 Chapter outline . . . 70

4.2 Background . . . 71
4.3 Attacker and Defender Algorithms . . . 75

4.3.1 Problem Statement . . . 75
4.3.2 Attacker Model . . . 76
4.3.3 Attacker-defender dynamics . . . 77

4.3.3.1 User strategy . . . 77
4.3.3.2 Attacker strategies . . . 78

4.4 Evaluation . . . 80

4.4.1 Datasets . . . 81
4.4.2 Behaviour that affects attacker accuracy . . . 81
4.4.3 Results . . . 84

4.4.3.1 Utility-privacy trade-off . . . 84
4.4.3.2 Trace length and location profiling . . . 85
4.4.3.3 Compromises in utility . . . 86

4.5 Integrating ZipPhone with emerging mobile networks 87

4.5.1 Background 88
4.5.2 Proposed ZipPhone Architecture 89

4.5.2.1 Overview 89
4.5.2.2 Purchasing Credentials 89
4.5.2.3 Communication without Leaking Identity or Location 90

4.5.3 ZipPhone Overhead 91

4.6 Discussion 94

4.6.1 Limitations 94
4.6.2 Ethical implications 95

4.7 Related Work 95
4.8 Concluding remarks 98

5. SUMMARY 99

5.1 Recommended policies 100
5.2 Recommendations for privacy-preserving behaviour 100

BIBLIOGRAPHY 102

LIST OF FIGURES

Figure **Page**

2.1 This ad was used for our data collection. 10

2.2 This is a simplified excerpt of the *dynamic* ad tag we used in experiments. It contains ad *macros* which are replaced by the DSP. It also contains a call to a JavaScript file on our server, which performs the marking functions. A static ad would only include the URL of the image *pixel.gif* file, conventionally an empty 1×1 pixel image. 11

2.3 Sequence diagram of a single real-time bidding cycle 12

2.4 Seven storage-based marking features that we evaluated. ✓$*$ indicates partial support. 15

2.5 Retargeting features available to advertisers, and the average proportion of devices matching a feature. 17

2.6 Each data set targeted a specific hypothesis. “Filter” indicates the type of parameter used for retargeting. 18

2.7 (Left) Persistence of different marking methods without renewal. (Right) Overall mark persistence with renewal. 20

2.8 Ability to rediscover using IP given different resetting policies. The dotted lines are cellular IPs, solid line is non-cellular. 22

2.9 Persistence of IP address from various ISP. 23

2.10 Extra impressions per day for an IP, with ad ID capping. Values shows the median. Many of our Windstream IPs exhibited no extra impressions. 24

2.11 Recall of GPS given different resetting policies. The left fact includes geolocation and the right includes precise GPS locations only. 25

2.12 Average number of impressions matching each static feature. 26

2.13 The impression rate of our ten target locations is correlated with the census-reported populations of the areas (r=0.93, p<0.0001) on a log-log scale. The upper bound of our sample had a rate of less than 1 impression per population per day. 27

2.14 Recall over one month of testing, against the daily cost of targeting a device. We tested users with and without using IP address for retargeting. We separately tested users on GPS apps. We used search radii of 0 (exact GPS match), 1, 8, 64, and 512 km; the costs of these monotonically increase with search radii. We also tested without a GPS filter (far right points that are most expensive). Shaded regions represent error. Daily costs were determined by multiplying impression rate by cost per impression ($4.02 CPM, or $0.00402 per impression). 28

2.15 The total spend of our campaign in each city. Overall, we successfully rediscovered 660 devices out of 1,727 in 48 hours, including 288 that never reported an ad ID. 29

2.16 CDF of time it takes to find user on clearnet. 32

2.17 The maximum radius of a geofenced re-target to determine the clearnet IP address of a device seen torrenting on a VPN. 33

3.1 For the 'metro' dataset, we gathered data on four popular paths in our area, in two directions for each path. All paths intersect in Amherst, MA, labeled as "X". For the 'campus' dataset, data was collected along the campus figure-eight bus loop around the University of Massachusetts campus, which runs in both directions. We compared both directions of paths along the West ("E"–"F") and East ("G"–"H"). 44

3.2 Details of the traces in our Measurement Sets. 45

3.3 On a per-second basis, the correlation between server-side throughput and client-side signal strength is 0.24. The plot shows a linear fit. Note: Android's reported signal strength correlates linearly to decibels, which is a logarithmic measure. 48

3.4 The mean throughput (left) and signal strength (right) of geographic areas in our measurements of paths to surrounding towns. On average, throughput in each cell is significantly different from throughput in 85.4% of other cells. This data suggests that latent information linking throughput and geography is available for training a classifier. 49

3.5 The mean throughput (left) and signal strength (right) of geographic areas in our measurements of our ‘campus’ data (a bus traveling within a 4 km^2 area). Throughput in each cell is significantly different from throughput in 73.6% of other cells. 49

3.6 Locations are represented by hidden states in a hidden Markov model, l_i. A user moves from one state to the next with a fixed transition probability of 0.7 multiplied by the probability an observed level of throughput is seen in that state according to the emission probability vector $\mathbf{e}_i$. 51

3.7 Classification accuracy for differentiating *stationary* vs. *mobile users*. Bolded entries have the highest accuracy; also bolded are entries that are not statistically different from the highest rate (two-sided, two-sample proportion test; 95% c.i.) The NB-KDE classifier performs flawlessly for the “metro” traces. 56

3.8 The empirical CDF of server-side roundtrip time (RTT) estimates for long-running TCP connections when a mobile device is static or moving. The same device is used for both cases. The static scenario demonstrates a noticeably different distribution of estimated RTTs compared to the mobile scenario; RTT is a primary factor in TCP throughput [88]. 58

3.9 Classification accuracy depending on which roads are included in the experiment. Bolded entries are not significantly less accurate than the most accurate result. Outward indicates “X”-to-…, and inward indicates the opposite. Traces were truncated to 10 minutes long. 58

3.10 The NB-KDE was able to differentiate between directions for most paths. The exception was “X-to-D” which was relatively symmetrical. Bolded entries are not significantly less accurate than the most accurate result. 60

3.11 We analyzed trace length vs accuracy to evaluate the efficacy of the attack if a connection session length is limited. The left graph indicates the *campus* experiment, and the right graph indicates the 8-class *metro* experiment. 62

3.12 Confusion matrices for the NB-KDE classifier in the corresponding *campus* and *metro* experiments for 600 seconds. 63

4.1 Left: Diverging paths that are regularly taken by two users. During training, an attacker would encode each labelled transition into a transition matrix for *location profiling*. Right: Separate, unlabelled activity where an unknown user reconnects using a new pseudonym at every tower. If the anonymous user does not successfully mix at these towers (i.e. does not remain offline long enough), the attacker can *link* the trajectories together and match the concatenated trace to User B's profile. 71

4.2 The movement of six users throughout a month, and on three weekdays within that month, produced from a log of tower transitions and the towers' estimated locations (each row represents a different user). Nodes and edges that are more opaque occurred with higher frequency. The axes are a consistent latitude and longitude, which have been redacted out of an abundance of caution to protect the privacy of the users. Each user displays somewhat routine (predictable) behaviour. However, the bottom three users (red, purple, and brown), who show similar activity, would benefit the most from ZipPhone. 72

4.3 User typology and their proportions in our target datasets, with a hypothesis about the amount of privacy a user could attain from ZipPhone. 81

4.4 **Left:** User predictability versus attacker accuracy, showing that attacker accuracy is near zero with low predictability. **Right:** User mixing versus attacker accuracy, showing that median accuracy falls to zero as user mixing increases. The plots were computed from the PhoneLab dataset. The presented results are for a maximum offline time period of 30 seconds and a set utility of 95%. 83

4.5 **Left**: PhoneLab. **Right**: RealityMining. In both datasets, predictable but mixing users (Type P/M) gain the most from using ZipPhone. Ten test traces were evaluated per user, and accuracy is represented as a mean of the proportion of successful reidentifications per user. Error bars represent a 95% confidence interval. 85

4.6 **Left**: PhoneLab. **Right**: RealityMining. Users lose a significant amount of privacy when traces are on the order of one day long. 86

4.7 **Left**: the effect of mixing-time on privacy while maintaining a 95% utility for the PhoneLab dataset. **Right**: privacy/utility of all users depending on whether their priority is privacy, phone calls, or screen use. Calls can be prioritized without sacrificing privacy. However, remaining online while the screen is on significantly reduces privacy. 87

4.8 Experimental setup for power measurements on 3G and 4G networks. 91

4.9 Power trace for 3G (left) and 4G (right) association and disassociation. 92

4.10 Time and power overhead incurred by a single association/disassociation procedure on 3G and 4G in our experiments. Results are averaged over 10 runs. 93

4.11 Battery usage does not exceed 1% per day, regardless of desired privacy or network type. 94

LIST OF ALGORITHMS

4.1 User identifier renewal strategy (*ZipPhone*) 77
4.2 Location profiling algorithm 78
4.3 Linking algorithm 80

CHAPTER 1

INTRODUCTION

You have zero privacy anyway.
Get over it.

Scott McNealy, *Wired*, 1999

Many consider "cell phone privacy" to be an oxymoron. Users depend on smartphones, but are usually forced into privacy policy agreements by monopolistic service providers: either trust us, or don't use our service. While a slim majority disapprove of surveillance actions by the government and technology corporations [94], many have accepted that a lack of privacy has simply become a way of life — a necessary cost for technological enrichment. But it has chilling effects on online behaviour: there are things that you might be afraid to Google.

When you use your phone, there are many opportunities for your privacy to be infringed upon. A mobile app may contain ads, which not only promote goods and services, but also send some data about your phone back to advertisers. The app itself may collect data about you and send information back to the developer — that is why phones require you to permit access to certain behaviours. The cell service provider themselves collect data about your movements and communications.

Location privacy in cell phones is particularly invasive and difficult to protect. Internet services and physical location are intertwined; every data packet is sent to a physical device that is somewhere on the planet, so any party who is responsible for forwarding it must have some information about where it is headed. At the same time, location data is the one aspect of privacy that has implications on a user's physical security. We are expected to trust corporate entities and governments with this data, but their trustworthiness is questionable. In July 2019, the Electronic Frontier Foundation filed a class-action lawsuit against AT&T for selling users' real-time locations to bounty hunters [5].

My thesis shines a light on this tension between privacy and utility of mobile Internet services. I ask how much privacy you must *necessarily* sacrifice to continue using a service. I also look at methods to protect your location information. I consider each of the parties involved in an app service — an advertiser, a developer, and a cell service provider — and study scenarios in which user privacy is at risk. I show that protecting privacy does not require a complete sacrifice of connectivity, but does demand some changes in policy and implementation, and conservative behaviour.

1.1 Approach to studying privacy and utility

Private information is data about an individual that she does not want someone else to have. Ideally, users choose to trust an entity with personal information, and any information revealed to the entity is passed **with consent**; alas, this is often not the case. To measure privacy, we look specifically at what the individual (or *user*) wants to hide, and the party whom wants to arrogate it (the *attacker*). Without an attacker, there is no privacy problem. The specific method that an attacker uses to gain this information against the user's will is an *attack*.

To investigate an attack, I define attacker models that formalize assumptions about user behaviour and attacker ability. In each of my studies, the attacker has access to time-stamped series of unlabelled information, or *traces*. This information is logged as a result of user connectivity — this may be web access records or phone *call detail logs*. These traces contain information in some form that can be used to link a user to their location. We measure privacy in terms of the probability that this information can accurately establish this link.

In these attacks, user identity or location information is inferred through the use of various classifiers. The location information itself is potentially revealed through several channels. I discuss mobile ads in Chapter 1, which can reveal information both at the application or network layers [37], through GPS and IP address respectively. In Chapter 2, I discuss how throughput measurements are affected by physical layer factors such as radio signal occlusions and cell tower distance. In Chapter 3, I discuss a method to remain anonymous against a cellular network provider, who oversees cellular signalling and controls both the physical and network layers. We quantify the performance of these inference attacks using datasets that we have collected or acquired from other institutions.

With this knowledge, we recommend solutions to fix these issues. This results in a sacrifice of some amount of utility — a reduction in performance or quality of service.

1.2 Overview of studies

In each of the three studies, I fundamentally ask (1) how connectivity and signalling give away your location or other private information about you, and (2) what you need to sacrifice to regain privacy.

1.2.1 Re-identifying mobile devices in advertising networks

Users are tracked through ads on the Internet. On the web, users are exposed to tracking cookies. On Android and iOS apps, users are tracked using an advertising identifier (ad ID). Because ad IDs put users' privacy at risk, both OSes allow the ID to be reset, or disabled completely by turning on Limit Ad Tracking (LAT). In apps, ads are shown in *web views* that support storage and caching mechanisms. These functions are a vulnerability that allows advertisers to mark devices whether LAT is enabled or not; our marking methods worked on more than all of the thousands of devices we encountered. Becoming an online advertiser has a very low barrier of entry; anyone with a few dollars and an Internet connection can perform this attack. In experiments where we deployed hundreds of thousands of ads, we quantified the cost of marking and retargeting without an ad ID; of the successfully marked devices, 49% of iOS impressions, and 59% of Android, were successfully labelled with a budget of $5 per day. We additionally verified this method on 1,727 devices and recovered 660 of them within 48 hours for $86.73. We recommend that app developers disable storage and caching in ad web views since it would result in a negligible performance cost.

1.2.2 Using throughput to deduce user location

Smartphones contain a GPS sensor that can finely locate users in real time. Because of the sensitive nature of location data, all smartphones give users precise control over app and OS access to the sensor. A user's location can be deduced by a second party simply through changes in connection quality; for example, I can tell when you are going through a tunnel when your signal drops briefly during a phone call. We leveraged this idea in an automated attack that can be performed by any web or app service. Given a sequence trace of per-second network throughput measurements, how well can an attacker infer the path of a user? We collected hundreds of throughput traces by streaming music to phones to simulate two scenarios: a campus user travelling to or from four different towns, and a user travelling within our campus. We evaluated three classifiers: k-nearest neighbours (k-NN), a hidden Markov model (HMM), and a naïve Bayes classifier with kernel density-based estimates (NB-KDE). In our experiments, NB-KDE performed best, identifying the path and direction among two roads within a University campus (four classes) with 77% accuracy, and the path and direction among four roads around campus (eight classes) with 83% accuracy. The only mitigation to this attack is traffic shaping — artificially randomly slowing the connection to confuse the classifier.

1.2.3 Establishing a protocol for anonymous usage of a cellular network

Because connectivity cannot be hidden by nature (the untrusted entity always knows when it is party to a connection), users seek anonymity instead. In cellular networks, this anonymity currently comes in the form of a prepaid, disposable phone and SIM card. But these phones are easily compromised, especially if the anonymous user exhibits habitual behaviour. Attackers can build a *location profile* about a user and guess which prepaid phone is theirs by matching the profile to unlabelled sequences of cell tower connections. Additionally, attackers can *link trajectories* of unlabelled sequences to better inform their inferences. We show that buying a new phone intermittently is insufficient to preserve privacy. In fact, if a user shows regularity in her behaviour, one month is enough for an attacker to identify with 69% accuracy among over a hundred users. Even one day is enough to reidentify a device with greater than 50% accuracy. We introduce a protocol called ZipPhone, which allows a community of privacy-seeking users to strategically time their connections to remain anonymous while incurring a minimal loss of utility. We evaluate ZipPhone from the perspectives of both the cell service provider and the user community, and quantify the privacy/utility trade-off using two datasets containing cell tower logs of hundreds of users. We present and assess a deanonymization algorithm that combines both location profiling and trajectory linking attacks, and show that by sacrificing 30 seconds of uptime every ten minutes, they can thwart the attacks and reduce identifiability from 69% to 24%.

1.3 Collaborators

All three studies were conducted under the supervision of Brian Levine. The study featured in Chapter 2 was a collaboration with Mark Corner, Brian Levine, and JianYi Huang. Chapter 3 was done in collaboration with Joydeep Biswas, Erik Learned-Miller, Brian Levine, Marc Liberatore, and Hamed Soroush. Chapter 4 was done in collaboration with Brian Levine, Marc Liberatore, and Mariya Zheleva.

CHAPTER 2

RE-IDENTIFYING MOBILE DEVICES IN ADVERTISING NETWORKS

2.1 Overview

Advertisers are often reviled, but they are an invaluable part of the mobile ecosystem. Not only do they fund many games and services that are free to the user, they also save the user from giving out their credit card number and personal information to vendors they may not trust.

On the other hand, by using mobile apps supported by advertisements, users allow fine-grained information to be shared with hundreds of entities in the advertising ecosystem. Advertising libraries embedded in apps facilitate the dissemination of the device's unique *mobile advertising identifier* (ad ID), IP address, user-agent string, application name, and fine-grained GPS coordinates to entities that bid for ad space via open *Real-Time Bidding* (RTB) marketplaces [87]. Advertisers collect this information on the showing of an ad, and can use the ad ID or other information to link together subsequent encounters of a device. These logs are used to build detailed profiles about a user device, including information about preferences, habits, and locations frequented.

Users should aim to protect their privacy by *(i)* preventing revealing information such as location or IP address from being associated to these encounters, and *(ii)* preventing the logged encounters themselves from being linked together, so that if an encounter is inferred, the damage is minimized. As we show in this chapter, even when users take steps to hide or disable application access to information that could identify them, advertisers can still *re-identify* devices in RTB networks. The lack of controls in advertising's access to web APIs for storage allow advertisers to leave a retrievable *mark* on the user's device, retarget groups of devices that may contain the user, and then verify the mark when an ad is shown. This re-identification of users can happen even when the user disables or changes the ad ID, uses a VPN, or disallows location.

In work submitted to MobiCom (2020) in collaboration with Brian Levine, Mark Corner, and JianYi Huang, we quantify the effectiveness of the tools available to RTB advertisers to re-identify mobile devices carried by users, including cases where the user has taken steps to prevent it. We determine effectiveness in terms of the cost in dollars based on real campaigns we executed. To fully control exposure of identifying information, a user must cobble together several privacy-preserving techniques, and is vulnerable to several pitfalls:

- First, a user can be uniquely re-identified by the ad ID. To control this technique, the user must turn on privacy features found in iOS and Android. Enabling iOS's *Limit Ad Tracking* (LAT) setting zeros out the identifier, making the device indistinguishable from other users of that feature [11]. On Android, enabling the *opt out of ad personalization* (OOAP) feature is merely an advisory flag that apps and advertisers are expected by Google to respect [1]. It's a limited tool for users, and continuously resetting the ad ID via a well-buried button is simply impractical.

- Second, advertisers can make use of various methods of *marking* users (LocalStorage, IndexedDB, cookies, etc.). The mark can be used by advertiser to link observations of the same device despite a reset or zeroed-out ad ID. There are no user-facing methods to control this and to a large extent Android and iOS do not allow applications to prevent advertisers from using those APIs.

- Third, advertisers can leverage features to re-target devices, including device model, app, and the IP address. Further, some IP addresses can be coarsely geolocated to approximate locations, and many advertising services allow ads to be targeted within geographic fences. Additionally, advertisers can make use of precise GPS coordinates, if the user consents to it, and many do. Users can only control this if they flawlessly use a VPN all of the time and disallow location access in the app.

We show that these three vulnerabilities are difficult to avoid, and allow an advertiser to link encounters together cheaply, and in many cases associate them with a location.

2.1.1 Main results

Our main goal of this study was to determine the cost and efficacy of linking together ad records that do not contain the ad ID. We additionally looked at whether users who were trying to protect their privacy by using a VPN could still be linked to a clearnet IP address (i.e. one that is assigned by the ISP, and often corresponding to a geographic location) because their ad ID was not disabled.

Our data was collected by deploying hundreds of thousands of ads (i.e. impressions) on a self-service RTB platform. Our focus was an advertiser that seeks to maximize the recall of impressions attributable to a specific mobile device, even when the user is trying to prevent re-identification. Our goal was not to optimize cost, but we favored solutions that are less expensive.

We first performed a *post hoc* analysis on impressions gathered by targeting the ad ID of 1,384 devices for one month. For about $5/day, about half of a user's impressions could be retrieved through marks and profile-based retargeting, compared with ad ID retargeting. We then performed an end-to-end experiment, attempting to re-discover 1,727 devices without using ad ID. We further recovered 38% of devices within 48 hours for $86.73. Finally, we deployed ads to VPN users on BitTorrent apps, and found that about two-thirds were seen on the clearnet within 26 days. Further, 68% of these devices reported a precise GPS coordinate at least once.

2.1.2 Chapter outline

Section 2.2 describes the app-based techniques we used for re-identification. To our knowledge, we were the first to discover and report that the in-app web views that display ads can be used to store marks. There are no methods that let the user disable storage or clear the marks, other than deleting and reinstalling the app. Our adversarial model is outlined in Section 2.3. Specific marking methods are then detailed in Section 2.4.

In Section 2.5, our main results and cost analyses are presented. We found that without the ad ID entirely, about 75% of Android devices can be re-identified in the first week using marks, and 81% of iOS devices. Even 30 days later, 61% of our Android impressions and 71% of iOS contained marks to re-identify the device. We then show that it is possible to re-identify devices in campaigns even without the ad ID. Devices were found again on previously visited non-cellular IPs 78% of the time on average. On average, devices seen in one week were found within 10 km in 80% of instances the next week. Devices with iOS are significantly more difficult to re-identify because the different Apple device models are not distinguished. We then perform a cost analysis to determine the cost to an advertiser to use this method.

Lastly, in Section 2.6, we perform two case studies. First, we determined the cost and performance of attempting to rediscover over a thousand devices using information from only one impression, and without using the ad ID at all. Second, we show that ads can very effectively reveal the clearnet IPs (and locations) of users of BitTorrent apps masked by a VPN service.

2.2 Background on mobile advertising

There are many modalities of purchasing advertisements, from walled-garden advertising systems such as Facebook, to relatively open *real-time bidding* (RTB) networks. In RTB, apps that want to sell ad space (i.e., *publishers*) work with *Supply Side Platforms* (SSPs), and entities that wish to show their ads (i.e., advertisers) work with *Demand Side*

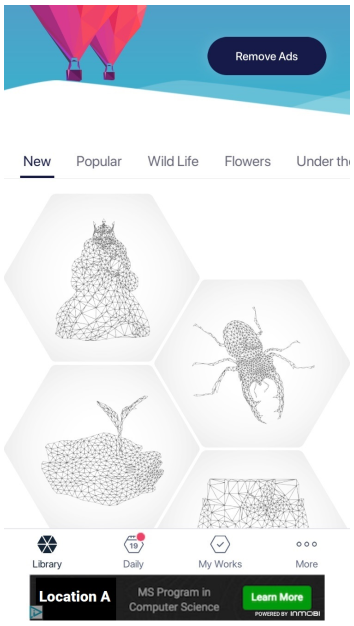


Figure 2.1: This ad was used for our data collection.

Platforms (DSPs). When an app shows an advertisement it contacts one or more SSPs, The SSP brokers an auction amongst a set of DSPs who bid on behalf of the advertisers they represent. The highest bidder wins, and the winning ad (known as a *creative*) is sent to the user's device. When the app shows the creative, it is called an *impression*. This auction happens in fractions of a second and is summarized in Figure 2.3. Further details of the RTB ecosystem can be found in works by Vines et al. [108] and Corner et al. [32]. Small ads are very inexpensive, e.g., typically \$0.10–\$5.00 per 1000 shown (*per mille*, in industry parlance).

2.2.1 Information return

Once a bid has been won, the device is sent a snippet of HTML code, commonly called an *ad tag*, and the app displays it within an embedded web view. This snippet may contain HTML elements that are requested directly from the advertiser. A sample of our deployed ad tag is shown in Figure 2.2. This HTML code contains the advertiser's creative image and tracking mechanics, including *pixels* or *beacons* that allow the advertiser to verify that the

```
<!-- This ad is part of a research project at Location A. ...
   //-->
<a href="https://www.example.tld">
<img id="splash_pixel" style="display:none"
src="https://api.example.tld/pixel.gif?impression=${impression_id
   }&device_identifier=${device_identifier}&${imp_id}&gps=${gps}&
   campaign=${campaign_id}..." /></a>
<script src="https://api.example.tld/mark.js">
</script>
```

Figure 2.2: This is a simplified excerpt of the *dynamic* ad tag we used in experiments. It contains ad *macros* which are replaced by the DSP. It also contains a call to a JavaScript file on our server, which performs the marking functions. A static ad would only include the URL of the image *pixel.gif* file, conventionally an empty 1×1 pixel image.

impression was successful. The DSP substitutes the ad *macros* with information obtained from either the device, or third-party data providers. This information includes an ad ID (when available), the time, the service provider, location (when available), the app, and some device information. Ad IDs are 16-byte UUIDs and are called the *ID for Advertisers* (IFA) in iOS and the *Google Ad ID* in Android. The location information is provided by the device GPS when it is available or IP-based geolocation service.

By including and serving an image in the tag, the advertiser can collect the device's IP address. Device information can be inputted as query string parameters in the image's URL, and the advertiser keeps a log of resource requests that would include the URL and the connecting IP address. If the advertiser returns JavaScript as part of the ad, then it can access browser APIs, such as storage in IndexedDB and LocalStorage. None of the above requires clicking the ad.

2.2.2 Device retargeting

Advertisers on RTB systems can target and retarget devices based on features such as location, IP address, ISP, app, device, and time of day. Advertisers often use ad IDs to retarget the same device over time. ad IDs are unique to the device and shared across all apps on the device (and not available from ads viewed with a normal web browser).

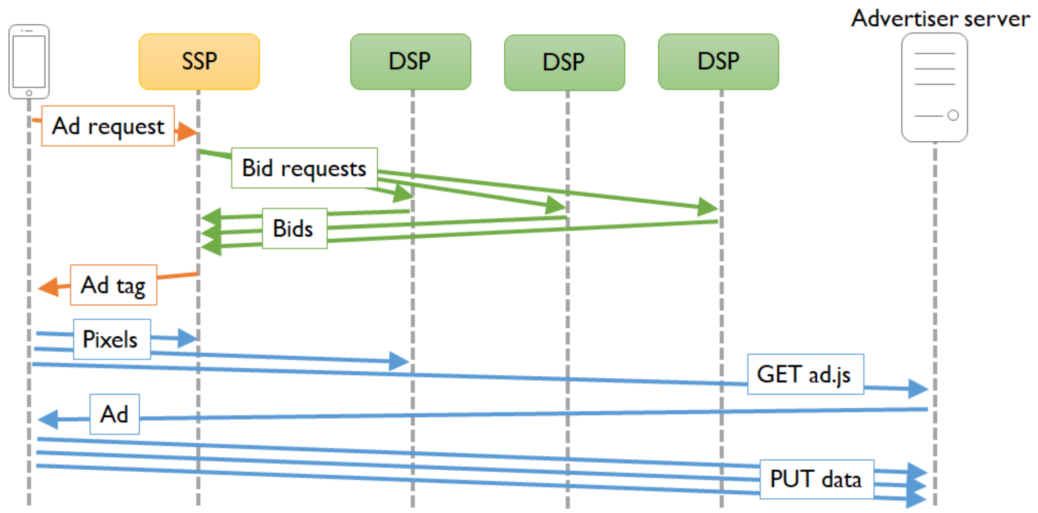


Figure 2.3: Sequence diagram of a single real-time bidding cycle

These device-level identifiers are very powerful as they are consistent across all apps on the device. Users have some control over their privacy: on both Android and iOS, users can reset the ID at any time, and on iOS the ad ID can be disabled outright.

Devices can also be retargeted by the coarse geolocation or their precise GPS location. Weather and dating apps often share precise GPS [33], and surprisingly so do BitTorrent apps. Home and work IP addresses typically match a small number of devices and are often very persistent [93]. But even in the case of shared IP addresses, such as those at a library or coffee shop, the geolocation of the IP address massively narrows down the number of possible users while retargeting devices. Cellular IPs are much broader as geolocating such IPs could be several states wide in the USA [21]. We discuss location in more detail in Section 2.4.2.

2.3 Adversarial model

Our adversary uses RTB to place advertising impressions on mobile devices with the following goal:

Goal: *to maximize the recall of impressions on a particular mobile device even when the device's user tries to prevent re-identification (such as resetting or clearing the ad ID, denying the application access to location, or using a VPN).*

Our goal is not to optimize cost, but we favor solutions that are less expensive.

The user may reset their ad ID periodically or disable it (setting it to all zeros). The user may make use of many IP addresses as a natural consequence of movement, or because of their use of a cellular NAT, or a VPN. The user may explicitly allow access to GPS locations, but cannot prevent the advertiser from using a geo-location service based on their IP address.

We evaluate an adversary's success rate and cost of re-identifying a device over the course of a month. During that time, the user can either disable her ad ID, reset it daily, or reset it weekly. The adversary does not aim to discover the true identity of the user; rather, the adversary tries to associate anonymous behaviour with previous activity.

The most obvious defense against the advertiser (excluding buying the ad-free version of an app) is to use a VPN, which can mask the user's *clearnet* IP address (i.e., an address assigned by an ISP that maintains DHCP records and accurate billing information). Thus, in Section 2.6.2, we explore an alternate goal: to determine the clearnet IP address (and therefore geographic locale) of a device masked by a VPN.

2.3.1 Self-service vantage point

In RTB systems, there are multiple vantage points to observe users: as the developer of the advertising SDK in the app, the SSP, the DSP, and as a ”self-service” user of a DSP. The developer of the advertising SDK can trivially track users as they have full native access to the device. The SSP and DSP see all bid requests coming from the device and can track users at no additional economic cost, though they can only access data available in the bid request. We have chosen to analyze the *weakest* vantage point: that of the self-service user. One can sign up with a self-service DSP easily; minimum spend is typically $100USD.

Becoming a full-fledged DSP is a major undertaking with additional controls applied by the SSPs. A self-service adversary can target ads to users, but only gets feedback when an impression is won. Therefore, collecting data has an economic cost, typically 10 cents to \$5 for each 1000 ads (called a CPM).

Several factors affect the ability of the self-service advertiser to retarget a device. First, there is only an opportunity to place an impression when the user opens the app. The advertiser must have funds to pay for the opportunity and win the auction. The adversary has greater success when features of the device can be used to increase precision. Some DSPs offer location-based targeting, and support for targeting specific device models and apps is common. The best case scenario for the advertiser is targeting of a specific ad ID.

2.4 Methodology

To evaluate the ability of advertisers to meet the adversarial goals we define in Section 2.3, we implemented a strategy for an adversary to re-identify users in RTB networks. Starting from an initial ad impression, the adversary uniquely *marks* the device, then later retargets advertisements to a profile that includes the original device, and finally confirms which device was the original one by reading the mark. Here, we detail our implementation for marking devices and then retargeting them to rediscover particular devices.

2.4.1 Marking implementation

We have discovered that the same APIs available to browser-based pages are available unheeded to advertisements. As in-app advertisements are shown within embedded web views, this allows ads to use any browser-based method of storing unique identifiers in the device and using it as an identification mark.

We identified and implemented seven different methods of marking devices, listed in Figure 2.4. In our implementation, we had access to JavaScript-based ads, giving us access to browser APIs for the first four features: LocalStorage, IndexedDB, Web SQL, and the

Feature	JS	Non-JS	iOS	Android
1. LocalStorage	✓		✓	
2. IndexedDB	✓		✓	✓
3. Web SQL	✓		✓∗	✓
4. Cache API	✓		✓∗	✓∗
5. Unique, Cached JS file	✓		✓	✓
6. Cookies	✓	✓	✓∗	✓
7. HTTP ETag	✓	✓	✓	✓

Figure 2.4: Seven storage-based marking features that we evaluated. ✓∗ indicates partial support.

cache. As a fifth feature, our implementation served unique JavaScript files to each user that contained the mark; that mark is accessible when the cached JavaScript is executed in subsequent ads.

In some cases, an advertiser may not have access to serving JavaScript as part of the advertisement. In those cases the advertiser typically uses 1x1 tracking pixels, which the advertisement fetches from the advertiser's servers. To support these cases, we implemented our sixth and seventh marking methods, respectively: set and retrieve a cookie when responding to the tracking pixel; and HTTP ETags, which browsers transmit to the server to re-validate an expired cache item. The ETag is set in one ad, and then sent to the server by the phone in a subsequent ad.

If either the user or developer of the ad SDK can prevent or erase such marks, then user privacy would be increased: the adversary cannot know that they found the exact device again, only that it is one of the many devices in the re-targeted group.

For a user to prevent marking, they only have three coarse options: *(i)* they can clear stored information by deleting all of the app's data and reinstalling; *(ii)* they can buy a version of the app that has no advertisements, if available; or *(iii)* not use the app at all.

A better option is for developers of ad SDKs to configure the web view to enable or disable the features in Figure 2.4 (as we show in Section 2.5, at least one storage method is typically available in ad SDK web views.) Completely disabling web view storage is non-trivial for developers on both iOS and Android. The programming calls to disable are not well documented; we discuss a method we discovered through testing in Section 2.7.

A disadvantage of re-indentifying devices with web view-based marks is that each app is sandboxed. Tags stored on one of these in-app web views cannot be seen by other apps. This means that all re-identification of devices must take place within a single app.

2.4.2 Retargeting strategy

The advertiser has to balance re-identifying the user with the economic cost. A typical DSP gives access to billions of advertising opportunities per day. When the ad ID is not available, the advertiser must avoid advertising to such a large crowd that it becomes economically infeasible.

Figure 2.5 lists the retargeting features available to us from the DSP we used. We filter impressions so that a minimum number of impressions is required to re-identify a single user. For example, IP addresses in homes typically do not change over time; and it is easy to retarget a device from that home by the address. Such a strategy will, however, miss opportunities to retarget the user's device at their workplace. Targeting a radius of geographic locations around a first sighting of a device captures more possibilities than a home IP alone, although at an increased cost. Costs can be reduced by refining the advertising targeting by including other factors, such as the device operating system, the bundle identifier for the application they use, and the device model.

Somewhat ironically, when users enable Limit Ad-Tracking on their devices, it can make them more identifiable as only one out of six users enable that option. The particular DSP we used did not allow us to target an empty string or zero ID. If other DSPs allow that, one could use the zero identifier as a way to filter users.

Feature	**Prop. matched**
Device OS	0.51
Bundle	0.24
OS Version	0.18
Android Model	0.012
GPS	1.1×10^{-4}
IP	6.3×10^{-5}

Figure 2.5: Retargeting features available to advertisers, and the average proportion of devices matching a feature. Further discussion is in Section 2.5.3.

2.5 Evaluation

Our evaluation is focused on quantifying the advertiser's success rate and economic cost of re-identification. We compare storage-based marks and feature-based retargeting against persistent ad IDs. We conducted the evaluation through a self-service DSP using banner advertisements purchased in a wide variety of mobile applications. We collected several data sets, comprising hundreds of thousands of ad impressions using a variety of targeting techniques. We begin with a description of the data we collected, and our evaluation comprises three results:

- First, we determined the *persistence* of each marking method; we looked at how long a mark would remain on a phone without renewing it.
- Second, we determined how often users could be rediscovered over one month, compared to ad ID targeting. We also performed *post hoc* analyses to evaluate the relationship between cost and rediscovery success.

We also conducted two case studies, in Section 2.6.2. One of these is an end-to-end measure of cost and efficacy in attempting to rediscover 1,727 users without using ad ID, from only one impression. Another is an evaluation of the behaviour and privacy of VPN users.

Data set	Days	Imps	ad IDs	Filter	Section
PERSIST	81	52,719	1,715	ad ID	§2.5.2
LONG	33	108,084	1,384	ad ID	§2.5.2
IP	3	72,508	44,055	IP	§2.5.3
GPS	3	44,570	28,673	GPS	§2.5.3
FIRST	4	22,319	9,986	IP/GPS	§2.6.1
VPN	28	74,026	13,778	ad ID, App	§2.6.2

Figure 2.6: Each data set targeted a specific hypothesis. "Filter" indicates the type of parameter used for retargeting.

2.5.1 Data sets

We created an account on a self-service DSP and purchased JavaScript-based banner advertisements in hundreds of Android and iOS applications. These included games, weather applications, chat, dating, and others. Embedded in the advertisement's HTML code were seven different marking methods, as well methods that requested the DSP return the device's IP address, geolocation, user-agent string, OS version, and any available ad ID. While the end goal of our advertising adversary is to re-identify devices even if users rotate or block their ad ID, we use the ad ID as the ground truth for our experiments. This ground truth allowed us to know with certainty that we re-identifed the user or if we failed to re-identify them via retargeting. Our bids were a maximum of $10 CPM; the average winning bid was $4.02025 per thousand ads. We bid extremely high to increase our data set.

The data sets we collected are summarized in Figure 2.6. We targeted ads to eight small US towns and two larger US cities over a period of two days. From these two days, we randomly sampled a set of 3,099 ad IDs and split the set into two data sets.

- We placed 1,715 devices into our PERSIST data set. We retargeted each of these using their ad ID after 7, 20, 70, and 81 days from the original impression, with no impression in between. The purpose of this data set was to verify the persistence of different marking methods.

- We placed the remaining 1,384 devices in our *LONG* data set. These were retargeted every two hours over a period of one month. We targeted devices that did not disable the ad ID, and in addition to storing and reading marks, we collected geo-coordinates, IP address, device information, and app. We used these features to compare the performance of our strategy to ad ID retargeting.

The overhead cost of this rediscovery strategy is proportional to the number of impressions allowed by the retargeting filter. We deployed several campaigns to measure the impression rates of various filters. We used these measurements in conjunction with the `LONG` data set to analyze *post hoc* the cost and efficacy of several targeting scenarios.

- Our `IP` data set consists of impressions collected from targeting a sample of 920 IP addresses from devices in the `LONG` data set. Addresses were collected from the first five days (i.e. the training period) and were used to quantify the cost of retargeting by IP.

- Our `GPS` data set consists of impressions from retargeting a 10 km radius within 8 towns and 2 cities. From this data, we determined the correlation between impression rate and population density.

We then conduct two case studies.

- In our `FIRST` data set, we attempted to rediscover 1,727 devices from five US towns using information from only the first impression, and without using ad ID at all. While 1,041 devices of these devices reported ad ID, which we ignored, 686 did not report one at all.

- Finally, we have separate data set for our `VPN` case study, discussed in Section 2.6.2.

2.5.2 Mark persistence

The persistence of marks varies between storage mechanisms, and the adversary's goal is to re-mark a device before the original mark is cleared. For example, the mark may

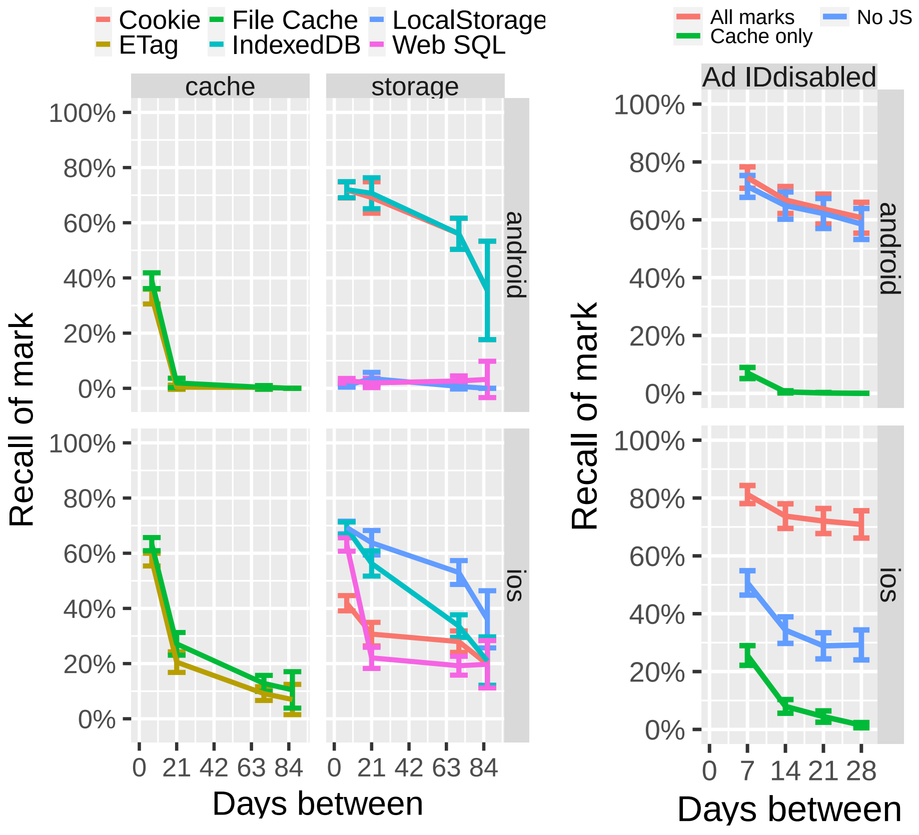


Figure 2.7: (Left) Persistence of different marking methods without renewal. (Right) Overall mark persistence with renewal.

be cleared when the app has exceeded its assigned quota [9], or will be cleared when an app or OS is reinstalled or updated.

To measure how long marks persist, we used our `PERSIST` data set. We set marks using the seven techniques (described in Section 2.4.1) and intermittently retargeted the same set of ad IDs to check if the marks disappeared. Figure 2.7 (left) shows these results. While caching marks were mostly cleared after one week, storage methods lasted much longer. On iOS, LocalStorage was most usable, seen again in 69.3% of impressions after a week. On Android, LocalStorage is disabled by default, but IndexedDB and cookies were both available on most apps, seen 72.0% of the time. More than half of the most persistent marks were still retrievable after two months.

Unless all stored marks are cleared at the same time, the missing values can be restored using the existing identifiers, similar to the *evercookie* [69]. Using the `LONG` data set, we quantified the persistence of overall marking if users were retargeted every two hours for a month. These results are summarized in Figure 2.7 (right). Marks on Android devices had about the same persistence as IndexedDB or cookies alone, indicating that when marks are cleared, they are all cleared at once. On the other hand, iOS impressions were recalled at a significantly higher rate, remaining at 75% recall after one month, indicating that different marks were cleared at different times.

2.5.3 Retargeting

The cost to rediscover a device depends on the retargeting filter; more permissive filters are more likely to find a particular device again, but also increase the number of non-target devices and consequently the overall cost. IP targeting can be very effective and specific if a device connects using a consistent IP address. Geocoordinate targeting is also especially useful since it only targets devices within a fixed location, and users generally stay in the same area.

We investigated a strategy of targeting the IP addresses seen during training, additionally targeting some radius of the GPS coordinates seen, and filtering out irrelevant bids based on static features (device model, OS, and app used). The success of this strategy depends on the devices' impressions matching consistent geographic coordinates and IP addresses over time, as well as consistent use of apps.

Our basis for evaluating cost was impression rate (i.e. number of extraneous impressions per day) measured over three days of testing. In both `GPS` and `IP` data sets, we attempted to limit impressions to one per device using a flag provided by the DSP. This impression limiting does not work perfectly; devices that do not share an ad ID could be counted more than once. We expected the impression rate to go down, but running a longitudinal experiment to quantify this decrease was not financially viable.

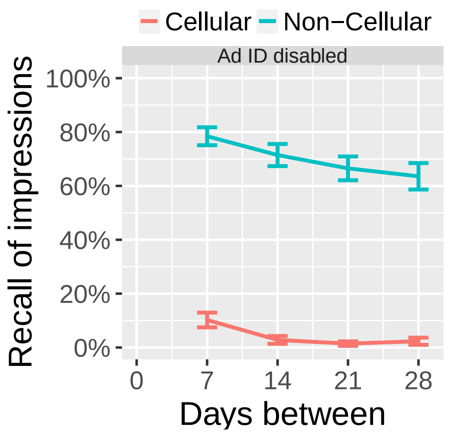


Figure 2.8: Ability to rediscover using IP given different resetting policies. The dotted lines are cellular IPs, solid line is non-cellular.

2.5.3.1 IP addresses

We first evaluated the persistence and specificity of IP addresses used by mobile devices. We expected that many non-cellular IP addresses would lead to a higher rate of device rediscovery, and incur a low overhead of unwanted impressions, and we expected the opposite to be true of cellular IP addresses. The latter tends to assign temporary IPs from a pool of addresses with many users sharing a particular IP.

We considered the first five days of the `LONG` data set to be the training period, and evaluated the overall usage of those IPs by each device during the next four weeks. Using MaxMind [13], we were able to determine the ISP of each IP, and label them as cellular or non-cellular. In Figure 2.8, we show that non-cellular IP is a very effective retargeting feature: in many cases, users return to *sticky IPs* [93], addresses that are assigned to a modem long term. These IPs have a much higher recall than cellular IPs which are typically ephemeral. Figure 2.9 shows a breakdown of the same IP recall by ISP, which further highlights the amount of privacy gained simply by using a cellular ISP. Non-cellular IPs from some ISPs are found 80–90% of the time over the entire month, while cellular IPs are rarely useful with less than 10% recall.

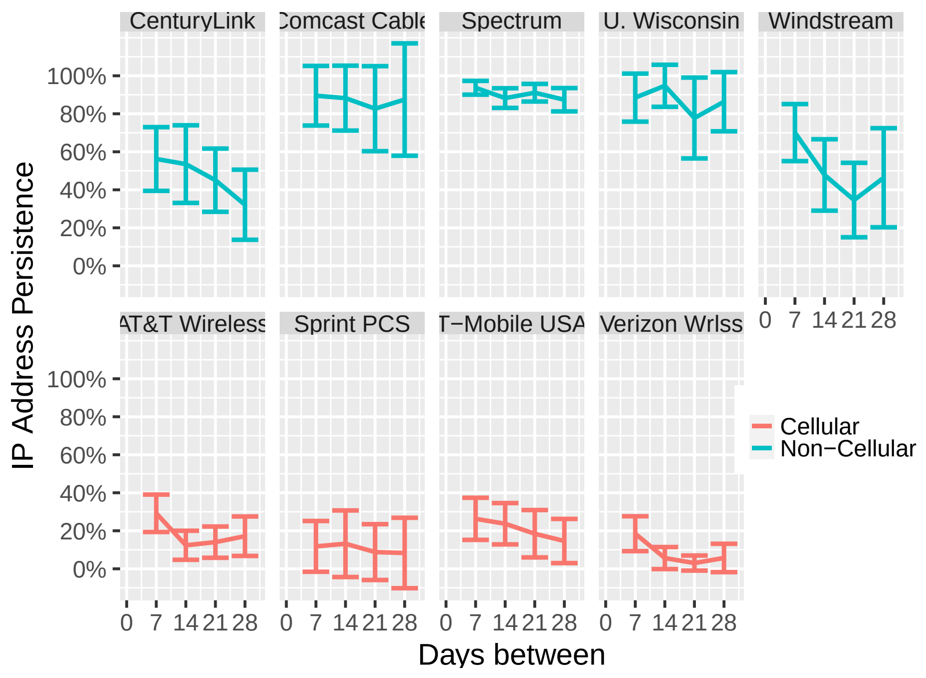


Figure 2.9: Persistence of IP address from various ISP.

However, we must also consider how many other devices may be using the same IP address to determine how expensive it will be to retarget a particular device. Figure 2.10 shows the impression rate of different ISPs, as determined from our `IP` data. Cellular IPs have an impression rate in the thousands (save for Verizon with hundreds), while non-cellular IPs, being more specific to individual users, were in the tens. In sum, devices on non-cellular ISPs can be targeted cheaply and effectively.

2.5.3.2 Geocoodinates

IP targeting only works well if the device connects through a consistent address, however, *geofencing* can boost recall if the device uses another IP but remains in the same general area. The downside of geographic targets are that they are less specific, thus more expensive. The reported geocoordinates are provided by the DSP, but its original source could either be from the device (GPS or network triangulation) or inferred from IP using a service like IP2Location [12] or MaxMind [13]. We were able to identify the more ac-

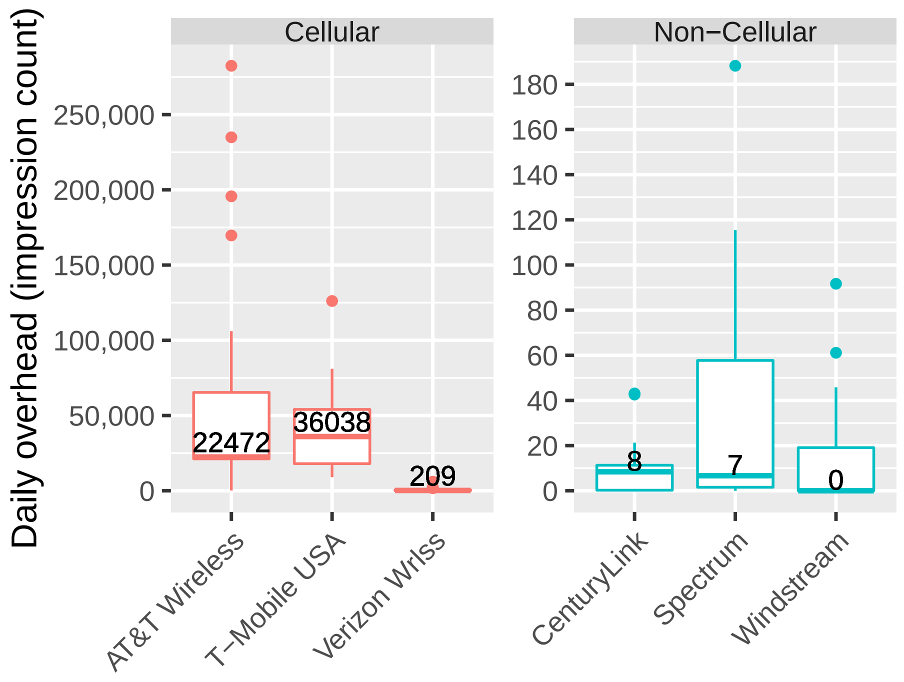


Figure 2.10: Extra impressions per day for an IP, with ad ID capping. Values shows the median. Many of our Windstream IPs exhibited no extra impressions.

curate device-sourced locations from the number of digits of precision [33]—coordinates truncated to the thousandth digit were likely inferred from IP.

Using coordinates for the training period of the first five days, we analyzed the recall over time given different search radii. Figure 2.11 shows these results. With a 0 km radius (i.e. exact match), recall was 42% in the first week, falling to 27%. This is not because users are returning to the exact same location many times; rather, they are connecting to the same IP, which geolocated to the same coordinates. Conversely, looking only at the 0 km recall for precise geocoordinate retargeting, we can see that devices are not often in the exact same position. Using a radius of 1 km dramatically increases recall to 67% for all GPS impressions, and 82% for precise GPS impressions. This confirms that users and their devices usually return to the same place; compared to locations reported during the first five training days, devices regularly continued to appear in the following month within 1 km.

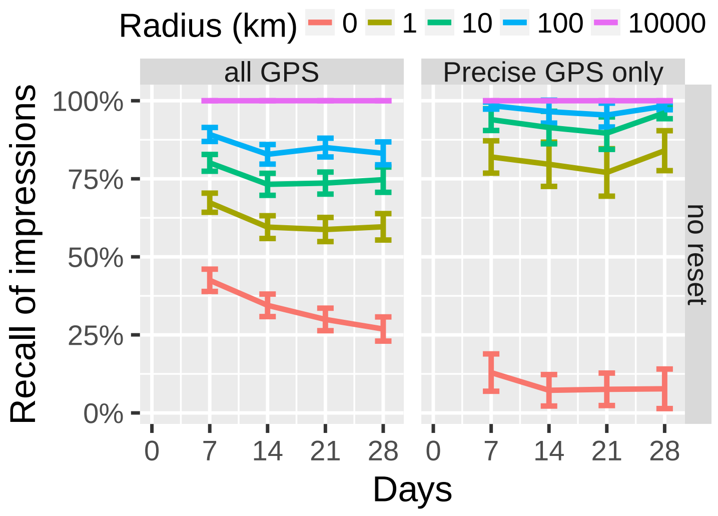


Figure 2.11: Recall of GPS given different resetting policies. The left fact includes geolocation and the right includes precise GPS locations only.

To estimate the overhead, we used our GPS data set, which targeted areas with a 10 km radius in ten population centers in the US. We compared the resulting impression rates with measurements from the Gridded Population of the World, v4.11 data set [4], which contains population estimates within 1×1 km cells. Figure 2.13 shows these measurements; we verified that impression rate was correlated with known population (r=0.93, p¡0.0001).

2.5.3.3 Additional filtering using device attributes

Finally, our filter excludes all irrelevant ad requests based on device model and app used, to further reduce the cost. These device attributes are unchanging, and we found that most users are found on 2 apps, with 89.2% on their favorite app more than half the time. We did not evaluate using OS version, which may be discriminating, but changes over time.

Figure 2.12 shows the proportion of matching devices of each feature. Because there are a wide variety of Android devices, model is a high entropy feature for that OS. However, all iPhones self-report as “iPhone”, making the iOS filter nine times less effective.

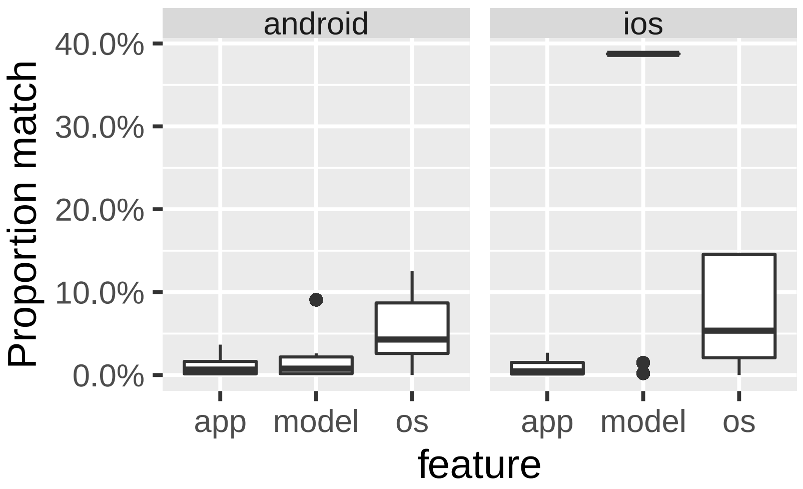


Figure 2.12: Average number of impressions matching each static feature.

Overall, the impression rates on Android devices is reduced to 0.007% using a static filter, compared to the rate without a filter; 0.065% for iOS devices.

2.5.4 Cost

We evaluated the combined strategy of targeting the IP addresses and GPS coordinates associated with a device, and filtering out non-matching device models and apps. We used our `LONG` data combined with measurements from `IP` and `GPS` to perform *post hoc* analyses and quantify the effect of cost on recall. To compute recall of a device and filter, we compared the number of impressions that both matched the filter and had a retrievable mark, and compared it to the total impressions seen from that ad ID. We developed filters for each device using impressions during the five day training period, and tested the re-identification strategy during the following month.

Impression rates during training were determined using the `IP` and `GPS` data sets, which targeted addresses and coordinates from the first five days of impressions for each device, and reduced depending on our measurements of the popularity of the device model and app. We determined costs using these rates, multiplied by the effective CPM of our impressions which was $4.02. This CPM is very high for RTB banner advertisements, but

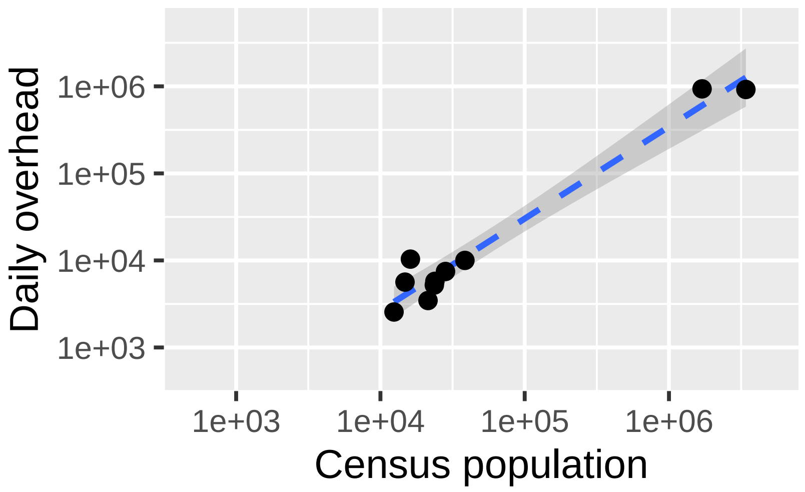


Figure 2.13: The impression rate of our ten target locations is correlated with the census-reported populations of the areas (r=0.93, p<0.0001) on a log-log scale. The upper bound of our sample had a rate of less than 1 impression per population per day.

it was intentional to maximize recall—bidding optimization could lower costs in many cases.

We evaluated the cost and recall of several scenarios, which are shown in Figure 2.14. In each scenario, the target GPS radius increased from zero to 512 km, with the cost monotonically increasing with radius. In general, targeting IP address alone results in at least 43% recall, and GPS alone results in at least 36% recall for an 8 km search radius. Combining IP and GPS targeting, this strategy achieved 49% recall on iOS, and 59% on Android, for less than $5/day. Without IP, performance dropped to 29% and 42% respectively. On the other hand, precise GPS devices could be rediscovered with greater than 70% recall for $2/day, regardless of IP targeting. Far right points (which match the entire US) resulted in recalls of 68% and 76% on iOS and Android, but these tracking efforts would cost thousands of dollars per day. Android retargeting was substantially cheaper because of the higher entropy device model targeting.

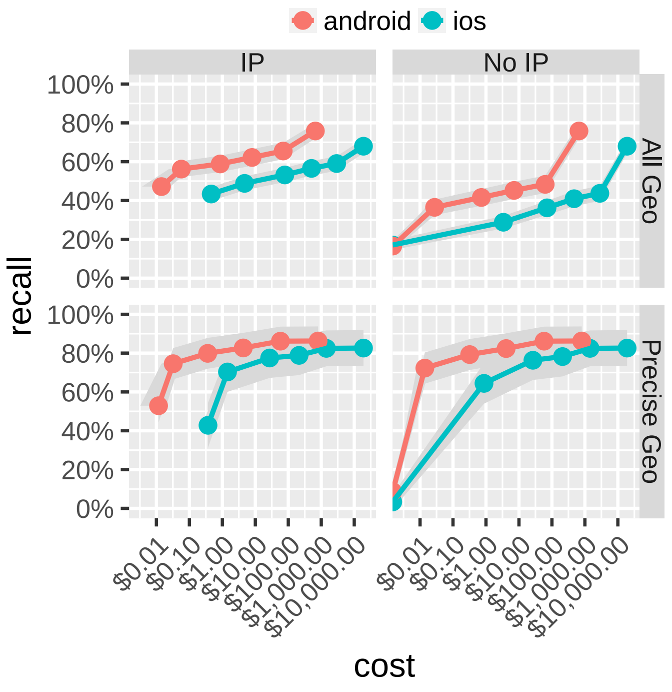


Figure 2.14: Recall over one month of testing, against the daily cost of targeting a device. We tested users with and without using IP address for retargeting. We separately tested users on GPS apps. We used search radii of 0 (exact GPS match), 1, 8, 64, and 512 km; the costs of these monotonically increase with search radii. We also tested without a GPS filter (far right points that are most expensive). Shaded regions represent error. Daily costs were determined by multiplying impression rate by cost per impression ($4.02 CPM, or $0.00402 per impression).

2.6 Case Studies

To provide further insight into the cost of rediscovering a user without ad ID, and the effectiveness of users' privacy decisions, we conducted two additional studies. In our first case study, we deployed retargeting campaigns to rediscover devices based on single impressions, and directly measured the total cost (as opposed to the *post hoc* calculation in Section 2.5.4). In our second case study, we studied a population of VPN users, and determined the ability for an attacker to find them on a non-VPN IP address.

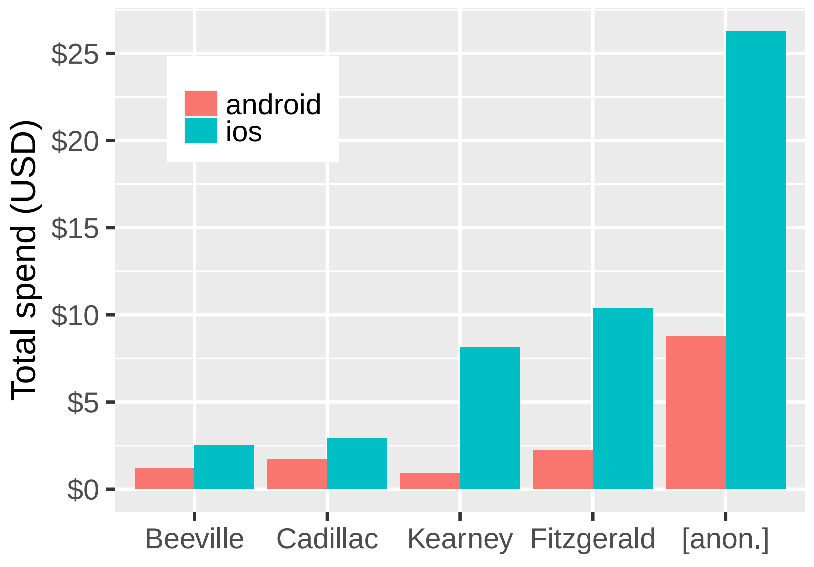


Figure 2.15: The total spend of our campaign in each city. Overall, we successfully re-discovered 660 devices out of 1,727 in 48 hours, including 288 that never reported an ad ID.

2.6.1 Case Study 1: Rediscovering Devices with One Impression

As a real-world test, we attempted to re-identify devices using a single impression and without targeting ad ID. This case study differs from from the `LONG` data set in two ways: *(i)* impressions were obtained using retargeting filters, instead of from ad ID targeting with scenarios evaluated *post hoc*, and *(ii)* we use one impression only. Since we do not target ad ID, we do not have ground truth; we simply aim to re-discover as many devices as possible while minimizing costs. This scenario allowed us to directly measure cost rather than estimate it.

We targeted five different towns and obtained a sample of 100 *device-app combinations* in each; for example, Words with Friends on Samsung Galaxy S9+. We targeted these 500 combinations for one day, within a 10 km radius from a point within each town, and used the DSP's impression capping feature to limit one impression per device. (Impression capping sometimes works for Limit At-Tracking (LAT) enabled devices; some advertis-ing SDKs assign a proprietary temporary identifier for the purpose of per-device rate limiting.) This campaign resulted in 2,204 impressions; using our marking methods, we determined that these actually came from at most 1,727 unique devices. Of these devices, 686 (40%) hid their ad ID. This proportion is higher than known proportions of devices with LAT enabled [11, 14], but we have found that some apps protect this information on the user's behalf, or require their own explicit agreement with the user before revealing such information to the advertiser.

We subsequently targeted the same five towns with the same combinations, except with a larger radius of 50 km. These were carried out in 10 campaigns (5 towns $\times$ 2 OSes), with all target apps and devices whitelisted within those campaigns. More specific targeting is possible (e.g., match app AND device rather than app OR device). In a concurrent campaign, we targeted all of the non-cellular IP addresses from the 1,727 devices. The cost of the retargeting campaigns totaled \$86.73. The IP campaign cost \$21.65, the iOS campaigns cost \$50.23, and Android campaigns cost \$14.85. The disparity is due in large part to the inability to target specific iPhone models.

Within 48 hours, we successfully rediscovered 660 (38%) of devices using mark and retarget, 433 of them using only geotargeting and not IP address. Among the rediscovered devices were 288 that never reported an ad ID.

2.6.2 Case Study 2: Re-identification of VPN Users

BitTorrent apps are well known to be a vehicle for trading copyrighted and illicit content [73, 107], and many BitTorrent users mask their activities behind VPNs. Commercial VPN services are marketed as tools to increase privacy and often explicitly declare a "no logs policy" and a lack of cooperation with investigators. VPNs are also used by journalists to evade government restrictions [84].

In this section, we quantify the effectiveness of using RTB ads to unmask mobile devices behind VPNs, using the same strategies from the previous sections. Our goal was to determine the clearnet IP (and geographic location) of devices masked by a VPN. This modified goal is much closer to how investigators would apply RTB techniques; they would not focus on impression recall.

We evaluated the two simplest mechanisms in this case study. For each device we observed using a mobile BitTorrent app behind a VPN, we attempted to

1. find the device on a non-VPN/clearnet IP given only the *ad ID*;

2. find the device on a non-VPN/clearnet IP given only a *geographic location*.

We targeted ads to μtorrent, BitTorrent, tTorrent and atorrent mobile apps[1]. The iOS appstore does not permit torrent applications (because they are used for illicit purposes), and so all devices in our study are Android. We used a commercial library to determine what IP addresses are owned by commercial VPN services.

Our campaign targeted VPN IP addresses and torrent apps, resulting in impressions on 13,778 distinct Android ad IDs. Many of the impressions were on devices without an ad ID because it was filtered out by the SSP or DSP due to a LAT-enabled flag. We then retargeted these ad IDs (on any app) but restricted the campaign to clearnet IP addresses. We were able to re-identify 3,116 (23%) of the original ad IDs. We used a maximum CPM of $10, which is far above the level needed to be the highest bidder in the auction.

2.6.2.1 Clearnet Identification using ad IDs

In the case where a BitTorrent user does not set the Limit Ad-Tracking flag, and the device shares its ad ID, the adversary's strategy is simply patience: wait for each user to not use the VPN on any app. Figure 2.16 shows the CDF of the delay between when we observed a device on a VPN and when it was observed on a clearnet IP. Half of the devices are observed within 12 days, and 16% within 5 days. About a third of the devices were not seen on a clearnet IP within 26 days.

[1]To be clear, we did not identify users beyond an ad ID, nor did we identify what torrents were shared. We did not share any data with investigators.

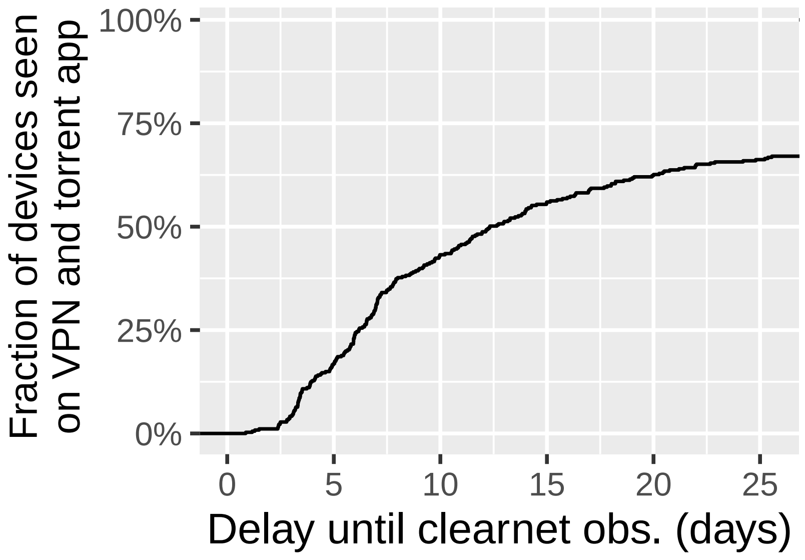


Figure 2.16: CDF of time it takes to find user on clearnet.

2.6.2.2 Clearnet Identification using Location

There is no readily apparent reason for torrent apps to require a user's geographic location. However, many torrent apps do request location access, and among the devices that shared their ad ID, 68% reported precise, 6-digit GPS coordinate at least once in response to our impressions. These GPS values must be explicitly allowed by users before the app shares with the SSP, DSP, and advertisers.

Using this information, we performed a *post hoc* analysis of geofenced retargeting of 68% of ad IDs seen on the VPN. For each ad ID, we determined the maximum radius of a geofence that would find the same device on a clearnet IP address. This result is shown in Figure 2.17. About 30% of the devices can be located within 10 meters of the location they reported while on the VPN; and about 60% within 1 km.

We also ran an auxiliary experiment to get a sense of real costs. We operated a campaign for one day that targeted BitTorrent apps in two major cities, without ad ID restrictions. By spending less than $7 on ads, we were able to re-discover 63 of the ad IDs from the `VPN` data set on clearnet IPs.

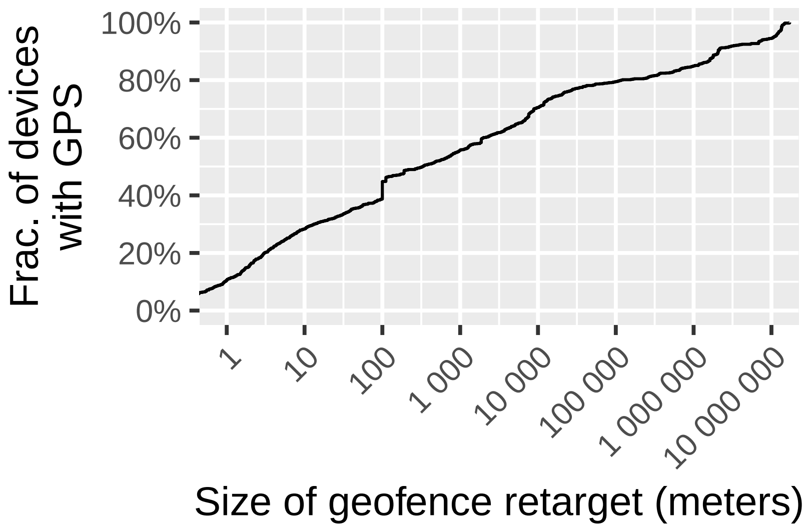


Figure 2.17: The maximum radius of a geofenced re-target to determine the clearnet IP address of a device seen torrenting on a VPN.

2.7 Discussion

Proposed privacy-preserving solution

In-app ads are small, ephemeral messages. While caching these items may save bandwidth if the same ad is called again, remaining unmarked may be worth a lot more to the user. To defend against mark and retarget, all web storage and caches that are used for advertising should be disabled when Limit Ad-Tracking is enabled.

If all storage and caching methods are disabled, attackers could only track users with fingerprinting techniques. These methods are stochastic, and do not guarantee that the device is the same.

Obfuscating the IP address and precise GPS location will reduce the risk of compromising an ad ID. To prevent precise location targeting, one could use a differentially private [19] obfuscation to continue allowing advertisers to target location while making it more difficult to target single people.

Existing RTB conditions that preserve privacy

Some advertising networks take actions that protect the privacy of users. For example, we encountered SSPs who ran JavaScript to pre-render an ad-image on a server and overlay that with a watermark; while this was likely not done specifically to protect user privacy, it did prevent users from being properly marked. We were able to detect instances of this happening by looking at the IP address, exchange, marks, and ad ID.

Apps and SSPs also try to protect the user to some degree. Some SSPs masked the last octet of the IP address to protect the user. While this alone does not prevent user identification, it does make users more expensive to target. Most apps, even ones that request location permission, do not pass precise location on to advertisers. Finally, some apps require users' explicit consent for personalized advertising before passing the ad ID to advertisers.

How users can preserve privacy in existing mobile operating systems

In current systems, users can reset their identifiers manually on a regular basis, and disable it if possible. When identifiers are reset, app cache and data should also be reset before the app is used again. IP address was the most effective targeting feature. To prevent easy retargeting, users could browse through the cellular network, or use a VPN over WiFi, though it is not always practical.

How app developers can preserve user privacy

We tested the effect of different Java methods in disabling various marking strategies. We found that Android developers who want to disable all marking capabilities must periodically call WebStorage's `deleteAllData` to delete HTML5 storage data, and CookieManager's `removeAllCookies` to delete cookies [8]. They may also completely disable some, but not all marking features: WebStorage's `setDomStorageEnabled` and `setDatabaseEnabled`, and CookieManager's `setAcceptCookie` methods affect LocalStorage, Web SQL, and HTTP Cookies respectively. However, these latter methods do not affect IndexedDB, which must be cleared using `deleteAllData`. Developers on iOS may follow a similar procedure discussed in [10].

Study limitations

Using a DSP allowed us to perform evaluations from the perspective of an attacker. This means we are prone to variations in quality of service. JavaScript sometimes fails, perhaps because the user exits an app while the ad is loading. Additionally, using a DSP to conduct this study allowed us a holistic view of the attack, but as this data is collected through a third party, there are no guarantees of randomness; we balance out known biases by collecting data using a variety of campaign parameters.

For marking, we do not look at stochastic methods. Bytecode caches store the parsed JavaScript to enable faster processing. It may be possible to infer the load time of a script to see if it has been previously compiled and cached [100]. We did not pursue any user inference methods (such as IP, usage patterns, etc.) to verify user identity, since marking was highly effective, and users generally used few apps. However, these techniques could allow advertisers to track users *between* apps, without an ad ID.

Human Subjects

The protocols used for gathering data were reviewed and approved by our Institutional Review Board, under protocols (redacted for double-blind review). The IRB determined that the lack of PII in the study, the difficulty in obtaining informed consent, and the low risk of harm to subjects, did not necessitate obtaining informed consent.

2.8 Related work

Our work is related to several papers on RTB networks. None examine the accuracy of retargeting devices using marks (only ad IDs) and none examine devices masked by VPNs.

Corner et al. [33] quantified the accessibility over time of devices to RTB advertisers, and measured features such as IP-based geolocation accuracy and bandwidth. They found that at least 14% of the population were accessible within a geo-targeted area, and found that IP-based geolocation services are generally inaccurate. (Our retargeting methods rely on geolocation consistency, rather than accuracy.) They examined device fingerprints based on features, which resulted in an accuracy of less than 50%, even if IP address is provided; our method results in zero false positives. Corner and Levine [32] explore the mobile advertising as a means of large scale scientific experimentation, quantifying the availability of sensors, and the willingness of users to click an ad to participate in an experiment.

Vines et al. [108] estimated the cost to track a single user using its ad ID. They used a DSP to track users' specific locations, behaviours, and routines. They tracked ten devices, measuring win rates, location targeting accuracy, and found that GPS (not IP) geolocation targeting was accurate and had a low update latency. Their study also included a survey of 21 DSPs and their features; the majority of them supported HTML ads, and device and location targeting. In contrast, our study quantifies the overhead cost of tracking when an ad ID is unavailable or reset. Also related is work by Olejnik et al. [86], who measured the cost to reveal users' browsing history through ad placements, and Bashir et al. [22] who gathered 35,000 impressions and modeled information flow between ad exchanges. Both studies quantify browsing privacy lost to advertising on websites; in contrast, our focus is on mobile apps.

In 2010, Eckersley [43] performed the first large-scale study of browser uniqueness using Java and Flash fingerprinting. However, these features are not available in mobile ads, and are largely obsolete in modern browsers. The marking aspect of this work is closely related to research in *evercookies* [69] and browser fingerprinting techniques. Gomez-Boix et al. [54] looked at many of these browser features, including font lists, canvas fingerprinting, and user-agent, and found that 33.6% of browsers were unique among 2 million users. Cao et al. [27] looked at canvas features and re-identified 1900 users with 99% accuracy. While these features are more available, they rely on hardware uniqueness and do not work well on mobile devices; mobile device hardware is much more homogeneous than computer hardware.

Phone fingerprinting has been studied to a limited extent, mostly at a smaller scale. Some studies look at fingerprinting techniques available to web and app developers that use additional device information or sensors. Das et al. [36] look at accelerometer and gyroscope and find some entropy among 63 mobile devices. These features are based on fine hardware differences between the inertial measurement units on devices; we were not able to replicate the results in preliminary studies using ads. Zimmeck et al. [122] look at cross-device tracking using 126 user-disclosed web histories and IP addresses, showing that both history and IP are effective in linking users' mobile and desktop usage. Finally, Smith et al. [100] look at bytecache timing attacks to sniff the history of a device. We could not replicate these results using ads due to the possibility of getting banned by the DSP.

Work by Ikram et al. [64] and Perta et al. [90] on security for VPN users is complementary to ours. Both focused on vulnerabilities and exposures present in the client software itself, such as the presence of malware and leakage of IPv6 traffic. Our results would hold even if the client software had no issues.

2.9 Concluding Remarks

We demonstrated that advertisers can re-identify devices on RTB networks even when users have taken steps to prevent it, such as disabling the device ad ID or using a VPN. We have shown that advertisers can take advantage of the permissive configuration of web views to place long-lasting marks on the device, retarget groups of devices using device profiles, and then re-identify the original device within that group. We find that at least 68% of the impressions from these devices can be re-identified without an ad ID if the attacker had a large advertising budget. For less than $5 per day, we could rediscover a device with 49% success, compared with using an ad ID, over one month. User privacy can be increased if OS developers disable web view storage by default, or give API access to disallow storage and caching within advertising web views.

CHAPTER 3

USING THROUGHPUT TO DEDUCE USER LOCATION

3.1 Overview

When users place a phone call or download data over the Internet, the other party can make inferences about the user's location simply by looking at the quality of the connection. For example, you can often tell when someone you are calling is driving and entering a tunnel.

In this chapter, I quantify the extent to which a remote attacker, including app and web services, can deduce location without the user's permission. This work, published in PETS (2013) [102] and TMC (2019) [103], was conducted in collaboration with Joydeep Biswas, Erik Learned-Miller, Brian Levine, Mark Liberatore, and Hamed Soroush. To study this problem, we conducted a simulated attack wherein we deduced a user's travelled path while streaming data the mobile device, and watched changes in the connection quality.

The remote attacker discussed throughout this chapter could potentially be anyone running a server that is accessed by a user. Even the advertisers studied in Chapter 2, who run an ad server and receive location and IP information, could attempt a similar attack on a large scale. They can implement this attack by running video ads which can last up to 30 seconds and are 2–10 MB.

3.1.1 Main results

We were interested in a subset of the Internet-based remote localization problem: *Can an attacker, providing an Internet-based sevice to a mobile user who has disabled geolocation features, infer the path taken by the mobile user from among a limited set of paths, using* *only information visible at the server?* We evaluated this issue on both a larger scale (tens of miles) and smaller scale (less than 3 miles).

Our analysis showed statistically different throughput means among small geographic areas (0.9 km^2 each). Phones that move between locations travel through consistent and distinct network conditions that are remotely observable. We leveraged these differences in our design of several classifiers.

Our best performing approach, the NB-KDE classifier, can correctly determine the path taken by the phone from one of four longer paths to neighboring suburbs with greater than 90% accuracy, and the path and direction (8 classes) with 76% accuracy. In a separate experiment involving data collected only from within a 4km^2 area, in and around our campus, the same approach could identify direction and which part of campus the user was traveling with 76% accuracy.

3.1.2 Chapter outline

In Section 3.3 I define the Internet-based remote localization problem and demonstrate for the first time that cellular phones can be remotely localized based on the quality of an Internet connection. I then describe our data collection methodology in Section 3.5, and discuss some properties of our dataset. In Section 3.6, I detail three classifiers: a k-nearest neighbors (k-NN) classifier, which trains on the ordered sequence of throughput values of each route, a hidden Markov model (HMM) classifier, which exploits the consistency in throughput values at each location, and a naive Bayes (NB-KDE) classifier that uses kernel density estimation of throughput at each second along a path.

In Section 3.7, I examine the performance of the three classifiers, demonstrating that the k-NN, HMM and NB-KDE approaches can distinguish between a small number of geographic routes taken by mobile users using only throughput measurements. We examined two different scenarios: a user traveling to a different town from campus (or the reverse direction); a user traveling within our campus. Finally, in Section 3.8 I examine the limitations of these techniques. We determined the limits on the amount of data and the length of the trace required to achieve a certain accuracy. These limits provide a starting point for the development of a defense.

3.2 Background on streaming

Streaming data typically works in two different ways. First is the User Datagram Protocol (UDP), typically used for livestreams or phone calls. The data sent has no quality guarantee; if audio is dropped for any reason, it is never recovered, and the user hears noise or silence.

The second streaming method is Transfer Control Protocol (TCP), which guarantees that data is transferred completely. Most multimedia streaming services such as Spotify use TCP. In guaranteeing that data is transmitted successfully, and to measure current network conditions, the sender receives an acknowledgement (ACK) to each data packet sent. This allows the attacker to remotely monitor connection quality.

In Internet services, the variation in connection quality is typically caused by the "last mile" problem: the last mile of a connection is the most unpredictable in quality. In cellular systems, this occurs in the broadcasting of radio waves between a cell phone tower and a mobile device. Many factors influence the last mobile mile. Chiefly, these factors are the distance from cell towers, occlusions such as buildings and trees, the individual radios of the tower and device, and even weather. Both the network layer and link layer [1] contains mechanisms to correct dropped data. The TCP throughput (i.e. data bandwidth) is directly affected by these corrections; the more retransmissions required, the slower the overall connection.

The attack I present here takes advantage of the fact that cell towers and occlusions are generally geographically static. If a mobile user travels down a well-studied road, the

[1]see `https://en.wikipedia.org/wiki/OSI_model` for more information on communication layers

attacker could match the changes in throughput to information about the typical connection disruptions that happen on that road.

This scenario in which the attacker is a server far away from the user is called a *remote localization attack*.

3.3 Attacker model

The attacker is a proprietor of a web service that sends data to a device that is in motion, and attempts to deduce the user's location without permission. The attacker is either the remote end-point of the target's communication, as is the case for TCP-based streaming services such as Spotify, Pandora, and many others, or has access (perhaps unauthorized) to network-level metadata along the network path between the end-point and the user.

We assumed that the carrier, who can localize the mobile node by examining the cell towers to which it has associated, is not assisting the attacker. The attacker does not have access to any of the internal cellular infrastructure, and cannot geolocate the user using the carrier-assigned IP address [21]. (I discussed the geolocatability of both cellular and non-cellular IP addresses in Section 2.5.3.1 of the previous chapter).

The attacker need only passively monitor the throughput on its local server as it streams data to the user. Our attacker uses only throughput measurements of the target's data stream and not the content, which could be encrypted or otherwise unavailable — though we do assume the attacker could link flows if the remote end-point IP address changes.

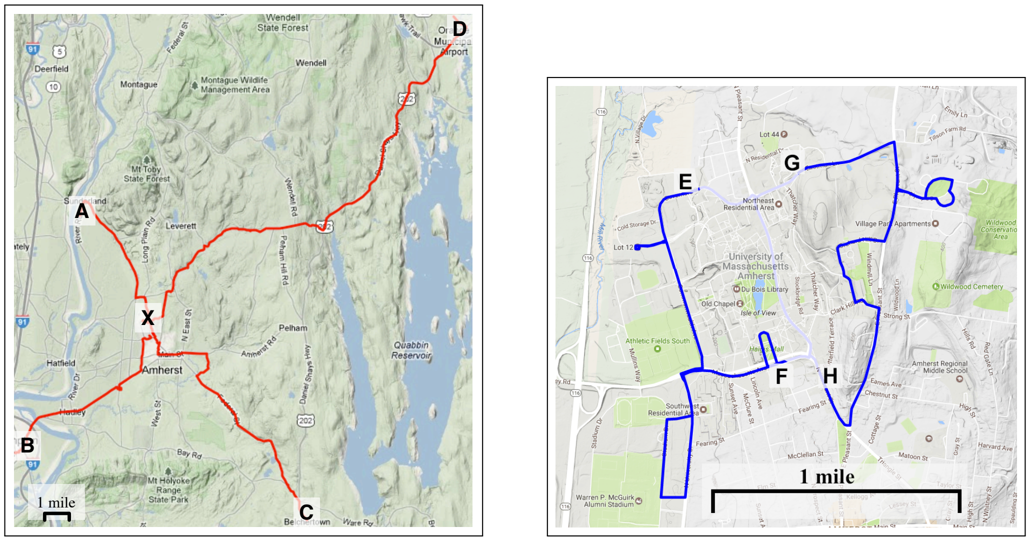
We look at two different scenarios in our experiments. First, we evaluate a scenario where the attacker knows the user's starting or ending location. This is possible if one of the user's adjacent connections are geolocatable. For example, a user starting at home or work is likely to be connected to the Internet through WiFi using an IP addressed assigned by their landline/non-cellular ISP. The attacker could also determine a user's home location from another source, such as billing information. In our second scenario, we assume that the attacker does not know the endpoint, but does know a user's coarse location. In both experiments, we assume that the attacker knows that the user is taking one of a few explicit paths.

These assumptions are widely applicable. Not only are TCP-based audio and video (i.e., music and movies) streaming apps are commonly used, but any installed app with network access could also initiate data transfers from a server without the user's knowledge. Even a web page could stream data to a mobile browser, (e.g. using JavaScript) in some cases in the background. We found this background connection could be kept alive for five minutes on Android, using the Chrome browser, but not on iOS.

3.4 Approach

Our approach builds models of the effects of mobility upon network traffic, and uses these models to determine the mobility of users. Specifically, the attacker compares a trace of network traffic generated by a mobile user against a set of models representing specific paths through a targeted geography. The attacker creates these models by gathering information about TCP's performance on a set of possible routes that he assumes the target may take. The attacker may gather this information, which consists of traces of network traffic, during any period when the traffic observed would be similar — it need not be done strictly prior to the attack. This training information may be gathered from other users that have not disabled GPS. We conjecture that greater temporal locality will improve the attacker's performance, though we did not explicitly test this assumption.

We collected hundreds of throughput traces and quantified their correlation with geography. Then we designed classifiers based on these correlations, and evaluated them against our datasets.



(a) metro (b) campus

Figure 3.1: For the ‘metro’ dataset, we gathered data on four popular paths in our area, in two directions for each path. All paths intersect in Amherst, MA, labeled as “X”. For the ‘campus’ dataset, data was collected along the campus figure-eight bus loop around the University of Massachusetts campus, which runs in both directions. We compared both directions of paths along the West (“E”–“F”) and East (“G”–“H”).

3.5 Data Exploration

We collected 407 measurements of mobile phones traveling around campus and to nearby towns, illustrated in Figure 3.1. Two hundred eighty-six of those traces — the *metro* dataset — were on paths to and from a central location to four remote locations, each about 25 minutes away (roughly a 360 km^2 area). One hundred twenty-one of those traces — the *campus* dataset — were on paths encircling a central location around a 4 km^2 area. We also collected 29 stationary traces to serve as a simple baseline to contrast the effect of motion on throughput changes.

Signal strength and throughput characteristics are tied to geography. To explore this relationship, we discretize geographic data into square areas of varying granularity (see Figures 3.4 and 3.5). In our first data set, the mean throughputs of 95% of 1 km^2 areas are statistically different from at least 90% of the other areas. In our second dataset, which cover a smaller area, the mean throughputs of 85% of 0.01 km^2 areas are different from

Route	Distance (mi)	Num. Traces Collected	Throughput (KB/s) mean ± s.d.	Duration (min) mean ± s.d.
A-to-X	7.0	63	157.4 ± 115.2	16.1 ± 4.2
B-to-X	21.2	24	63.4 ± 94.9	30.5 ± 4.8
C-to-X	10.5	29	116.5 ± 111.4	34.8 ± 7.4
D-to-X	8.5	19	34.6 ± 64.4	36.1 ± 5.1
X-to-A	7.0	68	134.8 ± 110.0	16.4 ± 4.3
X-to-B	21.2	28	47.1 ± 81.3	33.6 ± 9.0
X-to-C	10.5	31	120.1 ± 109.4	32.1 ± 7.8
X-to-D	8.5	24	45.0 ± 76.1	34.9 ± 10.3
E-to-F	2.6	57	142.9 ± 117.4	9.0 ± 9.3
F-to-E	2.6	76	126.2 ± 122.4	7.8 ± 11.2
G-to-H	2.8	53	206.7 ± 154.0	11.7 ± 4.6
H-to-G	2.8	56	194.5 ± 148.1	12.8 ± 4.5
Stationary	0	29	206.6 ± 122.5	20.4 ± 5.5

Figure 3.2: Details of the traces in our Measurement Sets. Letters refer to landmarks labeled in Figure 3.1. In total, we recorded 407 mobile and 29 stationary traces.

at least 85% of other areas. The implication is that the route traveled by a mobile node is through a largely unique sequence of mean throughputs that is classifiable. Figure 3.2 illustrates these differences in throughputs.

3.5.1 Data Collection Methodology

We recorded traces[2] of GPS location and signal strength using four Android cell phones. A server in our building streamed music continuously to the phones during measurement trials, while TCP traces were logged at the server. We later combined sets of corresponding phone and server traces, synchronized using the timestamps within the traces. Note that it is impossible without carrier participation to take measurements within the network. Moreover, our goal is to assess a weaker attacker who is without special access to cellular infrastructure, but who can take measurements at an end host. We used *Samsung Nexus S*, *Samsung Galaxy S*, *Motorola Atrix*, and *HTC Inspire* phones, all connected

[2]Traces from our experiments are available for download from `http://traces.cs.umass.edu`.

to the AT&T UMTS (3G) network, to record traces. The 802.11 radio on the phone remained powered off during the experiments. We collected the data under varying traffic and weather conditions.

We took three sets of traces:

- **Mobile 3G Measurement Set — 'Metro'**: We collected data during a one-month period. Each measurement was taken as a phone traveled along one of four routes going either toward or away from our central location (point X in Figure 3.1). The individual paths are shown on a map in Figure 3.1 and summary statistics appear in Figure 3.2. In total, we recorded 286 traces in this set.
- **Mobile 3G Measurement Set — 'Campus'**: We recorded 141 traces from the same phones, along one of two directions around a bus loop on campus. Traces were collected over a period of eight months.
- **Stationary 3G Measurement Set — Baseline**: We recorded 29 traces from stationary phones, connected to the UMTS (3G) network, located in different locations near our central location.

The phones collected traces of GPS location (with 10m accuracy) and signal strength.[3]

Each element of the traces was sampled once per second. Traces of network activity on the server consist of standard `pcap` logs. We did not limit traces to periods of cellular connectivity, and some traces consisted of several TCP connections.

In the mobile sets, we hired several persons to collect data on these specific paths, and no person was assigned to a single path or phone. Our goal was to avoid learning the phone model or user behind the movement. Each path differed in distance. In the first dataset, each took about 25 minutes on average to travel by car or bus. The travel time to location A was the shortest and D the longest. In the second dataset, two bus routes ran bidirectionally in a loop; each cycle took about 45 minutes. We later divided the bus

[3]As reported by `android.telephony.SignalStrength.getGsmSignalStrength()`.

loop data into East and West paths (see Figure 3.1). We discuss the implications of path duration on classifier bias in Section 3.7.

Because we relied on a consumer phone platform, on some occasions the experiment failed because either the end time or start time were not recorded correctly, due to a GPS failure or write-to-flash failure. We did not attempt to even out the number of traces per path after our collection period completed. Though the number of traces per route and direction varies, we did not alter which traces to collect. In our experiments, we discarded traces that did not exceed 10 kilobytes transferred (indicating a network error).

3.5.2 Geographic Analysis

We grouped all server-side throughput and client-side signal strength measurements into small geographic areas (much smaller than and having no correspondence to carrier cells) to determine if each area had consistent and differentiable mean throughput. The efficacy of any throughput-based remote localization scheme depends on such consistency. We found geographic consistency in both cases and a weak correlation between the two features.

Server-side throughput is influenced by the wireless link between the phone and cell tower [29], the network conditions and infrastructure [117] between the phone and server, the TCP algorithm [88], and other factors. Signal strength is just one factor that influences the wireless link but it is the factor with the strongest tie to geography. Received signal strength is influenced by many physical features, including occlusions between the radio and cell tower from tree foliage, the body of the person carrying the phone, buildings, and other structures. Most of these physical features do not change from one day to the next, and therefore can have a permanent effect on throughput in a particular area.

We found a weak correlation between client-side signal strength and server-side throughput of 0.24 when considering mean throughput and mean signal strength on a per-trace basis. Figure 3.3 shows the distribution of throughput values per signal strength value as

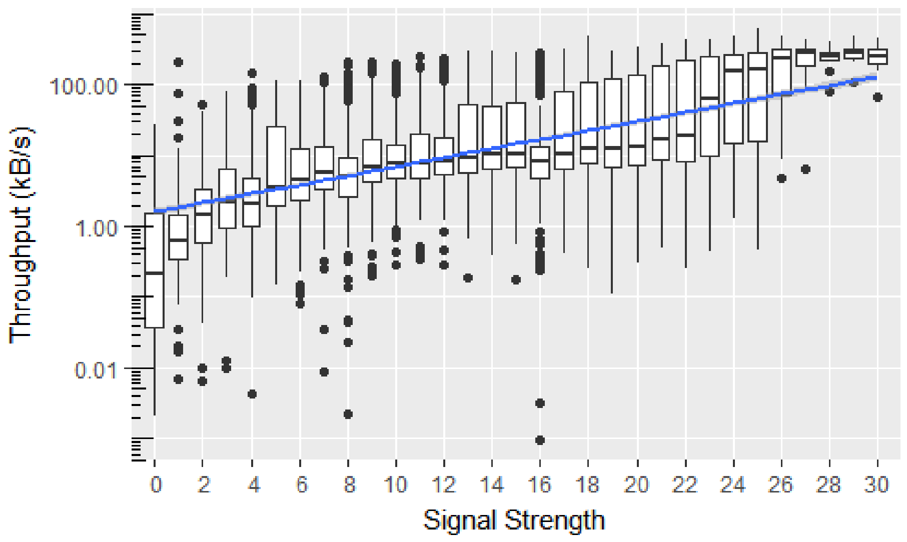


Figure 3.3: On a per-second basis, the correlation between server-side throughput and client-side signal strength is 0.24. The plot shows a linear fit. Note: Android's reported signal strength correlates linearly to decibels, which is a logarithmic measure.

a boxplot. The figure also plots the least-squares linear fit of the two variables as a visual guide. The figure is based on the range of signal strength values measured by the phones. These values can be converted to integers, as defined in GSM standard TS 27.007, with 0 referring to -113 dBm or less, 31 referring to -51 dBm or greater. Each value between 1 and 30 is a linear increase from -111 to -53 dBm. We discarded values of 31, as the unbounded range it captures is too large for a meaningful regression.

Figure 3.4 shows the mean throughput (left) and signal strength (right) of geographic areas in our measurements of paths to towns ('metro' dataset); Figure 3.5 shows the same information in left and right plots for the smaller measurement area within our campus. The error bars of each mean indicate the 95% confidence interval of the mean. Each plot is sorted by an increasing mean value, and therefore the order of areas in the plots is not the same. Using a two-sided, 95% confidence interval t-test, we performed a pairwise comparison of the mean throughput of the areas. In the 'metro' dataset, each cell is on average significantly different from 85.4% of other cells. In the 'campus' dataset, the discretized areas are smaller (i.e. finer granularity) and there is more similarity, with each cell dif-

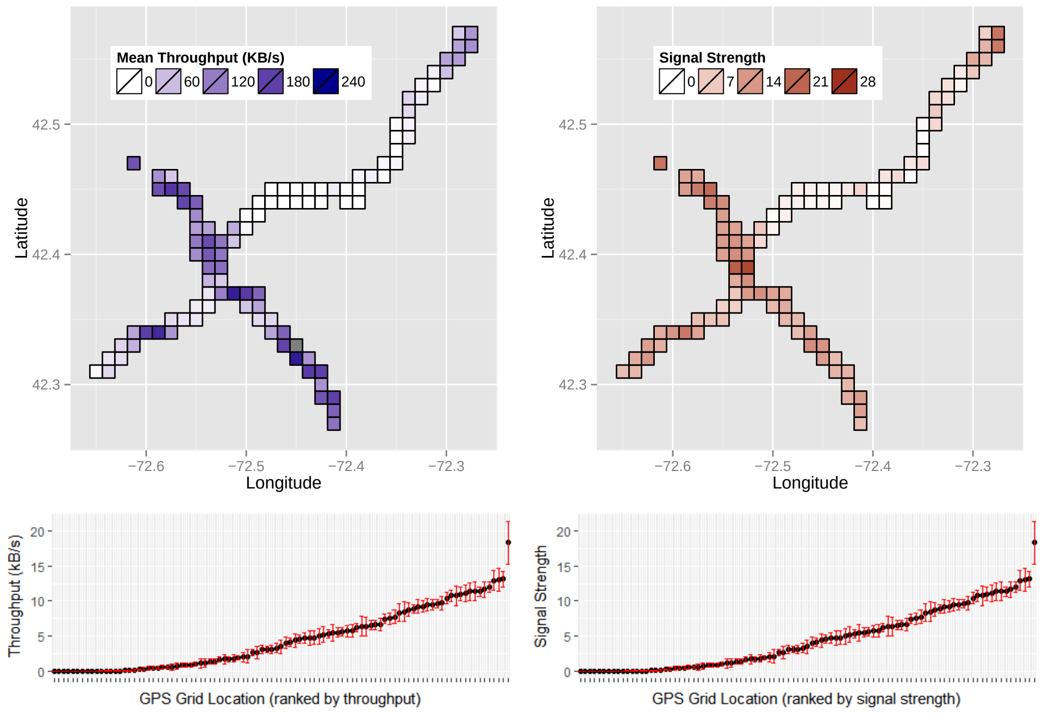


Figure 3.4: The mean throughput (left) and signal strength (right) of geographic areas in our measurements of paths to surrounding towns. On average, throughput in each cell is significantly different from throughput in 85.4% of other cells. This data suggests that latent information linking throughput and geography is available for training a classifier.

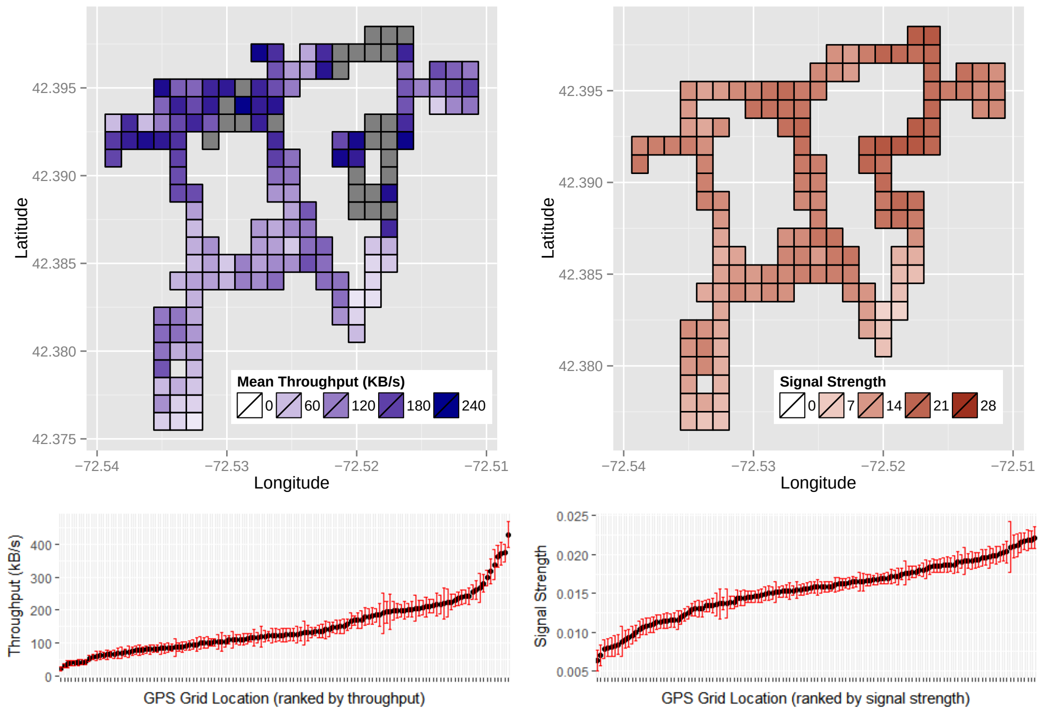


Figure 3.5: The mean throughput (left) and signal strength (right) of geographic areas in our measurements of our ‘campus’ data (a bus traveling within a 4 km^2 area). Throughput in each cell is significantly different from throughput in 73.6% of other cells.

fering from 73.6% of others. The consistency of these values and the differences among areas suggests that latent information linking throughput and geography is available for training a classifier.

The takeaway of the plots is that mobile nodes will travel through a sequence of areas that has a relatively unique signature of mean throughputs. The task of classification is to match the observed throughput to a training set that captures these means. In the simplest approach, we can classify based on the mean throughput that a mobile device obtains from visiting a series of areas. In three more-advanced approaches, we can classify based on the per-second mean throughputs of each trace. We evaluate all three approaches, detailed further in the next section.

3.6 Classifiers for Mobile Throughput Traces

In this section, we describe several *classification* algorithms that could be used to identify which path a mobile phone user is traveling down.

Classifiers build models of labeled training instances of data, and use these models to decide which class an unlabeled test instance belongs to. The instances we considered were created from packet capture (pcap) files, and the specific data we were interested in was TCP throughput. We discretized this data into one-second intervals, and treated each instance as a sequence of per-chunk mean throughputs. With one second chunks, the index of each throughput value is the time since the start of the trace. Corresponding GPS coordinates are recorded on a per-second basis as ground truth.

We present a hidden Markov model (HMM) [44], k-nearest neighbors (k-NN), and naive Bayes (NB-KDE) classifier with respect to a motion-throughput model.

Each classifier is derived from this model, with different assumptions. The HMM classifier is most general, and attempts to maximize the estimated hidden states and transitions. The k-NN and NB-KDE classifiers are more rigid, and make the assumption that users move through a certain path with consistent velocity. The k-NN classifier finds k

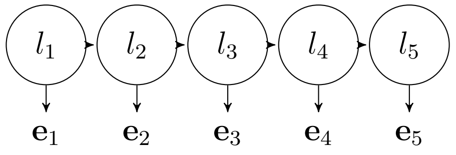


Figure 3.6: Locations are represented by hidden states in a hidden Markov model, l_i. A user moves from one state to the next with a fixed transition probability of 0.7 multiplied by the probability an observed level of throughput is seen in that state according to the emission probability vector $\mathbf{e}_i$.

of the closest traces by comparing the throughput at each time point of the trace. The NB-KDE classifier determines the distribution of throughputs at each time point for each path, and finds the most likely trace.

In our evaluation, we tested the identification of a path rather than a sequence of locations. Therefore, we have split the location sequences into separate path classes.

3.6.1 Model

We have a set of discrete locations, L, and a sequence of throughputs $\mathbf{b}$. In general, we want to associate $\mathbf{b}$ with a sequence of locations $l_{0,}l_1 \ldots$, where $l_i \in L$.

In fact, we are trying to match $\mathbf{b}$ with the most likely path (or class) $\mathbf{c}$ in a set of paths C. A path $\mathbf{c}$ represents the sequence of locations in a path: $\mathbf{c} = l_0^{\mathbf{c}}, l_1^{\mathbf{c}}, \ldots$.

In order to match a sequence of throughputs $\mathbf{b}$ $=b_0, b_1, \ldots$, we store all observed throughput at each location in L, and use this information to compute the likelihood that a location exhibits a certain throughput. We denote the observations at a location l as $\mathbf{o}_l = o_1^l, o_2^l, \ldots$.

Each classifier extends this model. The HMM discretizes L as square cells on a geographic map. However, the k-NN and NB-KDE classifiers assume the location sequences are traversed at a consistent velocity between traces, so each point in $\mathbf{c}$ represents a single second along the path. In other words, each class $\mathbf{c}$ is expected to be traversed in $|\mathbf{c}|$ seconds, and has a sequence $l_{0\ldots|\mathbf{c}|-1}$.

3.6.2 HMM

The HMM classifier can account for users traveling at variable speed down a path. However, this is a disadvantage if speed is consistent between traces, since it disregards timing information. We predicted that an HMM, which does a basic alignment of the data, would allow us to infer location with finer granularity.

We represent each class as a separate hidden Markov model. Each approximate square area (as shown in Figure 3.4) is represented by a state $l \in L$. Each state has a set of emission probabilities — the probability that a certain discrete level of throughput is seen at that state. We smoothed the observed throughput into sequences of w second windows. The value of w varies between 3-fold cross validation tests. Based on the resulting sequence, we compute the likelihood that a certain sequence of states was traversed. We choose the class that has the highest likelihood as our guess.

The states are fixed as approximate square areas on the map. Emission probabilities at each state are determined by counting the number of occurrences of each throughput level in the training data in each area and normalizing; see Figure 3.6. Sixteen throughput levels were represented by $e \in 0 \ldots 15$. The probability of a certain throughput level occurring at a certain location is from a categorical distribution.

In particular location, the probability that an emission level e occurs is

$$E[p(e|l)] = \frac{Count(e)}{\sum_{e'} Count(e)}. \tag{3.1}$$

Following the Markov assumption,

$$\begin{aligned} p(\mathbf{l}|\mathbf{b}) &= p(l_0|b_0)p(l_1|b_1, l_0), p(l_2|b_2, l_1, l_0) \ldots && (3.2) \\ &= p(l_t|b_t, l_{t-1}) && (3.3) \\ &= \prod_t p(l_t|b_t, l_{t-1}). && (3.4) \end{aligned}$$

In each sequence, we count the number of transitions between each state, and normalize. The transition from l_x to l_y is denoted by $l_{x \to y}$, and its probability by $p(l_y|l_x)$. In other words, if we are in l_x at time t, we determine the probability we are in l_y at time $t+1$:

$$p(l_y|l_x) = \frac{Count(l_{x \to y})}{Count(l_x)}. \tag{3.5}$$

Given both the emission probabilities and transition probabilities of each state, we can now compute the likelihood of the model given a test sequence (also discretized into chunks),

$$p(\mathbf{l}|\mathbf{b}) = \prod_{t=1}^{|\mathbf{b}|} p(l_t|l_{t-1})p(b_t|l_t). \tag{3.6}$$

We use the log probability,

$$\log p(\mathbf{l}|\mathbf{b}) = \sum_{t=1}^{|\mathbf{b}|} \log p(l_t|l_{t-1}) + \log p(b_t|l_t). \tag{3.7}$$

We use the Viterbi algorithm [109] to compute this log-likelihood.

We want to determine the most likely class:

$$\arg\max_{\mathbf{c}} \log p(\mathbf{c}|\mathbf{b}). \tag{3.8}$$

The HMM is the most versatile classifier, robust to varying geographical speeds because it classifies based on throughput sequence, and using a sequence to infer not only path, but location. However, if speed does not typically vary much but is unique to a certain path, then this classifier may be less useful than non-Markov models, since the former discards this information.

3.6.3 Sequence-based k-Nearest Neighbor Classifier

In the k-NN and NB-KDE classifiers, we make an assumption that subjects traveled along the path at consistent speeds. Instead of considering discrete geographic locations as we did with the HMM classifier, we represent location as equivalent to the number of seconds a user has traveled along a path. In other words, $\mathbf{c} = l_0, l_1, \ldots$ represents a virtual location for each second along a path, rather than directly mapping to a geographic location.

First, we compute a distance between the test trace $\mathbf{b}$ and training traces $\mathbf{b}^{tr} \in B^{tr}$.

$$\text{distance}(\mathbf{b}, \mathbf{b}^{tr}) = \sum_{t=0}^{|\mathbf{b}|} |\mathbf{b}_t - \mathbf{b}_t^{tr}| \tag{3.9}$$

Subsequently, we rank the sequences $\mathbf{b}^{tr}$ by the computed distance from lowest to highest. We classify the instance as the label (i.e., the route) present in the largest fraction of the k nearest neighbors. If there is a case of a tie, we increment k for that case until the tie is broken.

The choice of k tunes a smoothing effect in the data: A larger k reduces erroneous labeling due to matching against outliers, while too large of a k can result in simply choosing the most common training label.

This classifier is powerful because it does not try to model any attributes about the geographic paths — assumptions which may turn out to be incorrect. It simply votes based on similarity to the labeled traces in the database. However, its accuracy depends on there being enough training traces to properly match the unknown trace.

3.6.4 Naive Bayes with a sequence of kernel density estimators

The NB-KDE classifier makes a similar assumption to the k-NN classifier and assumes that subjects are traveling at consistent speeds. The algorithm obtains a distribution of throughputs for each location state, and finds the closest matching sequence of distributions to the observed throughputs.

Similar to k-NN, $\mathbf{c} = l_0, l_1, \ldots$ represents seconds along the path; thus, we do not consider transitions.

We wish to determine

$$\arg\max_{\mathbf{c}} p(\mathbf{c}|\mathbf{b}). \tag{3.10}$$

Each location is associated with a kernel density estimator, with Gaussian kernel K:

$$f(b|l) = \frac{1}{N_l} \sum_{i=0}^{N_l} K(b - b_i^l). \tag{3.11}$$

We use the KDE to estimate the likelihood of a certain bandwidth at a location, so

$$p(b_t|l_t) = f(b_t|l_t). \tag{3.12}$$

We consider each second as independent features in the naive Bayes algorithm. We compute the likelihood of a class c by multiplying the likelihood at each second, and choosing the most likely class.

$$\arg\max_{\mathbf{c}} p(\mathbf{c}|\mathbf{b}) = \arg\max_{\mathbf{c}} \prod_{t=0}^{|\mathbf{c}|} f(b_t|l_t^{\mathbf{c}}) \tag{3.13}$$

A kernel density estimator captures the varying distributions of throughputs at each location, as well as cases where there may be more than one distribution of throughputs at a location (for example, if the location is serviced by more than one possible cell tower). In other words, unlike the k-NN, it does not require every throughput sequence scenario to be captured in the training traces; rather, it measures the distributions at each second. By learning this information, the classifier performs much more accurately than the other two classifiers.

Path	Classes	NB-KDE	k-NN	HMM	Throughput	Frequency
A-to-X vs stationary	2	**100.0**	**97.7**	67.8	65.2	68.5
B-to-X vs stationary	2	**100.0**	92.2	45.1	86.8	54.7
C-to-X vs stationary	2	**100.0**	**96.5**	49.1	74.1	50.0
D-to-X vs stationary	2	**100.0**	**93.8**	39.6	87.5	60.4
X-to-A vs stationary	2	**100.0**	93.3	68.9	69.1	70.1
X-to-B vs stationary	2	**100.0**	**96.3**	48.1	89.5	50.9
X-to-C vs stationary	2	**100.0**	93.0	49.1	83.3	51.7
X-to-D vs stationary	2	**100.0**	**94.1**	43.1	77.4	54.7
E-to-F vs stationary	2	**68.6**	**72.5**	47.1	**60.5**	**66.3**
F-to-E vs stationary	2	**68.9**	**64.4**	37.8	**76.2**	**72.4**
G-to-H vs stationary	2	**87.0**	60.9	58.0	54.9	64.6
H-to-G vs stationary	2	**96.2**	74.4	64.1	56.5	65.9

Figure 3.7: Classification accuracy for differentiating *stationary* vs. *mobile users.* Bolded entries have the highest accuracy; also bolded are entries that are not statistically different from the highest rate (two-sided, two-sample proportion test; 95% c.i.) The NB-KDE classifier performs flawlessly for the "metro" traces.

3.7 Experimental Results

We evaluate several scenarios against the classifiers described in Section 3.6. Below, we describe these scenarios with respect to our attacker model, and then discuss the limitations of our classifiers in different situations.

Summary of Experiments and Results. First, we show that an attacker can differentiate a mobile user from a stationary user, that is, make the binary choice of mobile or stationary. The attack succeeds with very high accuracy (100% in the *metro* dataset for NB-KDE). Next, we show that given a user's starting or ending location and choice of four paths, an attacker can determine which path a user traveled, that is, it can choose the origin or destination correctly among four suburbs. This attack also succeeds with high accuracy (83.0–93.0% for NB-KDE depending on the scenario).

We then use our data to explore how well our method will scale to more choices. We show that given just the choice of four paths, an attacker can determine both the path and direction traveled (from among the eight possibilities) with good accuracy (75.7% for NB-KDE). This problem parallels the problem of choosing from one of eight known paths given a starting or ending point, but is no easier: when forced to determine both path and direction, the attacker is choosing among paths where pairs are quite similar in some respects (since they are essentially mirror images of one another). We also show that in our data, the mirroring accounts for some drop in accuracy, though it is less pronounced in the NB-KDE classifier.

Finally, we show that an attacker can identify a user's path with a fine granularity, determining the direction and path of travel within campus with high accuracy (76.5% for NB-KDE and 81.1% for k-NN).

3.7.1 Overview

Our experiments take the form of classification problems, where an attacker trains a classifier on data consisting of labeled sequences of throughput as training instances, as described in Section 3.6, and attempts to determine the class of an unlabeled test instances. Varying the classifier and training and testing data allow us to determine how well the attacker can perform under different scenarios.

Classifiers. We evaluate an attacker's accuracy using the k-nearest neighbor, naive Bayes KDE (NB-KDE), and throughput-based HMM classifiers described in Section 3.6. We also evaluate two simplistic approaches to classification. In the first simplistic approach, called *Throughput*, the classifier models each path as the mean of the mean of the throughputs associated with each path. This is among the simplest approaches to modeling a path that actually uses observed throughputs. In the other simplistic approach, called *Frequency*, the classifier simply chooses the class that was most common in the training data. Performing no better than either simplistic approach, which use no information from the data stream, implies that a classifier models the situation poorly.

In all experiments, we use the standard definition of *accuracy* for a multi-class problem: the sum of correct classifications divided by the total number of classifications. We determined optimal parameters for each classifier in separate training sets using 3-fold

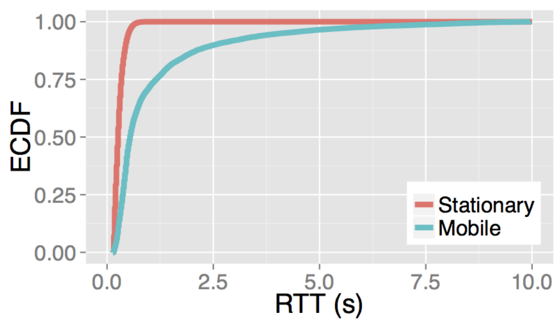


Figure 3.8: The empirical CDF of server-side roundtrip time (RTT) estimates for long-running TCP connections when a mobile device is static or moving. The same device is used for both cases. The static scenario demonstrates a noticeably different distribution of estimated RTTs compared to the mobile scenario; RTT is a primary factor in TCP throughput [88].

Data Set	Experiment	Classes	NB-KDE	k-NN	HMM	Throughput	Frequency
	4 paths$\times$ 1 (Outward)	4	**92.0**	52.9	29.7	48.3	45.0
Metro	4 paths$\times$ 1 (Inward)	4	**93.0**	47.3	69.8	48.9	46.7
	4 paths $\times$ 2	8	**83.0**	35.6	11.5	26.6	23.8
Campus	2 paths $\times$ 2	4	**76.5**	**81.1**	43.2	48.8	31.4

Figure 3.9: Classification accuracy depending on which roads are included in the experiment. Bolded entries are not significantly less accurate than the most accurate result. Outward indicates "X"-to-. . . , and inward indicates the opposite. Traces were truncated to 10 minutes long.

cross-validation. These were the value of k for the k-NN, the bandwidth selection method (Silverman's [99] or Scott's Rule [97]) for the NB-KDE, or the window size w for the HMM. Using these parameters within the held-out test set, testing and training was done using leave-one-out cross-validation. We evaluate the named classifiers themselves, not any of the specific parameters.

Assumptions and Limitations. Our attacker model and assumptions are described in Section 3.3. Our techniques assume the attacker knows the starting (or ending) location of the user. We assume the attacker has trained on all possible paths the user could have taken from (or to) that location.

Our measurement study is limited to one geographic location that is a small town with few tall buildings, as well as several paths to surrounding towns. Other small cities may be different, and cities replete with tall buildings may have very different characteristics. The speed of the mobile node was dictated by local traffic. We have no data on walking or bicycling targets.

We used Subsonic[4] to stream a constant bitrate mp3 from our server to the phone, resetting the cache before each run. However, a real target might not stream data the entire length of the path (or at all), and hence we gave the attacker an advantage. Further, we did not model the complexities of commercial streaming services, which may not stream from a single location on the network. Finally, our measurements do not include any competing traffic flows to or from the phone during trace collection, which may complicate classification in practice.

3.7.2 Differentiating Stationary and Mobile Users

In our first experiment, we compare our instances of data from stationary phones against those from mobile phones. In this set of experiments, all stationary traces are one class, and mobile traces corresponding to each of the routes shown in Figure 3.1 are treated as the other class. The results are shown in Figure 3.7. We see that the NB-KDE classifier works perfectly, and the k-NN classifier is near perfect. Both are noticeably better than simply comparing raw throughput information. As we show in Figure 3.8, there are obvious differences in RTT estimates, and RTT is a prominent factor in TCP throughput [88]. Dramatic variations of the estimated RTT are likely the result of the increased number of local link-layer retransmissions, which seek to mitigate the impact of wireless losses on TCP [29]. These retransmissions are more common in our mobile scenarios.

[4]`http://www.subsonic.org`

Data Set	Experiment	Classes	NB-KDE	k-NN	HMM	Throughput	Frequency
Metro	A-to-X vs X-to-A	2	**88.3**	61.4	33.3	61.1	51.9
	B-to-X vs X-to-B	2	**96.8**	62.6	43.5	57.7	53.8
	C-to-X vs X-to-C	2	**94.3**	66.5	29.6	50.0	51.7
	D-to-X vs X-to-D	2	**95.5**	63.8	90.2	67.4	55.8
Campus	E-to-F vs F-to-E	2	**88.4**	55.1	**78.3**	63.2	57.1
	H-to-G vs G-to-H	2	**99.0**	82.3	74.0	59.6	51.4

Figure 3.10: The NB-KDE was able to differentiate between directions for most paths. The exception was "X-to-D" which was relatively symmetrical. Bolded entries are not significantly less accurate than the most accurate result.

3.7.3 Determining a User's Path

In our next set of experiments, we assume that the user's starting or ending location is known, and that our goal is to determine which of four paths were taken by the device. In these experiments, we truncated each trace to the first ten minutes to avoid biases introduced by the varying trace length between each class. In the *metro* situation, we train and test classifiers using only traces that are *Inward* to the central location ("…to X") described in Section 3.5; then we do so again, using only *Outward* ("X to …") traces. Each set of these experiments considers four classes. We also classified path and direction (8 classes). The results of these experiments are shown in Figure 3.9.

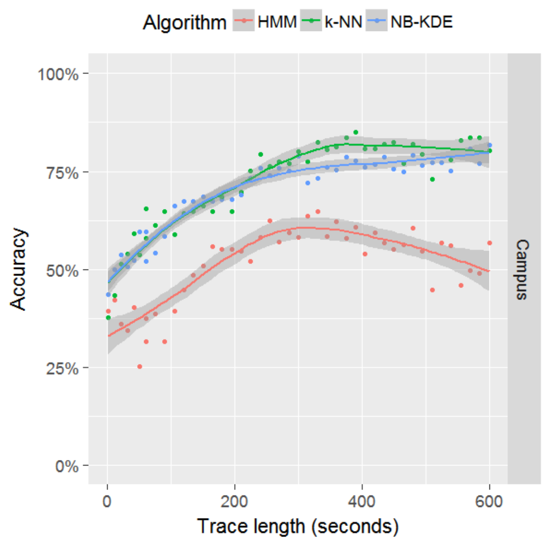
The NB-KDE and classifier significantly outperform the naive classifiers. The HMM does not work when there is a time-independent sequence of throughputs that are similar between two paths. This is particularly the case for paths X-to-C and X-to-A, which are both paths away from the more urban location X. If the value of k is tuned, the classifier achieves greater than 70% accuracy [101]; however, it does not perform well if k is chosen using a separate training set. Figure 3.12 shows the confusion matrix of the NB-KDE results. It misclassified the "A-to-X" path direction particularly often, indicating that the first and last ten minutes of this path are relatively symmetrical.

In the *campus* situation, both the k-NN and NB-KDE performed well. As shown in Figure 3.12, The NB-KDE classifier often misclassified the "F-to-E" as "H-to-G"; these are both south-to-north paths, which go from low-throughput to high-throughput. In both cases, the HMM performed poorly because while it may be able to learn the sequence of throughputs associated with each path and direction, it discards all timing information. Upon further analysis of the guessed states, there were many cases where the HMM assumed that the subject traveled either exceptionally quickly or slowly.

As we discussed in Section 3.5, the reason classification is possible is that throughput is geographically consistent in our dataset (see Figure 3.4). The experiments in this section demonstrate that path-level classification can make meaningful use of such consistency. In line with our intuition, we find that naive measurements of throughput alone does not adequately differentiate routes, but classifiers perform significantly better.

We found a positive correlation of 0.15 between throughput and trace, shown in Figure 3.3. The lower correlation — compared to a correlation of 0.24 at the per-second level — speaks to the challenge of this task: network performance is generally consistent for a path but it weakens significantly for shorter time scales. Additional features from the network traffic are likely needed to advance classification accuracy to work with finer time scales or geographies. Mobile usage patterns may also be useful in accounting for variances between traces [67]. A non-Markovian hidden state model [68] may be able to account for varying speeds, while still preserving timing information that is consistent within a class.

Determining direction given a path. As Figure 3.10 shows, the NB-KDE is best able to classify the direction given a path. The NB-KDE performs well because it differentiates the unique seconds along a path in each direction. Occasionally, the HMM does well: it benefits from the duration and length of both directions in each class being similar, thus it is unlikely to incorrectly infer a class due to misalignment. As well, the general trend of throughput decreasing in the "X-to-…" *metro* traces, or north-to-south *campus* traces is easily modeled. The k-NN sometimes fails by erroneously matching the middle of each path, which are likely consistent in both directions.



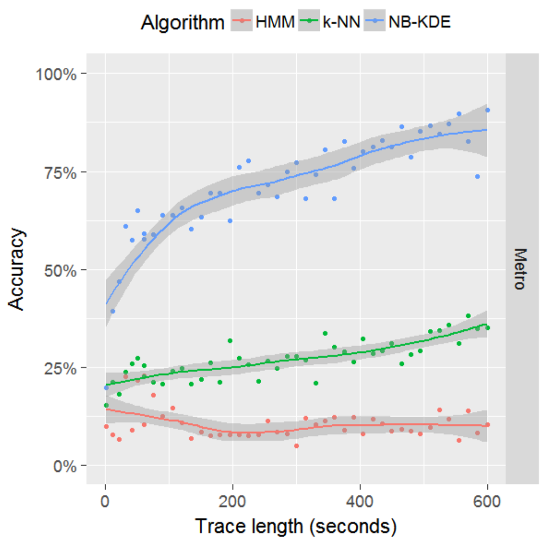


Figure 3.11: We analyzed trace length vs accuracy to evaluate the efficacy of the attack if a connection session length is limited. The left graph indicates the *campus* experiment, and the right graph indicates the 8-class *metro* experiment (Figure 3.9 indicates the results at 600 seconds). The HMM performs poorly in both cases because it confuses classes in which similar sequences, regardless of timing, occur. This was particularly true among the outgoing paths in the *metro* data.

3.7.4 Effect of trace duration on accuracy

A potential target may wish to protect her privacy by limiting her session length. We investigate the accuracy of each classifier, given different trace lengths.

Figure 3.11 shows the accuracy the classifiers given trace durations of 1–600 seconds. All three classifiers classify the *campus* paths more accurately when traces are longer. Most traces were classified within five minutes. In the 8-class *metro* experiment, the NB-KDE far surpassed the accuracy of the two other classifiers at all lengths. The HMM did not work in this case — wide variations in throughput of all traces strongly biased the classifier towards one class (i.e., path). The k-NN classifier improved linearly with trace length. The NB-KDE notably could identify traces with approximately 60% accuracy within one minute. This indicates that even short connection sessions may be enough to reveal information about a person's location.

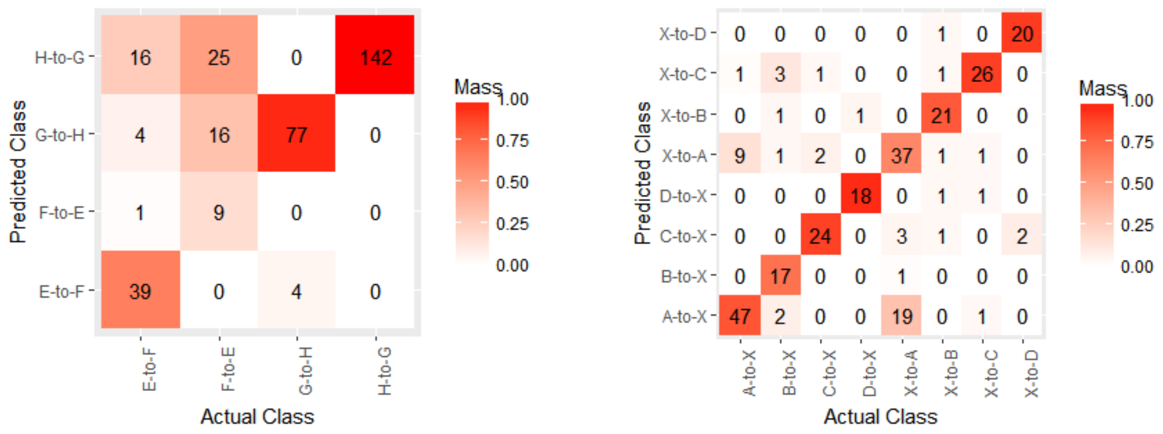


Figure 3.12: Confusion matrices for the NB-KDE classifier in the corresponding *campus* and *metro* experiments for 600 seconds.

3.8 Discussion

3.8.1 Approaches to Enhancing Privacy

Besides limiting trace duration, we did not test any approaches for enhancing the privacy of users against this attack, but many existing techniques are likely to be effective. To prevent revealing their travel paths to nosy remote servers, phone users will need to traffic shape or otherwise perturb their data transmission, incurring a performance penalty.

For example, a trusted proxy located outside the cellular network can re-shape traffic before reaching the attacker. As noted in Section 3.3, we assume the carrier is not assisting the attacker. The proxy could be set up as a VPN as a traffic shapper, which we suggest not for the encryption but because it is a protocol widely supported by smart phones as a transparent method of redirecting traffic.

It is feasible that the mobile device could reshape the traffic on its own. Most simply, it could limit throughput to a peak level that is reasonable across a wide area of the cell network. Or it could enforce regions of zero throughput. Of course, the challenge is to shape traffic in a way that does not overall reduce throughput or the interactivity needed by the application. This type of shaping would be easy for bulk file transfer where perhaps no interactivity is required. However, traffic shaping would be more challenging for interactive audio and video calls, where users are most sensitive to throughput limitations and network delay jitter. Traffic shaping of interactive web browsing may also be a challenge, but caching and pre-fetching may help mask throughput ceilings.

3.8.2 Limitations

The traces we consider are on the order of tens of minutes long, and we also evaluate how trace length affects accuracy. The user must be traveling at a similar velocity at each location with limited deviations from a typical training trace. All our measurements were taken by users riding public buses, except one path where the measurements were taken from a car. In all cases, the vehicle went at the pace of the road and congested traffic, if any. Finally, our main result relates to selecting a path from limited possibilities rather than the full set of roads on Earth.

3.9 Related work

Mobile Phone Localization. Precisely localizing mobile phones or other similar devices on the basis of GSM and other location-explicit information is an active area of research. However, these works use information available only to the mobile user (such as which 802.11 base stations or cell towers are in range [59, 105]) or their carrier (such as the pattern of handoffs [23] or other administrative details [116]). In our work, we focus on *remotely* localizing another party based only on a TCP traffic stream rather than local information.

Kune et al. [77] propose a technique to test if a user is present within a small area or absent from a large area by simply listening on the broadcast GSM channel. The focus of their work is on lower layers of the GSM communication stack. We did not extend our study to analyze lower layers of 3G, because it is a legal violation in our jurisdiction. And again, our study is concerned with remote observation of network streams over cellular links, and largely treats the cellular infrastructure as a black box.

Xu et al. [116] present an approach for localizing performance measurements in 3G networks. They exploit the predictability of users' mobility pattern to develop a clustering algorithm for grouping related cell sectors and assigning IP performance measurements to fine-grained geographic regions. The proposed technique requires access to the cellular infrastructure. In contrast, our technique for network-based localization requires remote passive observation of the target, and data collection independent from the target and internals of the infrastructure.

Balakrishnan et al. [21] show that individual cell phones can expose different IP addresses to servers within time spans of a few minutes, and find that IP-based geo-localization is "impossible" in cellular networks. They show that application-level latencies can differ greatly among cities thousands of miles apart. Moreover, they show that the variation of latencies in short time spans is not high. Our work is complementary and extends similar notions further: we show the consistency of throughput at the finer granularity of square-kilometer regions, and we demonstrate successful classification experiments using such features.

Xu et al. [117] show that in contrast to wired Internet traffic, current cellular data traffic traverses through a limited number (4–6) of Gateway GPRS Support Node (GGSNs), which is the first IP hop of a data connection. The authors show that local DNS servers provide an appropriate approximation to estimate a user's network location (i.e., one of the 6 GGSNs) for purposes of mobile content placement and server selection, due to the restricted routing in cellular networks. By assuming availability of partial information about the possible routes a user could be on, our approach aims at a much more granular localization than the DNS method proposed by Xu et al., which is limited to finding approximate network locations.

Cellular Network and User Characterizations. An investigation of the performance of multimedia streaming over cellular networks is presented by Chesterfield et al. [31]. The authors show that the key constraint for the streaming applications is the interactivity level of the user. We have only considered non-interactive music streaming in this work. An open problem is extending our attack to other types of applications such as video streaming and VoIP.

Chan et al. [29] evaluate the impact of variable rate and variable delay on long-lived TCP streams over 3G wireless links. The authors demonstrate that local retransmission mechanisms used in cellular deployments to reduce the impact of losses on TCP throughput and to improve channel utilization come at the expense of increased delay and rate variability. We show that there are predictable differences among performances of a cellular network with regards to throughput and delay in different geographical locations that could be used for remote localization purposes.

Other Remote Attacks. Kohno et al. [74] present a technique for fingerprinting a physical device remotely by exploiting clock skews. Their approach could be used to remotely identify the same device connected to the Internet at different times or using different IP addresses. Our approach, which is focused on detecting the routes taken by a mobile node, is orthogonal to this work and could benefit from it when locating the end-points of a target's travel path.

NAT and firewall policies of cellular carriers are explored in the work by Wang et al. [112]. They identify a set of such policies that directly impact performance, energy, and security of mobile devices. For instance, they show that NAT boxes and firewalls set timeouts for idle TCP connections, which sometimes lead to a significant waste of energy on the mobile device. The authors show that in spite of the deployment of firewalls, cellular networks are still vulnerable to denial of service and battery draining attacks. In contrast, we explore another type of attack on the location privacy of mobile cellular users.

Perta et al. [91] identify mobile phone IP addreses by sending several PUSH notifications to target devices and measuring round-trip times, assuming the network carrier assigns public IP addresses. They exploit variations in the cell network architecture to narrow down a user's IP to about 1000 addresses within 20 messages. This may be enough to identify coarse grained location information about the user.Remote localization has been studied in some previous work. Kohno et al. [74] present a technique for fingerprinting a physical device remotely by exploiting clock skews. Their approach could be used to remotely identify the same device connected to the Internet at different times or using different IP addresses. Our approach, which is focused on detecting the routes taken by a mobile node, is orthogonal to this work and could benefit from it when locating the end-points of a target's travel path.

Perta et al. [91] identify mobile phone IP addreses by sending several PUSH notifications to target devices and measuring round-trip times, assuming the network carrier assigns public IP addresses. They exploit variations in the cell network architecture to narrow down a user's IP to about 1000 addresses within 20 messages. This may be enough to identify coarse grained location information about the user.

My work extends these studies by exploiting TCP throughput itself, an attack which can only be mitigated by artificially slowing the user's connection.

3.10 Concluding remarks

We have demonstrated that the patterns of data transmission between a server on the Internet and a moving cell phone can reveal the geographic travel path of that phone. While the GPS and location-awareness features on phones explicitly share this information, phone users will likely be surprised to learn that disabling these features does not suffice to prevent a remote server from determining their general mobility. Our work shows that a naive Bayes classifier with sequences of kernel density distributions can discover and exploit features of the geography surrounding possible travel paths to determine the path a phone took, using only data visible at the remote server on the Internet and training data collected independently.

While we had hypothesized that simpler alignment methods such as dynamic time warping [26] or hidden Markov models could improve accuracy, these failed because they discard timing information that is consistent within a path. More complex alignment such as hidden semi-Markov models [68] may be more of a threat in this regard. If the attacker has access to much more training information (for example, via a popular mobile application), they may be able to model a user's position in 2-D space using a Gaussian process [51], rather than only along a specific path.

In the present chapter, we have demonstrated that the attack can be successful with traces that are only one minute long. It is an open and important problem to quantify the extent to which a user's location can be compromised in this fashion — with greater accuracy and among larger numbers of paths and different geographies — and to determine just how much information is needed to make these inferences.

CHAPTER 4

ESTABLISHING A PROTOCOL FOR ANONYMOUS USAGE OF A CELLULAR NETWORK

4.1 Overview

In the previous two chapters, we examined remote attackers who try to determine user identity and location over the Internet, and suggested some defenses against these attacks. In this chapter, I explore the feasibility of continuing to use a mobile network provider despite not trusting it with identity and location. The naïve solution would be for users to use a prepaid mobile phone that is not associated with their identity. This is expensive, and the user identity could be discovered through either trajectory inference attacks or location profiling.

This project, submitted to WiSec (2020), was completed in collaboration with Brian Levine, Mark Liberatore, and Mariya Zheleva. We introduce a protocol, *ZipPhone*, that precludes this need for trust between user and service provider. This protocol uses ephemeral pseudonyms as a method of accessing the network anonymously. Connecting anonymously (without a pseudonym) is not viable; users could not be billed for service, and connected sessions could not be maintained.

The idea behind ZipPhone is for users to purchase and use different cell phone identities several times a day, and change identities when they are in the presence of other ZipPhone users to prevent the old identity from being linked to the new one. This process is called “mixing”. For successful mixing, users must remain offline long enough to be confused with other users, which results in a reduced utility of service.

The main focus of this study was to quantify user privacy in the face of linking and profiling, not to rigorously test the technical details of its implementation. However, we did propose a technical implementation, published in MoST (2014) [104], that is compatible with the current cellular infrastructure. The protocol itself is not only compatible with the current cellular network, but also robust to future methods of connectivity that may be more decentralized, and evaluate its efficacy.

4.1.1 Main results

We quantified the *privacy-utility tradeoff* for a user who wants to continue using a phone service anonymously. Since users behave habitually, they are susceptible to *location profiling* attacks, which match unlabelled sequences of locations to users based on their historical movement habits.

We simulated a ZipPhone implementation on two mobile call detail record datasets collected from real users [42, 83]. These simuations represent small deployments of the protocol with 100–150 users in the same locale using uncoordinated mix zones. Despite the small number of users, we show that privacy gains are significant, especially for users who are not unique but are highly predictable; a larger deployment would increase privacy while maintaining each user's utility and performance. We determined that predictable users (e.g. ones with a daily routine) are identifiable 69% of the time, but could reduce that to 24% if their location behavior is similar enough to others who also have ZipPhone, requiring only a 5% sacrifice in utility. Additionally, this has a minimal impact on battery life.

We conclude that it is indeed possible to use a wireless service without trusting it.

4.1.2 Chapter outline

In Section 4.2, I discuss the ideas behind profiling and linking, and how ZipPhone protects against them. To our knowledge, prior work had not analyzed the combination of both attacks. I also discuss two important user traits related to their geographic behaviour: predictability and mixing, which underpin the attainable privacy against these attacks. The ZipPhone protocol and formal problem statement are stated in Section 4.3. In

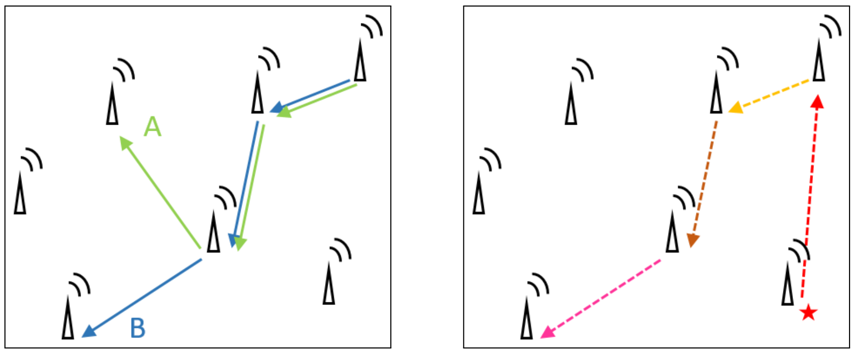


Figure 4.1: Left: Diverging paths that are regularly taken by two users. During training, an attacker would encode each labelled transition into a transition matrix for *location profiling*. Right: Separate, unlabelled activity where an unknown user reconnects using a new pseudonym at every tower. If the anonymous user does not successfully mix at these towers (i.e. does not remain offline long enough), the attacker can *link* the trajectories together and match the concatenated trace to User B's profile. Figure 4.2 show examples of this generated from users in the dataset.

Section 4.3.3, we introduce a sophisticated location privacy attack to reveal the identity of anonymous cell tower traces. We probabilistically link traces together to more accurately identify users using knowledge of users' historical movement patterns. We also describe a strategy to avoid this reidentification attack.

We evaluate our approach on two data sets collected from real cellular users [42, 83] in Section 4.4. Finally, in Section 4.5, we discuss how our model ZipPhone protocol can be employed in emerging mobile cellular networks without explicit cooperation of the provider. We also estimate the incurred battery use from ZipPhone for 3G and 4G networks. Specifically, we measured power consumption during network association and disassociation, and we estimate that a user may incur at most 1% battery overhead per day regardless of network technology or desired privacy if ZipPhone were used. In Section 4.6, we detail the limitations and ethical implications of this study.

4.2 Background

When mobile users connect to the Internet, they authenticate to a cell tower, allowing service providers such as Verizon and AT&T to store a log of the time, radio tower, and user

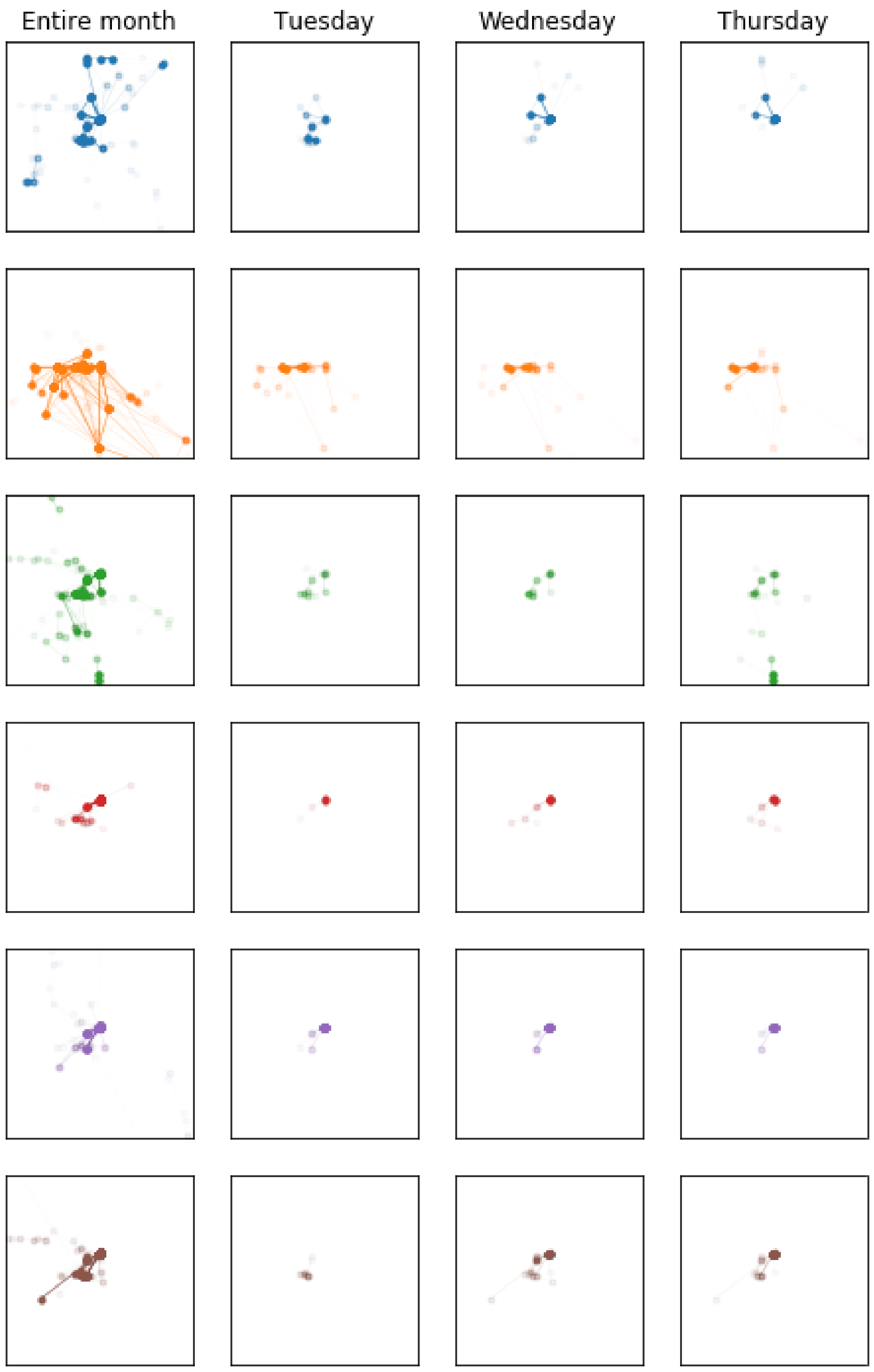


Figure 4.2: The movement of six users throughout a month, and on three weekdays within that month, produced from a log of tower transitions and the towers' estimated locations (each row represents a different user). Nodes and edges that are more opaque occurred with higher frequency. The axes are a consistent latitude and longitude, which have been redacted out of an abundance of caution to protect the privacy of the users. Each user displays somewhat routine (predictable) behaviour. However, the bottom three users (red, purple, and brown), who show similar activity, would benefit the most from ZipPhone.

identity [120]. As providers have advanced towards the current fifth generation of cellular networks, the density of towers has grown, allowing these logs to capture users' location with increasing precision. Many users are persistently connected, apprising providers of their location all day. Connecting to a large private Wi-Fi network provides similar information to its administrators. And some ISPs offer cable, cellular, and Wi-Fi hotspots as a unified package.

Users concerned about their location privacy [20] may use existing tools that allow protection only at the network and application levels. For example, VPNs and Tor [40] mask the IP address of a user from a *remote* server, and hide the remote server location from the service provider. Additionally, access control features allow users to hide or reduce location information sent to location-based services. No such tools exist for *protection of geographic locations from local service providers* — but that does not mean that users are complacent about their ISPs having knowledge of their locations. A recent class action lawsuit demonstrates that mobile users do not want cellular service providers to sell their historic movement records to third parties, such as *location aggregators* [28].

To gain privacy, a user u may attempt to anonymously use a wireless service by obtaining a mobile identity i_1 without revealing personal information. The user may switch to a new pseudonymous identity, i_2, before the first is compromised, eventually going through a series of identities over time [25]. In this scenario, two primary attacks prevent the user from having location privacy, as illustrated in Figure 4.1:

1. in *location profiling*, an attacker identifies one or more of the identities $i_1, i_2, \ldots$ as user u by exploiting the uniqueness of the locations the user is known to regularly visit.

2. in *trajectory linking*, an attacker infers that activity by i_1 is linked to activity by i_2 despite the change in identifier. The union of locations can enhance the success of location profiling.

A real-world demonstration of these transitions are shown in Figure 4.2.

There is a fundamental location privacy cost to connecting to a mobile service. To reduce the success of these attacks without modifying their behaviors, users can *(i)* switch identifies frequently, and *(ii)* remain offline for a period of time between connection sessions, which both reduce user utility. In this chapter, we model and quantify this trade-off between utility and location privacy.

Our work complements existing research in location privacy. Location profiling has been long known to be a problem [38]; attacks typically classify either the set of locations cells visited by an unlabelled user during a time period, or the list of transitions between locations [82]. Trajectory privacy studies, including a body of work in VANETs [58, 79], generally link disconnected traces using Euclidean information. Defenses against these attacks generally utilize a mixing strategy or, more recently, differential privacy. While the latter can separately protect against either location profiling or trajectory linking [45, 114, 115], it requires the cooperation of ISPs. In contrast, our work assumes the ISP is an adversary, and we evaluate robustness against attackers using both profiling and linking.

For our analysis, we model defensive strategies as a protocol we call *ZipPhone*, and we define specific ISP-based attacker algorithms as well. We assume a set of users employ ZipPhone, using ephemeral identifiers and go offline to prevent trajectory linking. Notably, users do not need to coordinate mixing; naturally occurring mix zones are enough to significantly reduce linking success. Our attacker model looks to historical transition probabilities to model linking, rather than Euclidean distance. Using two real-world datasets [42, 83], we quantify the *path predictability* and *mixing degree* of user activity. With the same data, we demonstrate how a small community can reduce an attacker's re-identification accuracy substantially while sacrificing a limited amount of utility.

4.3 Attacker and Defender Algorithms

Our primary goal is to quantify the privacy-utility trade-offs present in systems that provide geographic anonymity from mobile ISPs. To do so, first we instantiate a specific protocol for users and provide well-defined attacker algorithms. The protocol, ZipPhone, is based on mechanisms available to the user only; i.e., the ISP is not cooperative, an assumption not shared by many location privacy systems. In short, users can control only their active identity (i.e. pseudonym) and whether or not they are connected; providers attempt to link the activities of identities to existing user profiles.

4.3.1 Problem Statement

ZipPhone users seek to use the network, but not have their real identities associated with their mobility traces. Upon joining the network, the user u is assigned a pseudonym i. The pseudonym lets the user maintain a connection session for some period of time. The user attaches to a sequence of towers as it moves according to signal strength and the corresponding handoff procedures. By registering as identity i and then moving, the user provides to the ISP a *trace*: $(i, (s_1, s_2, \ldots))$, where each value of s indicates a specific wireless transceiver and a timestamp. The provider knows the locations of the transceivers and can, thus, trace a user's mobility. It is not the goal of the user to hide that they are using ZipPhone.

The goal of the attacker is to infer and label their identities from the traces. The attacker is a wireless provider such as a Mobile Network Operator (MNO) that already has a history of traces for each ZipPhone user. The attacker then tries to determine which user from a set $u_1, u_2, u_3, \ldots$ is the one that created the trace $(i, (s_1, s_2, \ldots))$ based on a classifier trained from the known history.

In Section 4.4.3, we demonstrate that longer traces are easier to identify and link with other traces; users should regularly renew their identifier in order to keep these traces short. We assume the user does not perturb their own movement patterns. Therefore important parameters are *(i)* the identity renewal frequency, and *(ii)* the user's offline duration. When the renewal frequency is higher, privacy also increases; but each identity renewal incurs an offline period and increases power usage. Longer offline durations improve privacy but reduce utility. We assume all such parameters are public and known to the attacker.

4.3.2 Attacker Model

We assume the attacker *(i)* has all traces of all ZipPhone devices, and *(ii)* has labelled traces of historic movement for all ZipPhone users in order to train a classifier; in other words, the attacker is a service provider such as a mobile network operator. The attacker performs *trajectory linking*, which patches together separate traces if a classifier predicts they are from the same user.

We assume that all ZipPhone users are of equal interest to the attacker, and that it uses only normal cellular infrastructure to attack. For example, we assume that the attacker does not install cameras on towers to identify users via facial recognition, nor would they follow a particular user by car. It does not make sense for the attacker to set up an IMSI catcher [35] since they already own the entire real infrastructure. We assume that location accuracy is on the level of cell tower; while features such as RSSI or TDOA could locate wireless devices with more precision, devices could in turn artificially slightly reduce performance as a defense, effects of which are outside the scope of this study.

We assume that the attacker gains no other information from the users; in mobile phones, information such as IMEI, device model, or OS signatures, are easily turned off via OS settings. In practice, such features would assist the attacker (see [34]), but are not the focus of this research as they are more easily obfuscated or falsified than real geographical movement. For example, IMEIs, which are akin to a MAC address, can be modified by the user since she controls the handset hardware (e.g., SilentCircle's blackphone [6]). Users are likely identifiable by the unique set of outgoing calls they make; however, they can

```
1: utility ← Minimum utility between 0.0 and 1.0
2: max_off_time ← Maximum time offline during renewal
3: while device is online do
4:     WAIT(until device moves outside range of tower)
5:     DISCONNECT
6:     off_time ← UNIFORM(0,max_off_time)
7:     WAIT(off_time)
8:     CONNECT                                    ▷ connect with new identifier
9:     cooldown_time ← utility × off_time
10:    WAIT(cooldown_time)
```

Algorithm 4.1: User identifier renewal strategy (*ZipPhone*)

make calls via VoIP through an anonymizing proxy or circuit rather than using the cellular carrier. Encryption of the VoIP stream can thwart carrier eavesdropping. Stronger protection is available by using VoIP over Tor [7].

On the other hand, a user's reidentifiability depends on their predictability and mixing behaviour. A user who visits vastly different location than her peers could not mix easily; her activity could be easily linked and profiled. And a user who is not predictable could not be easily identified regardless of mixing behaviour.

4.3.3 Attacker-defender dynamics

In this section, we define the exact algorithms used by the ZipPhone user and the service provider attacker.

4.3.3.1 User strategy

Algorithm 4.1 defines the ZipPhone user algorithm. As described in the previous section, ZipPhone users renew their identifiers when: *(i)* they are in the process of switching towers, and *(ii)* the renewal *cool down period* (in seconds) has expired; *(iii)* they are not actively using the phone. To renew, users first detach, then stay offline, and then reattach with a new profile. The offline time is selected uniformly at random from a *maximum offline period*. It must be random, otherwise linking traces would be trivial. The cool down period ensures that the loss of utility remains at a minimum for the user. This aggressive

```
1: function PROFILE_USER($u$)                                   ▷ $u$ is the user index
2:     $T^u_{0,q} \leftarrow \frac{Count(q)}{\sum_{q' \in \mathbb{C}} Count(q')}$       ▷ The prior for user's initial location
3:     for all $p \rightarrow q \in$ TRANSITIONS($u$) do             ▷ $p \rightarrow q$ denotes a transition
4:         $T^u_{p,q} \leftarrow \frac{Count(p \rightarrow q)}{\sum_{q' \in \mathbb{C}} Count(p \rightarrow q')}$   ▷ This transition matrix may be sparse
5:     return $T^u$
6: function CLASSIFY_USER($\boldsymbol{s}$)  ▷ $\boldsymbol{s} = (s_0, s_1 \dots), s \in \mathbb{C}$ is a sequence of tower IDs
7:     return $\arg\max_u T^u_{0,s_0} \prod_{i=0}^{n-2} T^u_{s_i,s_{i+1}}$
```

Algorithm 4.2: Location profiling algorithm

renewal strategy is frequent enough to allow the natural formation of mix zones, and does not require users to coordinate times or places to mix.

4.3.3.2 Attacker strategies

The attacker's goal is to take a timestamped sequence of visited towers and infer the user, given a training set. We first describe a *location profiling classifier* that could be employed by the attacker. We then define a *trajectory linking classifier* to aid the attacker in trajectory linking.

4.3.3.2.1 Location profiling algorithm Our classifier (Algorithm 4.2) is a Markov model that chooses the most likely user for a sequence of tower attaches; the classifier is adapted from Mulder et al. [82]. With this classifier, the attacker labels a sequence of locations with the most likely user, based on all possible users' transition histories. In our model, vector $\boldsymbol{s}$ is a sequence of locations in the location set $\mathbb{C}$: $\boldsymbol{s} = (s_0, s_1, s_2 \dots), s \in \mathbb{C}$. In the steps below, the attacker identifies the most probable user given each candidate user's history, $\hat{u} = \arg\max_u p(u|\boldsymbol{s})$.

We determine the most likely user, given a sequence of locations.

$$\Pr(u|\boldsymbol{s}) = \Pr(u|s_0, s_1, s_2, \dots)$$

We apply Bayes' rule, and consider the likelihood of a sequence given a user.

$$\Pr(u|\boldsymbol{s}) = \frac{\Pr(s_0, s_1, s_2, \dots |u) \Pr(u)}{\Pr(s_0, s_1, s_2, \dots)}$$

We assume that each user is equally likely.

$$\begin{aligned}\Pr(u|\boldsymbol{s}) &\propto \Pr(s_0, s_1, s_2, \dots |u)\\ &= \Pr(s_0|u) \cdot \Pr(s_1|u, s_0) \cdot \Pr(s_2|u, s_0, s_1) \cdot \\ &\quad \Pr(s_3|u, s_0, s_1, s_2) \dots\end{aligned}$$

Each transition is independent per the Markov assumption.

$$= \Pr(s_0|u) \prod_{i=0}^{n} \Pr(s_{i+1}|s_i, u)$$

We determine the most likely user $\hat{u}$.

$$\hat{u} = \arg\max_u \Pr(s_0|u) \prod_{i=0}^{n} \Pr(s_{i+1}|s_i, u)$$

The attacker computes a transition matrix T for each user in the training data by counting the occurrences of these transitions in history. The probability of the first location in the sequence $\Pr(s_0|u)$ is computed from the overall number of a user's occurrence at a location. The attacker does not consider the probability of a trace ending at a certain location, since a sequence can end for arbitrary reasons.

The success of such an attack depends on two factors: the number of users in the anonymous community, and the similarity of the user's location transitions to the other users. If there is one registered cell phone user on the network, then linking the user to location is trivial; however, if there are many users who behave similarly, it would be difficult for the attacker to tell the user apart.

We also designed and tested a classifier that exploited diurnal features of user mobility, however, it did not perform significantly better than the above outlined algorithm. In our subsequent evaluations, the attacker does not employ diurnal features.

4.3.3.2.2 Trajectory linking algorithm In Algorithm 4.3, we extend Algorithm 4.2 to model the attacker's ability to do trajectory linking. The attacker uses the transitions of all users and builds a semi-Markov *linking transition matrix*. This matrix is similar to the one described in Algorithm 4.3, except that it is built by considering all subsequent

```
1: max_t ← Maximum time offline during renewal
2: function TRAIN_LINK_TRANSITIONS
3:     for all $p \xrightarrow{\text{max\_t}} q$ do                ▷ all locations $q$ seen within max_t of $p$
4:         $T^{l}_{p,q} \leftarrow \frac{Count(p \xrightarrow{\text{max\_t}} q)}{\sum_{q' \in \mathbb{C}} Count(p \xrightarrow{\text{max\_t}} q')}$    ▷ transition matrix used for linking
5:     return $T^{l}$
6: function CLASSIFY_USER_WITH_TRAJECTORY($\boldsymbol{s}$)
7:     while link_count<max_links do
8:         candidates ←FIND_CANDIDATES($\boldsymbol{s}$)    ▷ traces ≤ max_off_time after $\boldsymbol{s}$ ends
9:         if EMPTY(candidates) then
10:            break
11:        $\hat{\boldsymbol{s}'} \leftarrow \arg\max_{\boldsymbol{s}'} T^{l}_{\boldsymbol{s}_n, \boldsymbol{s}'_0}$    ▷ $\forall s' \in$ candidates
12:        $\boldsymbol{s} \leftarrow$ CONCATENATE($\boldsymbol{s}$,$\boldsymbol{s}'$)
13:    return CLASSIFY_USER($\boldsymbol{s}$)
```

Algorithm 4.3: Linking algorithm

locations within a given offline time, rather than only the next immediate location. This strategy ensures that unreasonable transitions do not confuse the classifier, and any unseen transitions occurring within that time frame are accounted for.

Our trajectory linking algorithm first searches for candidate traces that *start* within the maximum offline time. If a number of traces start within the offline time, the targets have a chance to mix, and the attacker must infer which trace comes next by using the semi-Markov transition matrix. This process is repeated until the trace is of sufficient length for classification, or there are no more candidates.

4.4 Evaluation

In this section, we determine the parameters in our model and evaluate the algorithms using two real-world datasets that contain geotagged user data coupled with tower attachment logs: PhoneLab [83] and RealityMining [42]. First, we characterize the amount of *predictability* and *mixing* behaviour exhibited by users in these datasets. We demonstrate that both characteristics are related to the success of the attacker's accuracy. Next, we simulate a deployment of ZipPhone amongst a community of users, and determine their reidentifiability with respect to sacrificed utility.

Type	Trait		Privacy	PhoneLab	Reality
	Predictable	Mixing	hypothesis		Mining
P/M	Yes	Yes	Moderate-Low	18%	18%
P/nM	Yes	No	Low	26%	30%
nP/M	No	Yes	High	30%	24%
nP/nM	No	No	Moderate	26%	29%

Figure 4.3: User typology and their proportions in our target datasets, with a hypothesis about the amount of privacy a user could attain from ZipPhone.

4.4.1 Datasets

Both datasets were collected primarily from university affiliates who carried phones instrumented with software to log phone network attachment and activity.

1. PhoneLab [83] is an Android testbed comprising 593 phones distributed to students at the University of Buffalo campus. As a part of this testbed, users contributed geotagged traces of their cellular network associations. We use 24 months from January 2015 to January 2017 of cellular network association traces from PhoneLab to assess the privacy preservation potential of ZipPhone.

2. RealityMining [42] is a dataset released by MIT that tracks a group of 100 mobile phone users across various contexts. Similar to PhoneLab, RealityMining contains geotagged network association information. For our analysis, we leverage 12 months of RealityMining data from July 2004 to July 2005.

We are unaware of other public datasets that could be used to analyze our algorithms. (We filed IRB protocol 2017-3900 as part of this project, and it was approved as exempt.)

4.4.2 Behaviour that affects attacker accuracy

We begin by characterizing user behaviour. Intuitively, there are two behavioural traits that affect mobile users' privacy: *(i) Predictability*, or to what extent users travel over fixed routes; and *(ii) Mixing* behaviour, or how likely are users to visit popular locations that see a large volume of other ZipPhone users. To highlight the effect of user behaviour on privacy, we categorized PhoneLab and RealityMining users *post hoc* into four groups:

- predictable (P) or unpredictable (nP); and
- mixing (M) or not mixing (nM).

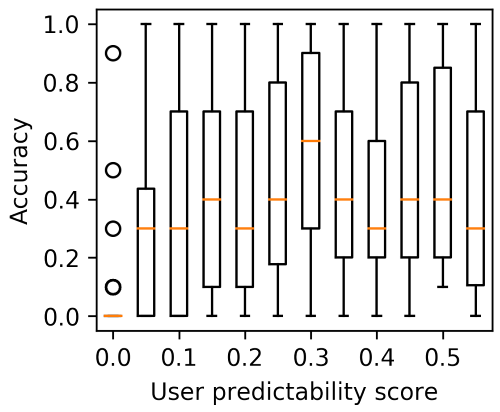
The four resulting user types are described in Figure 4.3, where we also set forth a hypothesis of how user behaviour would affect privacy. We verify and confirm these hypothesis in our evaluation (Section 4.4.3).

Predictability We calculate the user predictability in terms of the similarity of the set of cellphone towers they visited during the testing and training period. For each user, let C_{pre} be the set of towers visited during the training phase and C_{post} be the set of towers visited in the testing phase. We express the predictability in terms of a user's Jaccard similarity score between C_{pre} and C_{post}, defined as

$$J_C = \frac{C_{pre} \cap C_{post}}{C_{pre} \cup C_{post}}, \tag{4.1}$$

where $0 \leq J_C \leq 1$. $J_C = 0$ when the sets of visited towers in testing and training are completely disjoint, while $J_C = 1$ means that the sets of visited towers in testing and training are the same. Intuitively, a higher J_C means a more predictable trajectory.

Figure 4.4 (top) presents the attacker's accuracy (i.e., the probability that a user would be identified) as a function of the users' Jaccard score in the PhoneLab dataset. We note that the trends and respective thresholds are similar for the RealityMining dataset and omit these results due to space limitations. For this setup, 91% of users fall within the 0.0–0.4 Jaccard score range. For these users, we see an increasing trend in the attacker's accuracy as the Jaccard score grows. Using this analysis of our test dataset, we set the Jaccard score to 0.1 as a cut off to differentiate between predictable users (such with $J_C > 0.1$) and unpredictable users (such with $J_C \leq 0.1$).



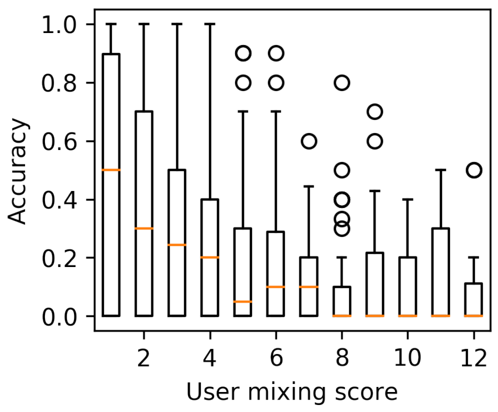


Figure 4.4: **Left:** User predictability versus attacker accuracy, showing that attacker accuracy is near zero with low predictability. **Right:** User mixing versus attacker accuracy, showing that median accuracy falls to zero as user mixing increases. The plots were computed from the PhoneLab dataset. The presented results are for a maximum offline time period of 30 seconds and a set utility of 95%. Utility and accuracy metrics are discussed in detail in Section 4.4.3.

Mixing behaviour We establish a mixing score $\mathcal{M}_\mathcal{C}$ as a metric that evaluates a user's likelihood to mix with other ZipPhone users. Intuitively, the higher the mixing score, the more efficient ID switching will be and the harder it will be for the adversary to evade a user's privacy. We calculate $\mathcal{M}_\mathcal{C}$ for each individual user. Let t_i^k be the duration of time a user $i \in (1, N)$ spends at tower $k \in (1, K)$. During the period t_i^k, other users $j \in (1, N'), j \neq i, N' \subset N$, may arrive and depart from tower k. Let τ_{ij}^k be the time of user j's arrival or departure. Intuitively, t_i^k and τ_{ij}^k define the temporal granularity of tower mobility and ZipPhone user encounter events, respectively, from the perspective of a single user i. Let $C(\tau_{ij}^k)$ be the number of users in user i's vicinity at time τ_{ij}^k. We define the mixing score as:

$$\mathcal{M}_\mathcal{C} = \sum_{k=1}^{K} \sum_{j=1}^{N'} \frac{C(\tau_{ij}^k)}{\tau_{ij}^k - \tau_{i(j-1)}^k} \tag{4.2}$$

Figure 4.4 (bottom) presents the attacker's accuracy as a function of the users' mixing score in the PhoneLab dataset. The trends and respective thresholds are similar for the RealityMining dataset. We see that the attacker's accuracy deteriorates as the users' mixing score increases. Based on this analysis, we set a mixing score of 4 as the cutoff to determine whether a user is mixing or not mixing, i.e. users with $\mathcal{M}_{\mathcal{C}} \leq 4$ are not mixing and these with $\mathcal{M}_{\mathcal{C}} > 4$ are mixing.

User typology in our datasets. As detailed earlier, we differentiate between four types of users based on their predictability and mixing behaviour. Using the presented analysis in Figure 4.4, we set a Jaccard similarity threshold of 0.1 and mixing score threshold of 4. We note that these thresholds are solely used to establish the user topology in the following evaluation and do not play a role in the profile classification carried out by the attacker. Figure 4.3 presents the amount of users that fall in each user type category. We see a relatively even user representation across all categories. We use these user types and the corresponding user populations in all results presented in the evaluation of ZipPhone (Section 4.4.3).

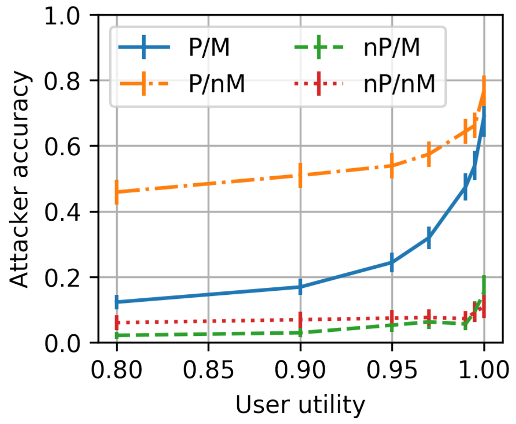
4.4.3 Results

To determine the affect of ZipPhone on the utility and privacy of users, we simulated the protocol using the PhoneLab and RealityMining datasets. In these simulations, the attacker uses the inference algorithms outlined in Section 4.3.3.2 to develop a location profile for each user based on two months of training data, and labels anonymous traces in the next month.

4.4.3.1 Utility-privacy trade-off

We evaluated the utility-privacy tension with regard to the four user types. We quantify privacy gained in terms of reduced attacker accuracy. We measured loss of utility in terms of time spent offline during the testing period. Figure 4.5 displays the privacy gained by each user group during the one-month testing periods.

Users gained significant privacy from sacrificing 5% utility, on average remaining online for 9.5 minutes, and going offline for 30 seconds. In particular, Type P/M (predictable but mixing users) gained 45% in the PhoneLab dataset, and 49% in the RealityMining dataset. Interestingly, Types nP/M and P/nM also show a similar trend: Type nP/M ben-



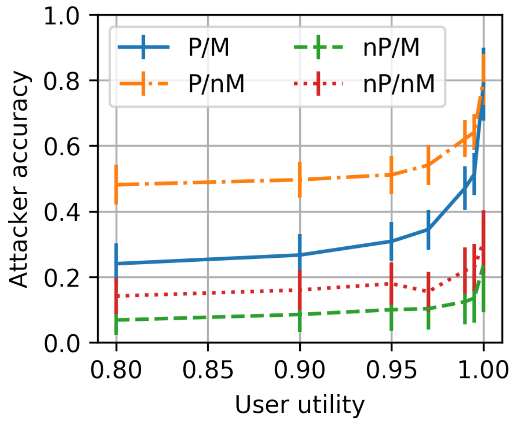


Figure 4.5: **Left**: PhoneLab. **Right**: RealityMining. In both datasets, predictable but mixing users (Type P/M) gain the most from using ZipPhone. Ten test traces were evaluated per user, and accuracy is represented as a mean of the proportion of successful reidentifications per user. Error bars represent a 95% confidence interval.

efits from having the divided traces be less predictable, and for Type P/nM any small amount of predictability is reduced to none. Type nP/nM does not mix, and enjoys uniformly high privacy because they are unpredictable. Users were more private in general in the PhoneLab, since it represented a larger community of users, making mixing easier for the user, and user inference more difficult for the attacker.

4.4.3.2 Trace length and location profiling

The main driver of attacker accuracy is trace length. In these experiments, the attacker tries to identify an independent trace of varying length. Figure 4.6 shows the result.

In general, the longer the trace, the more identifiable (and thus less private) an individual is. Users who exhibit more predictable behaviour have less privacy; generally, they benefit from traces that are at most one hour long. In other words, predictable users should change their identifier at least once per hour while in motion. Those who travel to unique locations as compared to others benefit significantly less from the shorter trace.

This result highlights the benefit of ZipPhone. Users should change their identifiers more than once per hour, and this system obviates the need to physically change an identifier, and handles this process automatically. While a temporary SIM device may grant

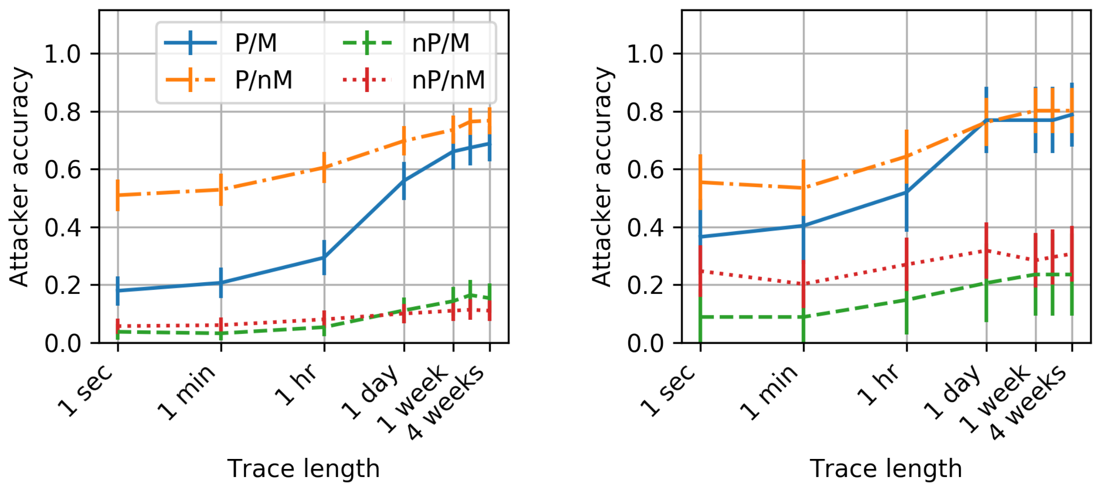


Figure 4.6: **Left**: PhoneLab. **Right**: RealityMining. Users lose a significant amount of privacy when traces are on the order of one day long. The accuracy at one month is equivalent to the accuracy in Figure 4.5 at 100% utility.

some measure of privacy, a system that renews a user's identifier a lot more quickly can be a lot more effective.

4.4.3.3 Compromises in utility

While users may renew identifiers by prearranging mixing strategies with other users, such coordination is impractical. A frequent enough renewal strategy and long enough renewal times allow mix-zones to naturally form, which enables users to mix without any coordination. In Figure 4.7 (top), we examine the amount of time a user should remain offline. The frequency of renewal is informed by the utility desired, which we set at 95%.

For users to gain privacy during identifier renewal, they must remain offline long enough to mix with other users. Additionally, users must not have a fixed offline time, since this would be susceptible to a timing attack. Users must choose an offline time that is not so long to be disruptive, but not so short as to offer little privacy. The ZipPhone population's policy should fix a chosen utility, and employ a cool down time between each user's identifier renewal based on that desired utility. For example, if users' offline-times are 30 seconds, and are aiming to maintain 95% utility, they will keep every identity for at least 30 seconds $\div (1 - 0.95) =$10 minutes.

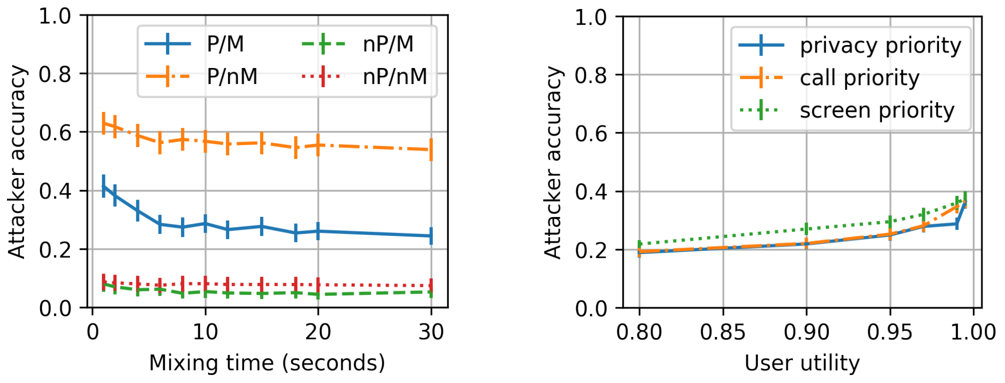


Figure 4.7: **Left**: the effect of mixing-time on privacy while maintaining a 95% utility for the PhoneLab dataset. **Right**: privacy/utility of all users depending on whether their priority is privacy, phone calls, or screen use. Calls can be prioritized without sacrificing privacy. However, remaining online while the screen is on significantly reduces privacy.

Because going offline for 30 seconds can be fairly disruptive, we analyzed scenarios where reconnections are disallowed if *(i)* the user is in the middle of a phone call, or *(ii)* the device screen is active. This data was available in only the PhoneLab dataset. Since phone calls were intermittent, active calls could be kept online without sacrificing privacy. However, within the offline periods, users would on average miss 4 calls out of 24 per month while maintaining 95% utility. Looking at screen usage, we show in Figure 4.7 (bottom) that users could preserve active usage of phone undisturbed, but in doing so would sacrifice additional privacy by a small amount (i.e. about 2% across all utility levels).

4.5 Integrating ZipPhone with emerging mobile networks

In this section we discuss how ZipPhone could be integrated in emerging mobile cellular networks towards improved user privacy. We first present necessary background on user authentication in emerging cellular networks. We then detail how ZipPhone can utilize these networks for privacy-preserving services without requiring network modifications. Finally, we present empirical results for user-side energy overhead.

4.5.1 Background

Traditionally, hardware SIM cards installed in mobile devices provide the basis for user provisioning in Mobile Network Operators (MNO). Each SIM has a unique International Mobile Subscriber Identity (IMSI). At the MNO's Home Location Registry (HLR) an entry is created connecting the IMSI with an Mobile Station International Subscriber Directory Number (MSISDN; i.e., a phone number). At the Authentication Center (AuC) the IMSI is paired with a K_i value and used for user authentication.

This approach is plagued with high overhead, wasted allocations, and manual processes. To address these limitations, the eSIMs standard [15] has been developed, which allows programmatic and on-the-fly provisioning of a user's identity on a network. With eSIMs, mobile users can maintain multiple simultaneous mobile network identities and use heterogeneous services from one or multiple MNOs. Three out of the four major carriers in the US currently support eSIM, with one major carrier supporting eSIM in 42 other countries worldwide [16].

eSIMs introduce new components to user management that are useful for ZipPhone. Similar to traditional SIMs, the eSIM functional *profile* [3] carries phone identification information and is jointly maintained in the MNO's HLR and the AuC. The Subscription Manager Data Preparation (SM-DP+), is responsible for provisioning a user's profile onto the eSIM. Thus, the SM-DP+ is the first point of communication between an aspiring subscriber and the MNO, from which the subscriber obtains their functional profile. There is no upper limit on the amount of *profiles* an eSIM can maintain; this number depends on *(i)* the size of a single *profile*, *(ii)* the eSIM integrated memory and, *(iii)* the operator's preferences. As an example, T-Mobile currently supports up to 10 concurrent eSIM *Profiles* [2].

4.5.2 Proposed ZipPhone Architecture

4.5.2.1 Overview

ZipPhone can be realized as a smartphone application. Upon installation and then periodically, the ZipPhone app will anonymously acquire multiple functional profiles and associated *service quants* from the MNO's SM-DP+. We define a service quant as a set of mobile services (i.e. data, text messages and voice calls) that the subscriber will use while active with the particular profile and note that these quants can be obtained in the form of an anonymous prepaid service [17, 18]. ZipPhone then programmatically swaps these profiles as discussed in Section 4.3 and uses the corresponding service quant for the duration in which a profile is active. This functionality can be achieved without explicit cooperation from the network provider or any modifications in the network as long as the network provider is eSIM-capable and offers anonymous prepaid plans.

4.5.2.2 Purchasing Credentials

ZipPhone requires that users anonymously purchase *profiles* without linking to a consistent financial or network identifier. This purchase would be a significant challenge to deploying ZipPhone as it must also not be used to profile the user. Here we offer a sketch of how it could be done.

Zcash [61,96] offers a basis for such purchases as follows. The MNO advertises a Zcash address for purchasing credentials and a price per purchased profile. The user creates a public-private key pair, and issues a transaction that transfers the purchase amount to the MNO, and includes its own public key as part of the transaction (which could be encrypted with the public key of MNO's address for greater security). The MNO responds with a data only transaction that includes the set of credentials, encrypted with the provided public key. This exchange must occur separately for each credential purchased, each with a different public key and each from a different Zcash source address, so that a series of purchases cannot be linked as belonging to the same user. These requirements are easy to provide with Zcash and can be performed programmatically. There is an effort to deploy Zcash's zero-knowledge mechanisms on Ethereum as well.

The user's transactions should not be overtly issued from the same IP address, as that would allow linking of the purchases. Protocols such as Dandelion++ [47] allow transactions to be issued to Zcash with network anonymity. A ZipPhone users would carry a series of pre-purchased profiles, and use them in a random order.

We note that this entire scenario could be handled by a third-party Mobile Virtual Network Operator (MVNO) that resells for the MNO. In that case, the MVNO should issue signaling to the MNO to cancel the IMSI a period of time after they are first used (e.g., 15–30 minutes).

4.5.2.3 Communication without Leaking Identity or Location

For an additional layer of privacy, ZipPhone users should ignore the MSISDN (phone numbers) provided by a profile. In other words, users should not use MSISDN-based services such as text and voice calls and instead should rely on IP based services over the data plan. If a ZipPhone user initiated or received overt LTE or unencrypted VoIP calls, they risk being identified via a profile of call records held by the carrier. Note that the E911 service, which is tied to a handset and not a user or SIM, would be still available if needed.

Some protection would be gained from using an encrypted VoIP service, since it would not reveal to the carrier the identity of the user's contact, whom she calls, or from whom she receives calls. However, if the IP address of the VoIP service is unique, then connecting to it would help the MNO link a collection of profiles together. An anonymous VoIP service, such as Torfone can be used; note that anonymous VoIP has a performance penalty [78].

In general, an anonymous communication systems, i.e., Tor, must be used for all ZipPhone communication (voice or data). However, there is one change required. Tor chooses

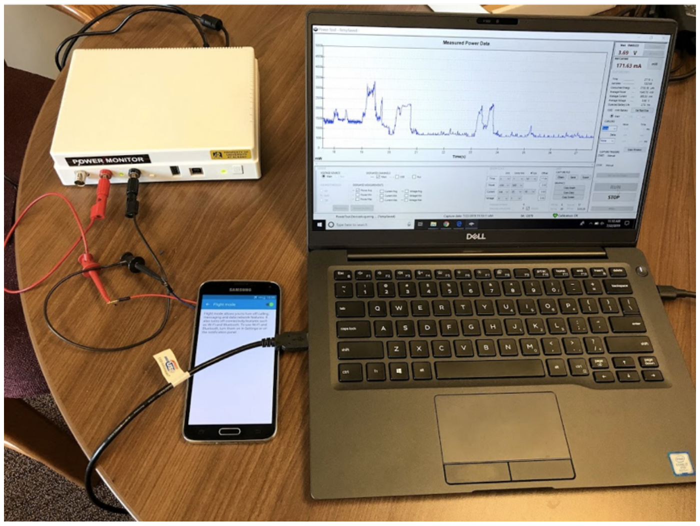


Figure 4.8: Experimental setup for power measurements on 3G and 4G networks.

a consistent, single *guard* relay to start all three-relay circuits through the Tor network. If ZipPhone users send all traffic to a single guard relays, it would be a consistent identifier despite changing IMSIs. Instead of a guard at the start of the circuit, ZipPhone users should use a consistent relay as the exit. This switching of roles allows ZipPhone users to receive all protections against the Predecessor Attack [113] that Tor normally provides via guard nodes at the entry.

4.5.3 ZipPhone Overhead

ZipPhone triggers periodic disassociation/association from the mobile carrier, which together incur additional battery draw and connect/disconnect delays on the mobile device. Thus, in this section, we quantify the overhead in terms of battery drain and latency, incurred by ZipPhone on 3G and 4G networks.

Experimental setup. In order to evaluate the power consumption of mobile network association/disassociation, we used a Samsung Galaxy S5 Duos phone with a bypassed battery and a Google Fi SIM card, and a Monsoon Power Meter. We connected the phone to the main channel of the power meter, as illustrated in Figure 4.8, which allowed us to both power up the phone and measure its energy consumption. In order to measure the

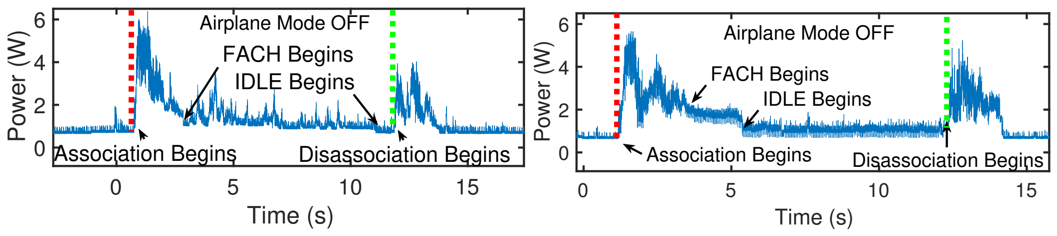


Figure 4.9: Power trace for 3G (left) and 4G (right) association and disassociation.

power draw at 3G and 4G networks, we forced the phone to the respective technology and sampled the power draw at a granularty of 200μs. We used the phone's Settings screen to toggle between Airplane Mode OFF and Airplane Mode ON every 10 seconds for 4G and every 20 seconds for 3G. We disabled all background services on the phone. This ensured that we are only measuring the power draw from association/disassociation, plus a baseline of about 700mW used by the display for the Airplane Settings page. For each of 3G and 4G we completed 10 full association/disassociation cycles. The average experienced time and power to connect inform our simulaiton results.

Figure 4.9 presents a zoomed version of a single associate/disassociate cycle for 3G (top) and 4G (bottom)[1]. There are several important points to note on each trace. First, the red vertical line indicates the phone's transition from Airplane Mode ON to OFF state, which immediately triggers a network association. After the association procedure completes, the phone enters FACH (Forward Access CHannel) state in anticipation for the user to begin accessing the Internet. Since this does not happen in our controlled activity, the phone futher transitions into IDLE state. At the instant designated with a green vertical line, we toggle Airplane Mode ON, which immediately triggers a disassociation procedure.

A ZipPhone user would experience two types of overhead: *(i)* offline time, and *(ii)* power draw. We measure the offline time as the time between the beginning of network associa-

[1]Note that the timescale (i.e. the x-axis range) for 3G is longer than that for 4G, because on 3G the phone takes significantly longer to transition to IDLE mode compared to 4G.

		mean	**(std dev)**
3G	Power to connect (mW)	2,098	(435)
	Power to disconnect (mW)	1,282	(157)
	Time to connect (s)	5.0	(0.8)
	Time to disconnect (s)	4.0	(1.0)
4G	Power to connect (mW)	2,006	(171)
	Power to disconnect (mW)	1,120	(295)
	Time to connect (s)	2.6	(0.2)
	Time to disconnect (s)	3.0	(1.2)

Figure 4.10: Time and power overhead incurred by a single association/disassociation procedure on 3G and 4G in our experiments. Results are averaged over 10 runs.

tion and the beginning of the FACH state. We measure the power overhead as the sum of power to associate and power to disassociate, whereby the power to associate is incurred from the begining of the network association to the beginning of the FACH state, while the power to disassociate is measured from the beginning till the end of the disassociation procedure.

Figure 4.10 presents the average incurred overhead for our measurement campaign. We see that the offline time incurred by 3G is nearly double that of 4G. The power consumption, on another hand, is comparable across the two technologies. We use these results to quantify the battery usage per day for users in our datasets. To this end, we convert the measured power consumption for a single connect/disconnect from mW to mWh using the values in Figure 4.10. We assume a 3.85V battery with a capacity of 2800mAh, which is typical. On the x-axis we control the desired user utility from 0.8 to 1, which effectively controls the amount of network disconnect/connect cycles a user will incur for the duration of a day. We multiply that number by the energy consumption (in mWh) and then divide by the battery's capacity to determine what fraction of the battery is consumed due to ZipPhone. Figure 4.11 presents our results, which indicate that the battery usage is at most 1% per day regardless of technology (3G or 4G) or desired privacy.

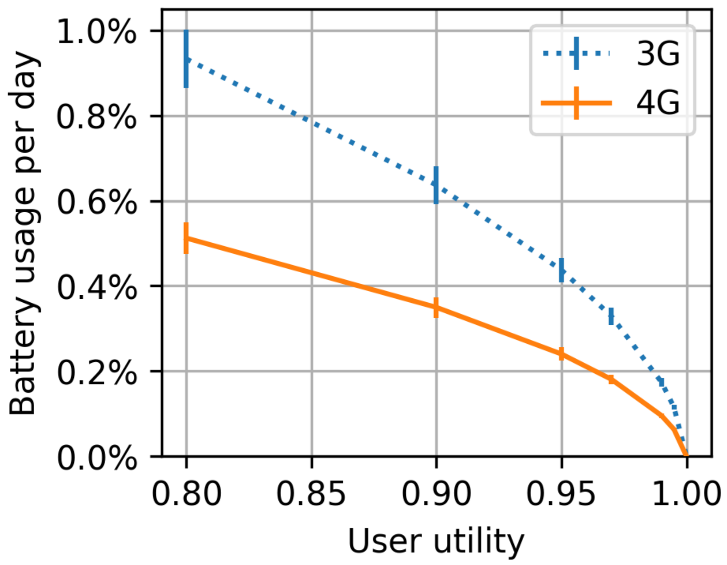


Figure 4.11: Battery usage does not exceed 1% per day, regardless of desired privacy or network type.

Network control overhead. Finally, although we do not explicitly quantify it, we do not expect that ZipPhone users would incur significantly higher signalling overhead on the cellular network compared to non-ZipPhone users. In order to release network resources and optimize clients' battery life, network providers forcefully disassociate users from the network after a network-defined timeout [92], typically in the order of a few seconds as illustrated by our measurements in Figure 4.9. Since ZipPhone only operates when a user is inactive, the control overhead incurred by the network will be comparable with that from non-ZipPhone users.

4.6 Discussion

4.6.1 Limitations

Our technique has limitations. We require devices that accept software SIMs; these are not common in the marketplace now, though easy to provide. Another limitation is that users would never be able to quantify their privacy gains as there is no way to determine the number of other ZipPhone users. In addition, we do not address other privacy risks, which include physical attacks (e.g., radio frequency fingerprinting [39]), software vulnerabilities, use of location-based services, advertising fingerprints, browser cookies, and malware.

Our evaluations are limited as well. For example, we do not explicitly consider users mixing when they are stationary; if they do, attackers could also consider these additional mixes when linking. A more advanced attacker's classifier might account for yet additional features (e.g., time of day or favourite locations [120]) to increase accuracy. Conversely, users could develop more efficacious methods to prevent linking. Finally, our results are tied to our datasets, which are relatively small and limited to populations from universities. Obtaining a usable large-scale dataset is difficult, as MNOs are generally unwilling to anonymize and share such data. Furthermore, collecting user mobility data first-hand requires a fairly involved longitudinal effort.

Despite these limitations, this study introduced an effective method for mobile network users to take charge of their own location privacy, and provided a detailed look at the efficacy of such a service.

4.6.2 Ethical implications

Mobile devices are an essential part of most people's daily routine. Accordingly, there is a tension between the right to location privacy and the need to investigate crimes and threats to public safety. The techniques we introduce and evaluate are effective to protecting privacy, but unfortunately would thwart a common method of investigation as well. Any deployment of ZipPhone would have to take into account this difficult, zero-sum game ethical dilemma.

4.7 Related Work

Our study is related to a broader category of prior work on location privacy. Most prior work assumes the service provider is trusted and in fact responsible for user privacy. Prior approaches have a variety of goals, including: *(i)* properly anonymizing mobility datasets before public release; *(ii)* adding privacy for users of locations based services; and *(iii)* increasing location privacy for mobile device users from third-party attackers but *not* the service provider itself. In contrast to these works, our goal is to provide mobile users location privacy from the wireless provider itself. This presents a unique challenge: the user is responsible for her own privacy, and the only control she has over this is whether to remain connected to the service at any moment in time.

Location privacy with provider cooperation. Many studies focus on enlisting a trusted carrier to protect against a third party attacker [50, 55, 56, 76]. Reed et al. [95] propose privacy from the carrier using onion routing, but does not consider the direct connection that must be made to a tower. Federrath et al. [49] propose a similar scheme that prevents linkability of calls between two parties but omit critical details regarding authentication to the carrier. Fatemi et al. [48] propose an anonymous scheme for UMTS using identity-based encryption, but unlike our approach, their scheme involves the carrier in the cryptographic exchange; they enumerate the vulnerabilities of similar works [66,89,118,121]. Kesdogan et al. [70] proposes using a trusted third party to create pseudonyms for GSM users, but also routes all calls through that provider, which allows it to characterize the calling pattern and infer the caller.

User-driven trajectory privacy. Mix zones [25,52] can be employed by a user against a provider attacker, in cases where the network service provider is non-cooperative. While the concept of mix zones is fairly old, it remains the only available option for users who want to hide their own location privacy from a service provider. Work in VANETs also uses mix zones to protect vehicle trajectory [46, 58, 79]. Given that the focus is on trajectory, these studies do not consider location profiling. Other work involves the introduction of false information [72, 98]. Few studies use this concept to protect the user from an omnipresent network attacker. Chan [30] focuses on call metadata privacy, rather than location privacy.

User-driven profiling privacy. Work that increases the privacy of location-based services (LBS) [62, 85, 110, 111] generally add noise to location queries. These works are not viable or applicable against an untrusted service provider: a user cannot manipulate which tower they connect to, and the provider knows the physical locations of the towers serving users.

Dataset protection. Works that aim to prevent leaks in personally identifiable information in shared or publicly released datasets [119] primarily rely on obfuscation. These works also include efforts to prevent trajectory recovery [57, 106]. Older work on deanonymization of private traces of mobile users assume the user's pseudonym is unchanged throughout the trace. But a small amount of external information, such as the person's home or work address [65], can deanonymize an obfuscated trace [24, 25, 53, 75, 80, 82] given a consistent identifier. Zang and Bolot [120] show that suitably anonymizing a trace of 25 million cellular users across 50 states (30 billion records total) requires only that users have the same pseudonym for no longer than a day. A day's duration is unsuitable for Zang and Bolot's goal of supporting researchers that wish to characterize the behaviour of users over time (while maintaining their privacy). On the other hand, the result is promising for users seeking privacy, who might be able to change their pseudonyms much more frequently than once per day.

Differential privacy. More recently, differential privacy approaches [41, 81] are used to add noise to datasets while preserving its aggregate characteristics. Palamidessi et al. [19] introduce geo-indistinguishability, and ElSalmouny & Gambs [45] further discuss (D, ϵ)-location privacy. Xiong et al. [114, 115] formalize situations where location queries can be temporally correlated and linked. These methods all assume the service provider is trusted and are, thus, not applicable to our problem setting.

Outside threats. Several studies protect against third party attackers and vulnerabilities in 3GPP implementations [60, 63]. Khan et al. [71] provide a cryptographic mechanism to generate LTE pseudonyms and prevent third-party attackers or IMSI catchers from linking users.

In comparison to related work, we differ in that we do not trust the wireless service to ensure the user's privacy, and we assume in our analysis that the adversary is attempting to link together traces. Our evaluations are based on traces of real users [42, 83], which allows us to quantify the periodicity of identifier changes in the context of modern cellular infrastructure.

4.8 Concluding remarks

Our work demonstrates that, fundamentally, users do not need to trust wireless service providers with their location information. We evaluated a deanonymization attack that uses a combination of location profiling and trajectory linking, and showed that it is effective in identifying long-term pseudonyms. Using two separate datasets of call detail records, we then demonstrated that a ZipPhone user can defend against such attacks by renewing her identifier regularly. We also evaluated the utility cost in terms of time offline and battery life, and showed it to be minimal. Users who do not use any anonymization scheme are always identifiable. In our trace-driven evaluations, a non-ZipPhone user who is habitual and conventional (predictable and mixing) who renews her pseudonym monthly is identifiable 69% of the time, and one who uses ZipPhone is identifiable 24% of the time if she sacrifices 5% of her utility and 1% of battery life, towards a lower bound of 19% if she sacrifices more. In other words, users can significantly reduce their identifiability by up to 45% by renewing their pseudonym after offline periods consuming less than 5% of their uptime.

CHAPTER 5

SUMMARY

The principles of, and rules on the protection of natural persons with regard to the processing of their personal data should, whatever their nationality or residence, respect their fundamental rights and freedoms, in particular their right to the protection of personal data. This Regulation is intended to contribute to the accomplishment of an area of freedom, security and justice and of an economic union, to economic and social progress, to the strengthening and the convergence of the economies within the internal market, and to the well-being of natural persons.

Recital 2, *General Data Protection Regulation*, 2016

These last two decades have seen a rapid progression in mobile technology, and along with it a slew of privacy issues. In this thesis, I looked at the privacy issues inherent in persistent cellular connectivity, with regard to each data stakeholder that we are expected to trust. My collaborators and I found a vulnerability in an industry-standard advertising framework that allows advertisers to continue tracking users that have enabled Limit Ad Tracking. We determined that cellular connection quality is enough for a remote attacker to narrow down the location of a user to a path travelled. We established that fundamentally, to be able to privately use a cellular network without revealing location to a cellular provider, a user must change their SIM information, and in doing so, sacrifice some amount of utility.

But I argue that these issues are merely growing pains. Policy is catching up to regulate private data usage. Aggressive study of privacy issues is paramount to ensuring personal data safety, and quelling technophobia. The GDPR already defines broad policies to protect data, for example, by requiring explicit consent for services to use cookies and persistent information storage to track users, and by requiring explicit consent for users to have their data processed. Enforcement and more specific legal clarification of these policies are another story, but this is progress in developing comprehensive privacy rights. While the GDPR protects citizens of the European Union, other jurisdictions have followed suit, and hopefully many more will also do so in the future.

In the end, the reason we research privacy is to inform users on how to protect information that is personal and intimate to them, and to inform policymakers, service providers, and developers on how better to protect the rights and freedoms to privacy. A combination of good policy and consumer vigilance is needed to protect personal privacy.

5.1 Recommended policies

1. When storage of user data is regulated, it is usually in the context of data being stored on a *server*. As we have discovered in Chapter 2, special consideration must also be given to information that is stored on the client *device*.

2. Network access traces can justifiably be logged for quality control purposes. These traces, even if they are solely metadata and contentless, should be treated like personal data, as they may help reveal a user's location (Chapters 3 and 4).

3. Make personally collected personal data transparent to the user, and easily accessible (i.e. Right of Access, or Subject Access Rights). This gives users an opportunity to study the potential of their data, and even sell it to researchers. Not only does this give users the knowledge of what is actually being collected, but it could also make it much easier to do privacy research.

5.2 Recommendations for privacy-preserving behaviour

These are recommendations — based on results from this thesis — to protect your identity, location, and trajectory from any attacker, while using a mobile device.

1. Use a device that is unlikely to have been compromised by a malicious party. All studies carried out in this thesis assumed that the user's device was trustworthy.

Totally ensuring the trustworthiness of a device is nearly impossible, but popular devices are well studied by security researchers, and tested by the public.

2. Untrusted apps should only be allowed to connect to the network or be given geolocation access if necessary. If temporary network usage is needed, then allow network access only temporarily. Question unnecessary permissions requests.

3. Revealing your location, even anonymously, is always against your privacy interests. Do so conservatively.

4. A device that you anonymously use long-term is likely to reveal your identity. You should consider such a device no longer anonymous after one day, especially if you bring it to areas you frequent.

5. To avoid marketing tracking, disable or reset the advertising identifier on your device regularly. If resetting, clear the data and cache on apps that contain advertising at the same time. In the same vein, clear browser cache and web storage regularly on any device that you own, and use an ad blocker.

BIBLIOGRAPHY

[1] Advertising ID. https://support.google.com/googleplay/android-developer/answer/6048248.

[2] eSIM settings: Apple iPhone on iOS 12. https://support.t-mobile.com/docs/DOC-39253.

[3] eUICC Technical Releases. https://simalliance.org/euicc/euicc-technical-releases/.

[4] Gridded Population of the World, Version 4 (GPWv4): Population Count Grid. http://dx.doi.org/10.7927/H4X63JVC. Accessed: 2019-09-19.

[5] Scott v. AT&T. https://www.eff.org/files/2019/07/16/2019-07-16_-_001_-_complaint.pdf.

[6] Silent Circle blackphone. http://silentcircle.com.

[7] Torfone. http://torfone.org.

[8] https://developer.android.com/reference/android/webkit/.

[9] https://developer.chrome.com/apps/offline_storage. Accessed: 2020-03-24.

[10] https://stackoverflow.com/questions/8500334/how-to-remove-html5-local-storage-of-an-ios-app-using-uiwebview.

[11] https://support.appsflyer.com/hc/en-us/articles/115003734626-FAQ-Impact-of-Apple-Limit-Ad-Tracking-on-attribution.

[12] https://www.ip2location.com/. Accessed: 2020-03-24.

[13] https://www.maxmind.com/. Accessed: 2020-03-24.

[14] https://www.verve.com/limit-ad-tracking/.

[15] eSIM Whitepaper: The what and how of Remote SIM Provisioning. https://www.gsma.com/esim/wp-content/uploads/2018/12/esim-whitepaper.pdf, March 2018.

[16] Find wireless carriers that offer eSIM service. https://support.apple.com/en-us/HT209096, July 2019.

[17] Boost Mobile Prepaid Plans. https://www.boostmobile.com/plans, March 2020.

[18] T-Mobile SimplyPrepaid Plans. https://prepaid.t-mobile.com/prepaid-plans, March 2020.

[19] Andrés, Miguel E, Bordenabe, Nicolás E, Chatzikokolakis, Konstantinos, and Palamidessi, Catuscia. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security* (2013), ACM, pp. 901–914.

[20] Auxier, Brooke, Raine, Lee, Anderson, Monica, Perrin, Andrew, Kumar, Madhu, and Turner, Erica. Americans and privacy: Concerned, confused and feeling lack of control over their personal information, 2019.

[21] Balakrishnan, Mahesh, Mohomed, Iqbal, and Ramasubramanian, Venugopalan. Where's that phone? Geolocating IP addresses on 3G networks. In *ACM IMC* (2009), pp. 294–300.

[22] Bashir, Muhammad Ahmad, Arshad, Sajjad, Robertson, William, and Wilson, Christo. Tracing information flows between ad exchanges using retargeted ads. In *25th USENIX Security Symposium (USENIX Security 16)* (2016), pp. 481–496.

[23] Becker, Richard A., Caceres, Ramon, Hanson, Karrie, Loh, Ji Meng, Urbanek, Simon, Varshavsky, Alexander, and Volinsky, Chris. Route classification using cellular handoff patterns. In *ACM UbiComp* (2011), pp. 123–132.

[24] Beresford, A.R., and Stajano, F. Location privacy in pervasive computing. *IEEE Pervasive Computing 2*, 1 (2003), 46–55.

[25] Beresford, A.R., and Stajano, F. Mix zones: user privacy in location-aware services. In *Proc. Pervasive Computing and Communications Wrkshps* (2004), pp. 127–131.

[26] Berndt, Donald J, and Clifford, James. Using dynamic time warping to find patterns in time series. In *KDD workshop* (1994), vol. 10, Seattle, WA, pp. 359–370.

[27] Cao, Yinzhi, Li, Song, Wijmans, Erik, et al. (cross-) browser fingerprinting via os and hardware level features. In *NDSS* (2017).

[28] Case No. 19-cv-4063. Scott, Jewel, And Pontis, et al. v. AT&T Inc.; AT&T Services, Inc.; AT&T Mobility, LLC; Technocom Corp.; and Zumigo, Inc. https://www.courthousenews.com/wp-content/uploads/2019/07/ATTlocationservices-COMPLAINT.pdf, July 2019.

[29] Chan, Mun Choon, and Ramjee, Ramachandran. TCP/IP performance over 3G wireless links with rate and delay variation. In *ACM MobiCom* (2002), pp. 71–82.

[30] Chan-Tin, Eric. Anoncall: Making anonymous cellular phone calls. In *2015 10th International Conference on Availability, Reliability and Security* (2015), IEEE, pp. 626–631.

[31] Chesterfield, Julian, Chakravorty, Rajiv, Crowcroft, Jon, Rodriguez, Pablo, and Banerjee, Suman. Experiences with multimedia streaming over 2.5g and 3g networks. In *of IEEE Workshop on Broadband Wireless Multimedia (BroadWiM)* (Oct. 2004).

[32] Corner, Mark D, and Levine, Brian N. Micromobile: Leveraging mobile advertising for large-scale experimentation. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services* (2018), ACM, pp. 310–322.

[33] Corner, Mark D, Levine, Brian N, Ismail, Omar, and Upreti, Angela. Advertising-based measurement: A platform of 7 billion mobile devices. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking* (2017), ACM, pp. 435–447.

[34] Corner, Mark D., Levine, Brian Neil, Ismail, Omar, and Upreti, Angela. Advertising-based Measurement: A Platform of 7 Billion Mobile Devices. In *ACM International Conference on Mobile Computing and Networking (MobiCom)* (October 2017).

[35] Dabrowski, Adrian, Pianta, Nicola, Klepp, Thomas, Mulazzani, Martin, and Weippl, Edgar. IMSI-Catch Me If You Can: IMSI-Catcher-Catchers. In *Proc. ACM ACSAC* (2014).

[36] Das, Anupam, Borisov, Nikita, and Caesar, Matthew. Tracking mobile web users through motion sensors: Attacks and defenses. In *NDSS* (2016).

[37] Day, John D, and Zimmermann, Hubert. The osi reference model. *Proceedings of the IEEE 71*, 12 (1983), 1334–1340.

[38] De Montjoye, Yves-Alexandre, Hidalgo, César A, Verleysen, Michel, and Blondel, Vincent D. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports 3* (2013), 1376.

[39] Deng, Shouyun, Huang, Zhitao, Wang, Xiang, and Huang, Guangquan. Radio frequency fingerprint extraction based on multidimension permutation entropy. *International Journal of Antennas and Propagation 2017* (2017).

[40] Dingledine, R., Mathewson, N., and Syverson, P. Tor: The second-generation onion router. In *USENIX Security* (2004).

[41] Dwork, Cynthia. Differential privacy. *Encyclopedia of Cryptography and Security* (2011), 338–340.

[42] Eagle, N., and Pentland, A. Reality Mining: Sensing Complex Social Systems. *Personal and Ubiquitous Computing 10*, 4 (2006), 255–268.

[43] Eckersley, Peter. How unique is your web browser? In *International Symposium on Privacy Enhancing Technologies Symposium* (2010), Springer, pp. 1–18.

[44] Eddy, Sean R. Hidden markov models. *Current opinion in structural biology 6*, 3 (1996), 361–365.

[45] ElSalamouny, Ehab, and Gambs, Sébastien. Differential privacy models for location-based services.

[46] Emara, Karim, Woerndl, Wolfgang, and Schlichter, Johann. Caps: Context-aware privacy scheme for vanet safety applications. In *Proceedings of the 8th ACM conference on security & privacy in wireless and mobile networks* (2015), ACM, p. 21.

[47] Fanti, Giulia, Venkatakrishnan, Shaileshh Bojja, Bakshi, Surya, Denby, Bradley, Bhargava, Shruti, Miller, Andrew, and Viswanath, Pramod. Dandelion++: Lightweight cryptocurrency networking with formal anonymity guarantees. *Proc. ACM Meas. Anal. Comput. Syst. 2*, 2 (June 2018), 29:1–29:35.

[48] Fatemi, M., Salimi, S., and Salahi, A. Anonymous roaming in universal mobile telecommunication system mobile networks. *IET Information Security Journal 4*, 2 (2010), 93–103.

[49] Federrath, Hannes, Jerichow, Anja, Kesdogan, Dogan, and Pfitzmann, Andreas. Security in Public Mobile Communication Networks. In *Proc. IFIP/TC6 Personal Wireless Communications* (April 1995), pp. 105–116.

[50] Federrath, Hannes, Jerichow, Anja, and Pfitzmann, Andreas. MIXes in Mobile Communication Systems: Location Management with Privacy. In *Proc. Intl. Wrkshp on Information Hiding* (1996), pp. 121–135.

[51] Ferris, Brian, Fox, Dieter, and Lawrence, Neil D. Wifi-slam using gaussian process latent variable models. In *IJCAI* (2007), vol. 7, pp. 2480–2485.

[52] Freudiger, Julien, Shokri, Reza, and Hubaux, Jean-Pierre. On the Optimal Placement of Mix Zones. In *Proc. PETS* (Aug. 2009), pp. 216–234.

[53] Golle, Philippe, and Partridge, Kurt. On the anonymity of home/work location pairs. In *Proc. Intl. Conf. on Pervasive Computing* (2009), pp. 390–397.

[54] Gómez-Boix, Alejandro, Laperdrix, Pierre, and Baudry, Benoit. Hiding in the crowd: an analysis of the effectiveness of browser fingerprinting at large scale. In *Proceedings of the 2018 World Wide Web Conference* (2018), International World Wide Web Conferences Steering Committee, pp. 309–318.

[55] Gorlatova, M., Aiello, R., and Mangold, S. Managing base station location privacy. In *Proc. MILCOM* (Nov. 2011), pp. 1201–1206.

[56] Gorlatova, M., Aiello, R., and Mangold, S. Managing location privacy in cellular networks with femtocell deployments. In *Proc. WiOpt Symposium* (May 2011), pp. 418–422.

[57] Gramaglia, Marco, Fiore, Marco, Tarable, Alberto, and Banchs, Albert. Preserving mobile subscriber privacy in open datasets of spatiotemporal trajectories. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications* (2017), IEEE, pp. 1–9.

[58] Guo, Nan, Ma, Linya, and Gao, Tianhan. Independent mix zone for location privacy in vehicular networks. *IEEE Access 6* (2018), 16842–16850.

[59] Hightower, J., LaMarca, A., and Smith, I.E. Practical Lessons from Place Lab. *IEEE Pervasive Computing 5*, 3 (July-Sept. 2006), 32 –39.

[60] Hong, Byeongdo, Bae, Sangwook, and Kim, Yongdae. Guti reallocation demystified: Cellular location tracking with changing temporary identifier. In *NDSS* (2018).

[61] Hopwood, Daira, Bowe, Sean, Hornby, Taylor, and Wilcox, Nathan. Zcash protocol specification version, April 18 2019.

[62] Huang, Haosheng, Gartner, Georg, Krisp, Jukka M, Raubal, Martin, and Van de Weghe, Nico. Location based services: ongoing evolution and research agenda. *Journal of Location Based Services 12*, 2 (2018), 63–93.

[63] Hussain, Syed Rafiul, Echeverria, Mitziu, Singla, Ankush, Chowdhury, Omar, and Bertino, Elisa. Insecure connection bootstrapping in cellular networks: the root of all evil. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks* (2019), ACM, pp. 1–11.

[64] Ikram, Muhammad, Vallina-Rodriguez, Narseo, Seneviratne, Suranga, Kaafar, Mohamed Ali, and Paxson, Vern. An analysis of the privacy and security risks of android vpn permission-enabled apps. In *Proceedings of the 2016 Internet Measurement Conference* (New York, NY, USA, 2016), IMC '16, Association for Computing Machinery, pp. 349–364.

[65] Isaacman, Sibren, Becker, Richard, Cáceres, Ramón, Kobourov, Stephen, Martonosi, Margaret, Rowland, James, and Varshavsky, Alexander. Identifying Important Places in People's Lives from Cellular Network Data. In *Proc. Intl. Conf. on Pervasive Computing* (2011), pp. 133–151.

[66] Jiang, Yixin, Lin, Chuang, Shen, Xuemin, and Shi, Minghui. Mutual Authentication and Key Exchange Protocols for Roaming Services in Wireless Mobile Networks. *IEEE Trans. on Wireless Communications 5*, 9 (2006), 2569–2577.

[67] Jo, Hang-Hyun, Karsai, Márton, Kertész, János, and Kaski, Kimmo. Circadian pattern and burstiness in mobile phone communication. *New Journal of Physics 14*, 1 (2012), 013055.

[68] Johnson, Matthew J, and Willsky, Alan S. Bayesian nonparametric hidden semi-markov models. *Journal of Machine Learning Research 14*, Feb (2013), 673–701.

[69] Kamkar, Samy. evercookie. `https://samy.pl/evercookie/`.

[70] Kesdogan, Dogan, Federrath, Hannes, Jerichow, Anja, and Pfitzmann, Andreas. Location Management Strategies Increasing Privacy in Mobile Communication. In *Information Systems Security*. 1996, pp. 39–48.

[71] Khan, Mohsin, Ginzboorg, Philip, Järvinen, Kimmo, and Niemi, Valtteri. Defeating the downgrade attack on identity privacy in 5g. In *International Conference on Research in Security Standardisation* (2018), Springer, pp. 95–119.

[72] Kido, H., Yanagisawa, Y., and Satoh, T. An anonymous communication technique using dummies for location-based services. In *Proc. Intl. Conf. on Pervasive Services* (2005), pp. 88–97.

[73] Kigerl, Alex C. Infringing nations: Predicting software piracy rates, bittorrent tracker hosting, and p2p file sharing client downloads between countries. *International Journal of Cyber Criminology 7*, 1 (2013), 62.

[74] Kohno, T., Broido, A., and Claffy, K. Remote physical device fingerprinting. *IEEE Trans. on Dependable and Secure Computing 2*, 2 (May 2005), 93–108.

[75] Krumm, John. Inference Attacks on Location Tracks. In *Proc. Intl. Conf. on Pervasive Computing* (May 2007), pp. 127–143.

[76] Kune, D. Foo, Koelndorfer, J., Hopper, N., and Kim, Y. Location leaks on the GSM Air Interface. In *Proc. ISOC NDSS* (Feb. 2012).

[77] Kune, Denis Foo, Koelndorfer, John, Hopper, Nicholas, and Kim, Yongdae. Location leaks on the GSM Air Interface. In *ISOC NDSS* (Feb. 2012).

[78] Liberatore, Marc, Gurung, Bikas, Levine, Brian Neil, and Wright, Matthew. Empirical Tests of Anonymous Voice Over IP. *Elsevier Journal of Network and Computer Applications 34*, 1 (January 2011), 341–350.

[79] Lu, Rongxing, Lin, Xiaodong, Luan, Tom H, Liang, Xiaohui, and Shen, Xuemin. Pseudonym changing at social spots: An effective strategy for location privacy in vanets. *IEEE transactions on vehicular technology 61*, 1 (2011), 86–96.

[80] Ma, Chris Y.T., Yau, David K.Y., Yip, Nung Kwan, and Rao, Nageswara S.V. Privacy vulnerability of published anonymous mobility traces. In *Proc. MobiCom* (2010), pp. 185–196.

[81] Mir, Darakhshan J, Isaacman, Sibren, Cáceres, Ramón, Martonosi, Margaret, and Wright, Rebecca N. Dp-where: Differentially private modeling of human mobility. In *2013 IEEE international conference on big data* (2013), IEEE, pp. 580–588.

[82] Mulder, Yoni De, Danezis, George, Batina, Lejla, and Preneel, Bart. Identification via Location-profiling in GSM Networks. In *Proc. ACM Wrkshp on Privacy in the Electronic Society* (2008), pp. 23–32.

[83] Nandugudi, Anandatirtha, Maiti, Anudipa, Ki, Taeyeon, Bulut, Fatih, Demirbas, Murat, Kosar, Tevfik, Qiao, Chunming, Ko, Steven Y, and Challen, Geoffrey. Phonelab: A large programmable smartphone testbed. In *Proceedings of First International Workshop on Sensing and Big Data Mining* (2013), ACM, pp. 1–6.

[84] Ndhlovu, Lungelo. Facing internet restrictions, journalists turn to vpns. Accessed: 2020-03-24.

[85] Niu, Ben, Li, Qinghua, Zhu, Xiaoyan, Cao, Guohong, and Li, Hui. Enhancing privacy through caching in location-based services. In *2015 IEEE conference on computer communications (INFOCOM)* (2015), IEEE, pp. 1017–1025.

[86] Olejnik, Lukasz, Minh-Dung, Tran, and Castelluccia, Claude. Selling off privacy at auction.

[87] Open Standards for Real-Time Bidding (RTB). OpenRTB Mobile RTB API v1.0, Feb 2011.

[88] Padhye, Jitendra, Firoiu, Victor, Towsley, Donald F., and Kurose, James F. Modeling TCP Reno Performance. *IEEE/ACM Trans. Netw. 8*, 2 (Apr. 2000), 133–145.

[89] Park, Jaegwan, Go, Jaeseung, and Kim, Kwangjo. Wireless authentication protocol preserving user anonymity. In *Proc. SCIS* (2001), pp. 159–164.

[90] Perta, V. C., Barbera, M. V., Tyson, G., Haddadi, H., and Mei., A. A glance through the vpn looking glass: Ipv6 leakage and dns hijacking in commercial vpn clients. In *Proc. PETS* (2015).

[91] Perta, Vasile Claudiu, Barbera, Marco Valerio, and Mei, Alessandro. Exploiting delay patterns for user ips identification in cellular networks. In *International Symposium on Privacy Enhancing Technologies Symposium* (2014), Springer, pp. 224–243.

[92] Qian, Feng, Wang, Zhaoguang, Gerber, Alexandre, Mao, Zhuoqing Morley, Sen, Subhabrata, and Spatscheck, Oliver. Characterizing radio resource allocation for 3g networks. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (2010), pp. 137–150.

[93] Raghavan, Barath, Kohno, Tadayoshi, Snoeren, Alex C, and Wetherall, David. Enlisting isps to improve online privacy: Ip address mixing by default. In *International Symposium on Privacy Enhancing Technologies Symposium* (2009), Springer, pp. 143–163.

[94] Rainie, Lee. Americans' complicated feelings about social media in an era of privacy concerns.

[95] Reed, Michael G., Syverson, Paul F., and Goldschlag, David M. Protocols using anonymous connections: Mobile applications. In *Security Protocols*, vol. 1361 of *LNCS*. 1998, pp. 13–23.

[96] Sasson, E. B., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., and Virza, M. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy* (May 2014), pp. 459–474.

[97] Scott, David W. *Multivariate density estimation: theory, practice, and visualization.* John Wiley & Sons, 2015.

[98] Shokri, Reza, Theodorakopoulos, George, Danezis, George, Hubaux, Jean-Pierre, and Boudec, Jean-Yves. Quantifying Location Privacy: The Case of Sporadic Location Exposure. In *Proc. PETS* (Aug. 2011), pp. 57–76.

[99] Silverman, Bernard W. *Density estimation for statistics and data analysis*, vol. 26. CRC press, 1986.

[100] Smith, Michael, Disselkoen, Craig, Narayan, Shravan, Brown, Fraser, and Stefan, Deian. Browser history re: visited. In *12th USENIX Workshop on Offensive Technologies (WOOT 18)* (2018).

[101] Soroush, Hamed, Sung, Keen, Learned-Miller, Erik, Levine, Brian Neil, and Liberatore, Marc. Disabling GPS is Not Enough: Cellular location leaks over the Internet. In *Proc. PETS* (July 2013), pp. 103–122.

[102] Soroush, Hamed, Sung, Keen, Learned-Miller, Erik, Levine, Brian Neil, and Liberatore, Marc. Turning off gps is not enough: Cellular location leaks over the internet. In *International Symposium on Privacy Enhancing Technologies Symposium* (2013), Springer, pp. 103–122.

[103] Sung, Keen, Biswas, Joydeep, Learned-Miller, Erik, Levine, Brian N, and Liberatore, Marc. Server-side traffic analysis reveals mobile location information over the internet. *IEEE Transactions on Mobile Computing 18*, 6 (2018), 1407–1418.

[104] Sung, Keen, Levine, Brian Neil, and Liberatore, Marc. Location privacy without carrier cooperation. In *IEEE Workshop on Mobile Security Technologies, MOST* (2014), p. 148.

[105] Thiagarajan, Arvind, Ravindranath, Lenin, Balakrishnan, Hari, Madden, Samuel, and Girod, Lewis. Accurate, Low-Energy Trajectory Mapping for Mobile Devices. In *USENIX NSDI* (Mar. 2011).

[106] Tu, Zhen, Xu, Fengli, Li, Yong, Zhang, Pengyu, and Jin, Depeng. A new privacy breach: User trajectory recovery from aggregated mobility data. *IEEE/ACM Transactions on Networking 26*, 3 (2018), 1446–1459.

[107] U.S. Dept. of Justice. The National Strategy for Child Exploitation Prevention and Interdiction: A Report to Congress. `https://www.justice.gov/psc/file/842411/download`, April 2016.

[108] Vines, Paul, Roesner, Franziska, and Kohno, Taayoshi. Exploring adint: Using ad targeting for surveillance on a budget-or-how alice can buy ads to track bob. In *Proceedings of the 2017 on Workshop on Privacy in the Electronic Society* (2017), ACM, pp. 153–164.

[109] Viterbi, A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory 13*, 2 (April 1967), 260–269.

[110] Wang, Jinbao, Cai, Zhipeng, Li, Yingshu, Yang, Donghua, Li, Ji, and Gao, Hong. Protecting query privacy with differentially private k-anonymity in location-based services. *Personal and Ubiquitous Computing 22*, 3 (2018), 453–469.

[111] Wang, Shengling, Hu, Qin, Sun, Yunchuan, and Huang, Jianhui. Privacy preservation in location-based services. *IEEE Communications Magazine 56*, 3 (2018), 134–140.

[112] Wang, Zhaoguang, Qian, Zhiyun, Xu, Qiang, Mao, Zhuoqing, and Zhang, Ming. An untold story of middleboxes in cellular networks. In *ACM SIGCOMM* (Aug. 2011), pp. 374–385.

[113] Wright, Matthew, Adler, Micah, Levine, Brian Neil, and Shields, Clay. The Predecessor Attack: An Analysis of a Threat to Anonymous Communications Systems. *ACM Transactions on Information and System Security (TISSEC) 4*, 7 (November 2004), 489–522.

[114] Xiao, Yonghui, and Xiong, Li. Protecting locations with differential privacy under temporal correlations. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (2015), pp. 1298–1309.

[115] Xiao, Yonghui, Xiong, Li, Zhang, Si, and Cao, Yang. Loclok: Location cloaking with differential privacy via hidden markov model. *Proceedings of the VLDB Endowment 10*, 12 (2017), 1901–1904.

[116] Xu, Qiang, Gerber, Alexandre, Mao, Zhuoqing, and Pang, Jeffrey. AccuLoc: practical localization of performance measurements in 3G networks. In *ACM MobiSys* (Aug. 2011), pp. 183–196.

[117] Xu, Qiang, Huang, Junxian, Wang, Zhaoguang, Qian, Feng, Gerber, Alexandre, and Mao, Zhuoqing. Cellular data network infrastructure characterization and implication on mobile content placement. In *ACM SIGMETRICS* (2011), pp. 317–328.

[118] Yang, Guomin, Wong, DuncanS., and Deng, Xiaotie. Efficient anonymous roaming and its security analysis. In *Applied Cryptography and Network Security*, vol. 3531 of *LNCS*. 2005, pp. 334–349.

[119] Yin, Ling, Wang, Qian, Shaw, Shih-Lung, Fang, Zhixiang, Hu, Jinxing, Tao, Ye, and Wang, Wei. Re-identification risk versus data utility for aggregated mobility research using mobile phone location data. *PloS one 10*, 10 (2015), e0140589.

[120] Zang, Hui, and Bolot, Jean. Anonymization of Location Data Does Not Work: A Large-scale Measurement Study. In *Proc. ACM MobiCom* (2011), pp. 145–156.

[121] Zhu, Jianming, and Ma, Jianfeng. A new authentication scheme with anonymity for wireless environments. *IEEE Trans. on Consumer Electronics 50*, 1 (2004), 231–235.

[122] Zimmeck, Sebastian, Li, Jie S, Kim, Hyungtae, Bellovin, Steven M, and Jebara, Tony. A privacy analysis of cross-device tracking. In *26th USENIX Security Symposium (USENIX Security 17)* (2017), pp. 1391–1408.