Hasselt University
Transnationale Universiteit Limburg

Assessing and improving security and privacy for smartphone users

---

Dissertation submitted for the degree of

Doctor of Philosophy in Computer Science,

at Hasselt University

to be defended by

Bram Bonné

on August 31, 2017

Promotor: Prof. Dr. Wim Lamotte

Co-promotor: Prof. Dr. Peter Quax

2011 – 2017

Abstract

Smartphone and other mobile device usage has increased greatly in the past years. This increased popularity has also led to a changed security and privacy landscape, with more personal devices being outfitted with a plethora of sensors that allow to track our every step, and a vastly larger attack surface than older, more static devices. This allows a variety of actors, including malicious hackers, state-sponsored entities and legitimate service providers, to have access to a large trove of mobile user data.

In this dissertation, we assess the current state of privacy and security on smartphones, we create and gauge awareness of smartphone users around these issues, and we provide solutions to enhance security and privacy on mobile devices. To show the ease with which smartphone users' data can be gathered surreptitiously, we describe a mechanism for involuntarily tracking visitors at mass events making use of Wi-Fi technology, and show that this can be implemented at a low cost, allowing location tracking of 29% of visitors at a major music festival. We show how these techniques can be (ab)used in different scenarios (notably, by using the gathered data to compare different opportunistic routing algorithms that can be used for ad-hoc communication at mass events), and provide an open platform to researchers that can be used to quantify the impact and remediation rate of similar wireless protocol vulnerabilities.

We create awareness about these issues, and we explain to smartphone users how they can secure themselves against them. For this, we provide a method to inform mobile device users when using wireless networks, showing privacy-sensitive (but anonymized) information about passers-by on a public display. We later use the same setup to inform audiences in talks on security awareness. Results from our user studies also show that specific, personalized scenarios may help to better inform users about security and privacy issues (increasing awareness of 76% of the participants in one study), and that the increased awareness leads to as much as 81% of device users willing to put an extra effort into securing their smartphones. Interestingly, we also show in a later study that an increased awareness does not automatically translate to better security practices.

Additionally, we perform two studies to measure users' privacy and security behaviors when using their smartphones. For the first study, we look at how aware users are about connections

being made by apps on their device, while taking into account the security of both the Wi-Fi networks used and the connections made over these networks. For the second study, we extend the Paco ESM study tool to be able to examine the reasons why Android users install or remove an app at the time this happens, to look at the motivation behind granting or denying a permission right after users make their choice, and to assess how comfortable and aware users are about their decisions at a later point in time.

We provide recommendations to different stakeholders (developers, manufacturers, network providers, researchers and mobile device users) on how to improve privacy and security on mobile devices without affecting usability, some of which have already been implemented by operating system manufacturers. Part of these recommendations are implemented as a tool that automatically mitigates Wi-Fi attacks for Android smartphones, which is distributed to the general public. In addition, we formulate a proposal to improve transparency in how user data is shared by service providers to third parties.

Acknowledgments

This dissertation is the result of work conducted during the period between September 2011 and June 2017 at Hasselt University. It would have never been possible without the help of some people, whom I would like to thank here.

First, many thanks go out to my promotor and co-promotor Wim Lamotte and Peter Quax. They provided me with the necessary guidance throughout the past 6 years, not only with respect to the research that went into completing my PhD, but also through assistance when performing teaching duties. When performing experiments involving deploying and monitoring large amounts of Wi-Fi scanners, they were also there to help with the hands-on work, soldering together the scanners and climbing structures to suspend them. Without their continued support, this thesis would not have been possible.

I also want to thank my PhD jury: Frank Van Reeth, Igor Bilogrevic, Karin Coninx, Lieven Desmet, Marc Gyssens and Tristan Henderson, for reviewing my text, and for their comments that helped make this dissertation.

I want to thank the rest of the EDM, and by extension the entire computer science faculty for the great working environment. Specifically, I want to mention Arno Barzan and Pieter Robyns, who have contributed greatly to this research, and helped in keeping me sane throughout this work. They were always there for support and laughs. I also want to thank Kris Luyten, Jo Vermeulen, and Fabian Di Fiore, for respectively encouraging me to go work at Google, for the genuine interest in my work, and for dark industrial techno.

Similarly, I want to thank Sai Teja Peddinti, Nina Taft, and the rest of the wonderful people I had a chance to work with at Google for the awesome working environment. They immediately made me feel like part of the team, and supported my work all the way through, even at the expense of other projects they had to manage at the same time.

There are some other people who provided great support for individual experiments. First of all, a huge thanks goes out to the Pukkelpop organization, including Chokri Mahassine himself, for allowing us to perform our experiments at their festival, and to The Safe Group for helping out and for staying incredibly nice while we abused their infrastructure. Thank you also to Bob Hagemann and the rest of the `WiGLE.net` team, for allowing the nearly

unlimited use of their wardriving database to inform people about the dangers of wireless networks.
Lastly, I want to thank my parents and my brother (who is incidentally also my best friend), my girlfriend Tine, all my other amazing friends[1] and possibly the best family in the world. They all contributed so much to this thesis without even realizing it, by encouraging me to pull through, and by just being the nicest and friendliest people around.

Thank you all so much,

Bram Bonné

[1] "Friends" sounds so offhanded. So here we go; at the very least (apart from family and (ex-)colleagues): Alicja, Andrea, Anneleen, Ben, Bert, Brecht, Carl, Cedric, Daniël, Dries, Eef, Hannah, Hanne, Hélène, Ian, Ine, Jasper, Jimmy, Joachim, Joke, Jolien, Julie, Kevin, Koen, Koenraad, Kristof, Lore, Lowie, Maarten, Marijke, Marleen, Michelle, Miet, Nathalie, Nel, Nico, Phung, Rachel, Roald, Ruben, Sam, Sarina, Sebastiaan, Sophie, Stephanie, Thierry, Thijs, Tiberd, Tinne, Tom, Wim, Xavier and Yifei.

Contents

Abstract i

Acknowledgments iii

Contents v

1 Introduction and research questions 1

I Security and privacy issues in smartphone connectivity 7

2 WiFiPi: Involuntary Tracking of Visitors at Mass Events 11

2.1 Introduction . . . 13
2.2 Related work . . . 13
2.3 Technical background . . . 15
2.3.1 Scanning for wireless networks . . . 15
2.3.2 Connecting to a network . . . 18
2.4 The WiFiPi detection mechanism . . . 18
2.4.1 Detecting a device . . . 19
2.4.2 Tracking the location of a device . . . 20
2.5 Implementation . . . 21
2.6 Experiments . . . 22
2.6.1 Pukkelpop 2012 . . . 24
2.6.2 University campus . . . 24
2.6.3 Pukkelpop 2013 and WiFiPi 2.0 . . . 25
2.7 Applications . . . 28
2.7.1 Real-time crowd management and marketing . . . 28
2.7.2 Mobility models for simulations . . . 29
2.7.3 Ubiquitous computing . . . 29
2.8 Privacy implications . . . 30

2.9 Conclusion 31

3 A Comparative Simulation of Opportunistic Routing Protocols Using Realistic Mobility Data Obtained From Mass Events 33
3.1 Introduction 35
3.2 Background and Related Work 35
3.3 Mobility 36
3.3.1 Collecting Data 37
3.3.2 Calculating movement paths 37
3.4 Simulations 38
3.4.1 Simulation Settings 39
3.4.2 Routing Protocols 41
3.5 Results and analysis 42
3.5.1 Metrics 42
3.5.2 Individual Protocol Examination 44
3.5.3 Comparison 45
3.5.4 Candidate Selection 46
3.6 Conclusion 48

4 Assessing the Impact of 802.11 Vulnerabilities using Wicability 49
4.1 Introduction 51
4.2 Capability aggregation 52
4.2.1 Acquisition 52
4.2.2 Matching and processing 53
4.2.3 Presentation 54
4.3 Case study: prevalence of devices susceptible to active probing attacks 54
4.4 Datasets 56
4.5 Conclusion 56

II Creating and assessing user awareness 61

5 Raising Awareness on Ubiquitous Privacy Issues with SASQUATCH 65
5.1 Introduction 67
5.2 Mobile Phones that “Never Forget” 68
5.3 The SASQUATCH System 68
5.3.1 Inferring a smartphone’s whereabouts 69
5.3.2 Determining a network’s authentication type 70
5.4 Study 71
5.5 Results 74
5.6 System analysis and limitations 77

5.7 Conclusion . . . 79

6 Understanding Wi-Fi Privacy Assumptions of Mobile Device Users 81
6.1 Introduction . . . 83
6.2 Related work . . . 84
6.3 Methodology . . . 86
6.3.1 Connection monitoring . . . 86
6.3.2 Exit survey . . . 88
6.4 Participants . . . 90
6.5 Results . . . 92
6.6 Discussion . . . 96
6.7 Conclusion . . . 99

7 Exploring privacy-sensitive decision making in Android using in-context surveys 101
7.1 Introduction . . . 103
7.2 Related Work . . . 105
7.3 Methodology . . . 107
7.3.1 Designing the Surveys . . . 107
7.3.2 Recruitment and Incentives . . . 110
7.3.3 Ethical Considerations . . . 110
7.3.4 Limitations . . . 110
7.4 Technical implementation . . . 112
7.4.1 App Installation and Removal Triggers . . . 112
7.4.2 Permission Change Triggers . . . 113
7.4.3 Generating and Surfacing Surveys . . . 113
7.5 App Decisions . . . 115
7.5.1 Data Summary . . . 115
7.5.2 App Installs . . . 116
7.5.3 App Removals . . . 118
7.6 Permission Decisions . . . 120
7.6.1 Permission denials . . . 120
7.6.2 Permission Grants . . . 125
7.6.3 Other influences . . . 128
7.7 Conclusion . . . 128

III Towards improving security and privacy for mobile devices and users 133

8 Technical solutions to smartphone privacy and security issues 137
8.1 Introduction . . . 139

8.2 Recommendations on smartphone security . . . 139
8.2.1 What the smartphone user can do . . . 139
8.2.2 What a developer / manufacturer can do . . . 141
8.2.3 What network providers and ISPs can do . . . 143
8.2.4 What security and privacy researchers can do . . . 143
8.3 Automatically solving privacy issues with Wi-Fi PrivacyPolice . . . 144
8.3.1 Preventing network leakage . . . 145
8.3.2 Preventing evil twin attacks . . . 145
8.3.3 From proof-of-concept to consumer app . . . 146
8.4 Comparison to other mitigation strategies . . . 147
8.5 Conclusion . . . 148

9 The Privacy API: Facilitating Insights In How One's Own User Data Is Shared 149
9.1 Introduction . . . 151
9.2 Definitions . . . 152
9.3 The API in practice . . . 153
9.4 Methods of enforcement . . . 154
9.5 Considerations and limitations . . . 156
9.6 Related work . . . 156
9.7 Discussion . . . 157
9.8 Conclusion . . . 158

10 A look at the future: the Internet of Things 159

11 Conclusion and Future Work 165
11.1 Conclusion . . . 167
11.2 Future work . . . 171

Appendices 173

A Nederlandse Samenvatting (Dutch Summary) 175

B Survey questions for the study on Wi-Fi privacy 177
B.1 Recruitment survey questions . . . 178
B.2 Exit survey questions . . . 178
B.2.1 Introductory questions . . . 179
B.2.2 Network questions . . . 179
B.2.3 General questions . . . 180
B.2.4 Connection awareness questions . . . 181
B.2.5 Feedback question . . . 181

C Survey questions for the study on Android permissions **183**
C.1 In-situ questions 184
C.1.1 App installation scenario 184
C.1.2 App removal scenario 185
C.1.3 Permission grant scenario 185
C.1.4 Permission deny scenario 186
C.2 Exit Survey 186

D Scientific Contributions and Publications **189**

Bibliography **209**

Chapter 1

Introduction and research questions

Since the earliest computers, computing devices have become smaller and more personal over time. The earliest commercial general purpose computers, such as the Ferranti Mark 1, the UNIVAC and the IBM 650, measured somewhere in the order of $10\,m^3$. The invention of the transistor – replacing large vacuum tubes in the earlier computers – caused computers to be much smaller in size, an effect that was reinforced by the advent of integrated circuits, which allowed for a personal computing device in every household. The System on a Chip (SoC) is the latest step in consolidating even more of the circuitry into a single chip, supporting the development of ever smaller and more mobile computing devices such as smartphones and smartwatches.
Smartphone and other mobile device usage has increased greatly in the past years: even during the six years it took to complete the research reflected in this dissertation, the device usage landscape has changed significantly. Data from Eurostat shows that over the past 5 years, the number of individuals in the EU aged 16 to 74 that are using a mobile phone to access the internet has increased from only 19% in 2011 to 56% in 2016 [Eurostat, 2016]. This trend is even more noticeable in emerging economies: in just two years, smartphone ownership rates have increased by more than 25% (with an increase of 42% in Turkey alone) [Poushter, 2016].

The rise in popularity of mobile devices has also led to a changed security and privacy landscape. Every step of the miniaturization of computers has caused them to become more personal. Indeed, the earliest computers were largely tied to a company and used by different employees of the same corporation; personal computers were tied to a household and used by different members of the same family; smartphones, finally, are almost always tied to a single person. As smartphones are considered to be at least as private by their owners as

desktop computers [Urban et al., 2012], they contain a trove of personal data, with access to even more data through apps that are connected to cloud services and persistent logins to websites in the device's web browser. Adding to this is the fact that smartphones contain a wealth of sensors (including, among others, an accelerometer, a GPS chip, a Wi-Fi chip and a light sensor), and that they are always carried around by their owners, something that has earned them the nickname of 'ideal tracking device'.

Not only do these devices contain more data, this data is also more easily available to (both benign and malicious) third parties. First, smartphones are always-on, always-connected devices, which increases the number of channels and time over which their privacy-sensitive data is available. Furthermore, because of their form factor and mobility, these devices can get lost or stolen more easily. Because of these reasons, smartphones are deemed to have a large *attack surface*, consisting of the number of ways a third-party malicious actor is able to attempt to access the data.

Not only malicious actors are attempting to gather privacy-sensitive data of smartphone users. Increasingly, legitimate entities are gathering this information for various purposes. One example is state-sponsored actors, of which we know from e.g. the Snowden leaks and WikiLeaks' CIA files that they actively use data-gathering techniques to collect information from mobile devices. Another common example of non-malicious actors gathering privacy-sensitive data are internet services that, instead of (or in addition to) asking for a direct payment, monetize their users' data as their main source of revenue. This includes not only social networks; non-free service providers such as mobile operators or hardware manufacturers have also been using these methods as a way to increase their revenue. [Lee, 2016; Troianovski, 2013]. All of these services have access to (part of) this large trove of mobile data, either because their app or website requires it as part of providing their service, or because they control (part of) the connection the smartphone has with the internet. In some instances, these third parties do not even require any cooperation from the user. Indeed, techniques that will be described throughout this text are already being used by commercial entities to gather data on their customers. Despite the predominant public opinion that this type of data collection is only possible for large organizations with world-wide networks, both legitimate and illicit data gathering can be done with a fairly simple setup using off-the-shelf wireless hardware.

Furthermore, the rise in popularity of mobile devices has also caused Wi-Fi networks to become more prevalent, often being offered by either commercial or public entities, or by internet service providers (ISPs) as a service to their customers. Indeed, as we will show in Chapter 6, users connect to an average of 8 Wi-Fi networks in a 30-day period. Connecting to the internet using one of these Wi-Fi networks entails some form of trust: the connections themselves – and, in case these connections happen unencrypted, their data – can be monitored by the provider of the network. Moreover, networks provided by these entities often lack any form of security, allowing not only the network provider, but also other third parties within range of the network to eavesdrop on communications.

The aim of this work is to assess the current state of privacy and security on smartphones, to create and gauge awareness of smartphone users around these issues, and to provide solutions to enhance security and privacy on mobile devices. We start from the main question "Are mobile device users aware about the security and privacy issues inherent in using their devices?", and deduce the following research questions from this:

RQ1 What privacy and security issues exist in using a smartphone on wireless networks? How can these issues be (ab)used by a third party?

RQ2 How prevalent are the aforementioned issues in today's devices?

RQ3 Are mobile device users aware about personal data transmitted by their devices?

RQ4 Does privacy sensitiveness influence the behavior of mobile device users?

RQ5 How do mobile device users make decisions regarding privacy sensitive data?

RQ6 Can we inform non-technical mobile device users about the aforementioned risks? Does the increased awareness cause better security habits?

RQ7 How can we improve privacy and security for mobile device users?

In the first part of this thesis, we describe the privacy and security issues that result from the shift to mobile device usage, aiming to tackle research question **RQ1**. We will see that on one hand, there is a major impact on the security of these devices: while attacks on Wi-Fi networks and their users have existed for a long time, they have become more prevalent as devices are more mobile, and the attack surface (in the form of Wi-Fi networks) has increased in size. On the other hand, privacy of mobile users might be impacted even more, as the shift to mobile devices has opened a whole new range of techniques for gathering privacy-sensitive information. As we will see, mobile devices allow the location of their users to be tracked surreptitiously by third parties, and allow these third parties to infer personally identifiable data from the user by monitoring only their signals. We show how these techniques can be (ab)used in different scenarios, and provide a way for observing how prevalent security issues are in today's devices (covering research question **RQ2**).
The second part provides an assessment of how aware mobile device users are about the issues discussed in the first part, tackling research questions **RQ3** to **RQ6**, while in the meantime continuing our assessment of the prevalence of security issues in modern devices (research question **RQ2**). This is done in three different studies. The first two studies include presenting participants with their own privacy-sensitive information that we were able to gather. These studies provide an image of how aware users are about the ability for third parties to gather privacy-sensitive information from their devices (and the consequences thereof), and of how concerned users are about this information being available. At the same time, our studies aim to increase awareness among smartphone users by informing them about existing issues. We

will see that, whereas these issues are very prevalent due to the way mobile devices are used nowadays, users are often unaware. While we will demonstrate techniques that are highly effective in increasing this awareness, we will also show that, interestingly, an increased awareness does not automatically translate to better security practices. We end the second part with a third study, exploring how users make privacy sensitive decisions when using their devices, by employing an experience sampling methodology in order to ask users the reasons influencing their decisions immediately after they decide.
In the third part of this thesis, we provide a variety of different solutions to the problems presented in the second chapter, from a variety of contexts. These solutions are presented as suggestions to changes in technology, habits and legislature, and in the form of actual implementations. This addresses our final research question **RQ7**.
Our main contributions are as follows:

C1 We describe a mechanism for involuntary tracking of visitors at mass events which makes use of Wi-Fi technology, and show how this can be implemented at a low cost (related to **RQ1**).

C2 We compare and select different opportunistic routing algorithms that can be used for ad-hoc communication at mass events, based on simulations with real mobility data. We were helped for performing the actual simulations by Arno Barzan.

C3 We provide an open platform to researchers that can be used to quantify the impact and remediation rate of wireless protocol vulnerabilities (related to **RQ2**). Pieter Robyns helped by extracting the information from various datasets.

C4 We provide a method to inform mobile device users about privacy and security issues when using wireless networks, and use this method to successfully create user awareness, explaining how to secure against these threats in the process (related to **RQ1** and **RQ6**).

C5 We provide statistics on mobile users' privacy and security practices on wireless networks, in the form of data about Wi-Fi usage and connections made by apps (related to **RQ2**).

C6 We assess mobile users' awareness about insecure connections being made by apps on their devices over insecure wireless networks, together with the users' level of comfort about these connections (related to **RQ3**). We analyze the impact of expertise in computer networks and the usage of privacy enhancing technologies on this awareness and **RQ4**). Gustavo Rovelo Ruiz generated statistics from the survey dataset.

C7 We extend the Paco ESM study tool to be able to query users about the reasons behind privacy-related decisions they make on their Android device, and make these modifications available to the broader research community.

C8 We conduct the first study that examines the reasons why Android users install or remove an app at the time this happens, and the motivation behind granting or denying a permission right after users make their choice (related to **RQ5**). We also assess how comfortable and aware users are about their decisions at a later point in time (related to **RQ3**). This study was performed together with Sai Teja Peddinti, Igor Bilogrevic and Nina Taft, all helping out in equal amounts with the data analysis.

C9 We provide recommendations to different stakeholders (developers, manufacturers, network providers, researchers and mobile device users) on how to improve privacy and security on mobile devices without affecting usability. These recommendations are based on the results from our previous studies (related to **RQ7**). In addition, we formulate a proposal to improve transparency in how user data is shared by service providers to third parties.

C10 We design and develop a tool that automatically mitigates Wi-Fi attacks for Android smartphones (related to **RQ7**), and make this tool available to the general public.

We presented our contributions to the larger research community, in the form of three journal publications, three presentations and publications at international conferences, and three presentations and publications at workshops. In addition to purely scientific contributions, we aimed to increase awareness through scientific outreach, by giving talks on security and privacy to the general public (including members and staff of the European Commission, members of multiple city councils, and younger children). A more complete list is available in Appendix D.

Part I

Security and privacy issues in smartphone connectivity

Introduction

For long, mobile phones have been thought of as "trusted devices": they are used for private communication, coupled to a single user. However, in a time where smartphones dominate the mobile landscape, this trust is often unfounded. In reality, smartphones are known to collect all kinds of data – ranging from a user's contacts to precise location data – via many different channels. For example, leaked documents by Edward Snowden show that popular smartphone apps were targeted by the NSA and GHCQ because of the vast amounts of user data they collect (as published in *The Guardian* on January 27, 2014).
Even though a significant amount of people carry a smartphone around all the time, most of them are unaware about the privacy impact of using such a device to connect to wireless networks. For example, we will show that the strategies currently used by mobile operating systems to search and connect to available wireless networks involve sharing a list of previously accessed networks.

The first part of this dissertation describes a method that allows smartphone users to be tracked by a third party surreptitiously (that is, without active cooperation of the smartphone users themselves). We show how this method can be implemented at very low cost on Raspberry Pi computers, and trial its implementation in two different contexts: a popular Belgian music festival and our university campus. The fact that this low-cost implementation allows a multitude of parties to surreptitiously track smartphone users leads to important security concerns.

Apart from discussing the security issues themselves, we also show how this method can be used for other research, providing the example of using the gathered mobility data to perform simulations of different opportunistic routing protocols. The routing protocol that is deemed to work best in the situations of real mobility could then be considered suitable for creating applications routing data between smartphone users directly, without the need for any network infrastructure.

We close the first part of this thesis by describing a system that allows researchers to find the impact of these kinds of wireless vulnerabilities, through the automated collection and

analysis of large datasets containing IEEE 802.11 Information Elements (IEs) transmitted by access points and stations. These IEs allow to see the capabilities that are supported by devices, thereby showing the fraction of devices that are at risk.

Chapter 2

WiFiPi: Involuntary Tracking of Visitors at Mass Events

2.1 Introduction 13
2.2 Related work 13
2.3 Technical background 15
2.3.1 Scanning for wireless networks 15
2.3.2 Connecting to a network 18
2.4 The WiFiPi detection mechanism 18
2.4.1 Detecting a device 19
2.4.2 Tracking the location of a device 20
2.5 Implementation 21
2.6 Experiments 22
2.6.1 Pukkelpop 2012 24
2.6.2 University campus 24
2.6.3 Pukkelpop 2013 and WiFiPi 2.0 25
2.7 Applications 28
2.7.1 Real-time crowd management and marketing 28
2.7.2 Mobility models for simulations 29
2.7.3 Ubiquitous computing 29
2.8 Privacy implications 30
2.9 Conclusion 31

2.1 Introduction

With the recent growth of smartphone usage, a large percentage of the population now carries a device frequently sending out signals, which can be detected by eavesdroppers nearby. At the time we published the work from which this chapter is derived (June 2013), it was estimated that over 50% of U.S. mobile subscribers owned a smartphone [Nielsen, 2012]. This number was only expected to increase over time as more than 1.3 million new Android devices are activated worldwide every day [Spradlin, 2012]; a prediction that, as we mentioned in Chapter 1, was confirmed afterwards. Modern smartphones have Wi-Fi communication enabled by default, allowing their owners to be detected at any moment by scanning the ether for Wi-Fi packets broadcast by their devices.

In this chapter, we show one specific way in which smartphone users are exposing themselves to third parties gathering privacy-sensitive data; we show how their Wi-Fi signals allow them to be tracked *involuntarily* across a specific area. To someone trying to gather movement data from specific smartphone users, involuntary tracking provides a significant advantage compared to other techniques for tracking where the subjects need to actively cooperate, either by carrying a specialized tracking device or by actively and willingly sharing information about their location. Systems like the one described in this chapter do not require user consent and are therefore capable of tracking a much larger sample set of the population. At the same time, this creates a disadvantage to the smartphone users themselves, who are tracked without their consent and possibly without knowing the tracking is taking place.

This chapter makes the following contributions. First, we describe a mechanism for tracking visitors at mass events which makes use of Wi-Fi technology. We explain how this mechanism works at a high level, and continue by discussing its implementation. Next, we describe how the detection mechanism was and is being used in two different contexts to infer movement patterns from visitors. One of these contexts is a three-day international music festival attracting 100 000 visitors every year. The other is a university campus, where we have been tracking students and staff for a period of three months. Section 2.7 provides an overview of possible applications of the tracking method in different domains. In Section 2.8, privacy implications for this technology are discussed. We conclude and present an overview of future work in section 2.9.

2.2 Related work

Several techniques have been proposed over the years to accomplish location determination and to provide the ability to track the movement of objects and people. One of these techniques is the use of wireless sensor networks (WSNs), where tiny motes are attached to objects to achieve tracking abilities [Kung and Vlah, 2003]. Although a high degree of precision can be obtained, the disadvantages of such a method are clear: the cost of motes is non-negligible, they are not available off-the-shelf and, as such, the number of objects that

could be tracked is limited in practice. A WSN system requires active consent and participation from users to carry the motes. Alternatively, the use of tracking applications (sometimes as part of applications providing other functionality) on more common hardware equipped with GPS sensors has been proposed. Although this approach solves the cost issue to some degree, users still need to actively collaborate in the tracking by installing an application and agreeing to be tracked (unless the tracking software is part of malware, a topic which is not discussed here). Given the fact that the goal is to show how a non-obtrusive tracking technique that requires no active consent from people being tracked can be designed, these methods are not applicable.

Versichele et al. [Versichele et al., 2012] developed a method for tracking people at mass events which uses Bluetooth signals sent out by mobile phones to detect a person's location. While similar to the approach described in this chapter, this detection method requires phones to have their Bluetooth functionality set to 'discoverable', a feature which is disabled on modern smartphones for security reasons [Android Open Source project, 2013; Apple Inc., 2012]. Because of this limitation, Bluetooth tracking can only be used to track older generation cell phones, resulting in a coverage rate of around 8% of the population at the time of our experiments. We expect this coverage rate to have decreased even further in 2017 as more people switched from older cell phones to smartphones. Our approach on the other hand requires only control signals sent out as part of the 802.11 protocol, which are required for Wi-Fi communication to function properly. This tracking method is future proof because unlike Bluetooth, Wi-Fi is enabled by default on modern smartphones. Furthermore, our method does not depend on a smartphone being put into a discoverable mode, nor does it require the smartphone to be actually connected to a wireless network. The tracking method of Versichele et al. can be used complementary to our own method to allow for tracking visitors using both Bluetooth and Wi-Fi signals. Using this combination, both older cell phones and smartphones can be tracked, which may provide a coverage rate close to the sum of the individual coverage rates for both tracking methods.

Other work has been done in the area of using Wi-Fi communication to obtain information (besides location) about smartphone users. Cunche et al. have used passive Wi-Fi monitoring [Cunche and Boreli, 2012] to derive social links between smartphone owners. Moreover, Rose et al. [Rose and Welsh, 2010] have shown that SSIDs found in 802.11 probe requests can be used to produce a list of locations a smartphone user has visited. The described technique works by looking up the broadcasted SSIDs in the WiGLE wardriving database [WiGLE.net, 2016]. This database contains the locations of different wireless networks all over the world, submitted by users, and identified by their SSIDs. Our approach differs from the work done by Rose et al. in that it allows for tracking users without the presence of an infrastructure of access points (with specific SSIDS). Rose's method also is unable to infer the time at which a device was at a certain location. In Chapter 5, we will build upon and extend Rose's method to create awareness among smartphone users. Similarly, Becker et al. [Becker et al., 2013] have described how cellular telephone networks can be used to study human mobility

on a large scale. This, too, measures mobility on a much larger, less granular scale than the technique described in this chapter.

Network simulations use either synthetic movement data or data gathered from real-life crowd tracking, with a preference for the latter [Aschenbruck et al., 2011; Camp et al., 2002]. The CRAWDAD data archive [Yeo et al., 2006a] is a resource containing wireless trace data from many contributing parties. Methods for acquiring this data vary from equipping people with sensors to using network traces from access points which are under the control of researchers. We believe that our method can provide substantial benefits for people willing to extend the CRAWDAD data archive, by allowing for tracking of visitors without requiring active cooperation.

2.3 Technical background

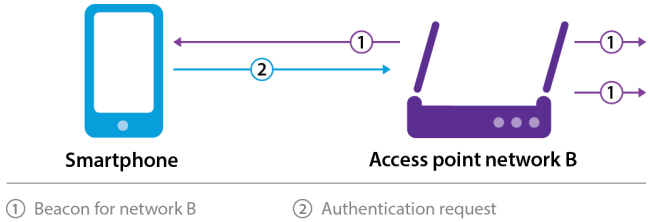
Before we discuss how the WiFiPi system is able to track the location of a device, we first provide a high-level overview of the parts of the 802.11 protocol which are exploited to achieve this, and which will allow us to collect privacy-sensitive information about smartphone users in the SASQUATCH study described in Chapter 5.

2.3.1 Scanning for wireless networks

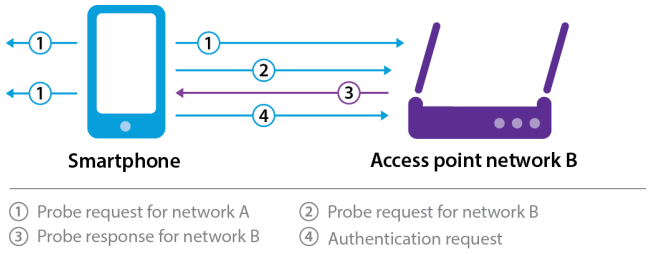
Before a wireless (IEEE 802.11) device can connect to a network, it needs to be aware of the different access points that are in range. The IEEE 802.11 standard [IEEE Standards Association, 2012] describes two different methods that can be used to scan for wireless (IEEE 802.11) networks: passive scanning (or 'stumbling') and active scanning. Passive scanning (see Figure 2.1(a)) is used with all regular networks. This method works by listening for specific packets – called *beacons* – which are sent out by access points with small intervals. These beacons contain, among other information, the SSID of the access point. By listening for these packets, a device knows exactly which networks are in range.

The active scanning method (see Figure 2.1(b)) is used in a second class of wireless networks, called *hidden* or *cloaked* networks. In order to remain invisible to devices unaware of its existence, this type of network does not send out beacons. Instead, devices that know of this network need to scan for the access point's presence in a proactive manner. For this purpose, *probe requests* are used. Broadcasting a probe request containing a specific SSID is similar to asking: "Is the network with name `SSID` around?". If an access point receives a probe request containing its own SSID, it responds by sending a *probe response* directly to the device that sent out the probe request, notifying the device of its presence.

Active scanning is also used when scanning for known (or 'remembered') networks in modern smartphones. Since constant passive scanning requires the smartphone's Wi-Fi radio to be powered even when no connection is available, this method would have a significant im-



(a) Passive scanning. The access point broadcasts its availability to everyone in range.



(b) Active scanning. The smartphone actively searches for access points that are in range.

Figure 2.1: The difference between active and passive scanning in modern Wi-Fi devices.

pact on the smartphone's battery life. Because of this, Android, iOS and Windows Phone implement a mechanism that works as follows:

1. Send out a probe request for the first 'remembered' network in the smartphone's preferred network list (PNL).

2. Keep the Wi-Fi radio powered on for a short amount of time, listening for access points sending probe responses.

   (a) If a probe response is received, initiate a connection with the network.

   (b) If no probe response is received within the chosen time window, go back to step 1 while choosing the next network in the PNL.

   (c) If no probe response is received, and we arrived at the last network in the PNL, put the Wi-Fi radio to sleep for n seconds (the *probe request interval*).

In reality, smartphones running Android send probe requests only for the first 16 networks in their PNL, as can be seen from the `WPAS_MAC_SCAN_SSIDS` constant in the source code.[1]

[1]The source code for `wpa_supplicant` as it is included in Android 7.7.1 is available at `http://androidxref.com/7.1.1_r6/xref/external/wpa_supplicant_8/src/drivers/driver.h`.

Table 2.1: Average values of probe request intervals for popular smartphone vendors.

Vendor	Interval	Sample size
Sony	11 seconds	3 devices
Samsung	22 seconds	30 devices
Huawei	23 seconds	5 devices
Apple	24 seconds	149 devices
Nokia	25 seconds	11 devices
HTC	26 seconds	9 devices
Asus	30 seconds	4 devices
Blackberry	30 seconds	2 devices
LG	32 seconds	16 devices

Devices running iOS are not known to limit the amount of networks for which a probe request is broacasted.

The probe request interval, i.e. the number of seconds during which the Wi-Fi radio is put to sleep in between scanning for the same set of networks, is dependent on the smartphone vendor, model, operating system and on the state of the smartphone (standby, screen on, connected to a network, etc.) [Cisco Systems, 2013]. Based on our preliminary experiments, we list the average probe request intervals for some popular vendors in Table 2.1. For this, we captured probe requests for all devices at the university's main hall. For every device, we calculated the average time between bursts of probe requests sent out. These values were then averaged per vendor. Note that we are only able to distinguish between vendors (as will be described in Section 5.3), and not between operating systems. While Apple and Blackberry devices will almost certainly be running respectively iOS and Blackberry OS[2], devices by other vendors may be running Android, Windows Phone or any of the other vendor-specific alternatives like Symbian (Nokia) or Bada (Samsung).

Note that the active scanning mechanism is still used when the smartphone is already connected to a network, either to find other access points with a better signal strength than the connected network, or to improve network performance when handoff happens between different access points [Lindqvist et al., 2009]. Smartphones also use probe requests to find unknown networks. In this case, a *broadcast probe request* is sent. This type of probe request does not contain an SSID, and invites all access points in range to respond with a probe response containing the network identifier of the access point's network. These broadcast probe requests can be considered as a device-initiated alternative to the beacons that are used in the passive scanning method. As a last remark, probe requests can also contain additional information in the form of Information Elements (IEs). We will discuss these in more detail

[2] At the time this experiment was performed, Research In Motion (the company behind Blackberry devices) did not yet offer any Android-powered smartphones.

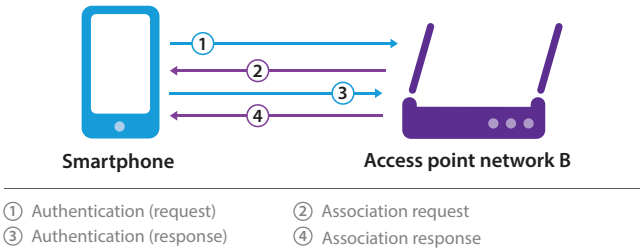


Figure 2.2: Connecting to a network: authentication and association.

in Chapter 4, but will note for now that major operating systems are currently in the process of phasing out these IEs in probe requests [Hogben, 2017].

2.3.2 Connecting to a network

Once a suitable network has been found, the mobile device (the *client* for this network) connects to it. This connection happens in two stages: an authentication and an association stage. The complete process is displayed in Figure 2.2.

The authentication stage allows devices to prove that they are authorized to connect to the network. If *shared authentication* is used by the wireless network, this stage involves the access point sending a challenge text to the client, which is then encrypted by the client to prove that it knows the key that is required to connect to the network. In modern networks, or networks that do not use authentication, shared authentication is not used[3], and the authentication process is deferred to a later stage. Instead, the network will use *open authentication*, exchanging only two authentication messages between the client and the access point.

In the association stage, the client sends an *association request* to the access point, informing it of its wireless capabilities[4] and its supported data rates and encryption protocols. The access point then responds with an *association response*, providing its capabilities to the client. Note that, although a client can execute the authorization stage with different access points at the same time (putting it in a *pre-associated* state), it can only be associated with one access point.

2.4 The WiFiPi detection mechanism

In this section, we lay out the structure of the mechanism used for tracking smartphone owners. Our system consists of a number of detectors, placed at different locations, with the

[3] The Wireless Equivalent Privacy (WEP) protocol, which is the only protocol that uses shared authentication, has been proven to have major security weaknesses, and its use has been discouraged [Fluhrer et al., 2001; Tews et al., 2007]. Modern wireless authentication protocols, such as the WPA2 industry standard, defer their authentication to a later stage.

[4] As noted in the previous section, we will provide some examples of these capabilities in Chapter 4.

aim of covering the entire (event) area. This allows for detecting smartphone users without their active cooperation. We first describe which elements of the 802.11 standard enable us to detect devices on a single location, and continue with an overview of how we use these detection techniques to track a device across different locations.

2.4.1 Detecting a device

In order to be able to detect a particular device that enters a detector's range, the detector needs to scan the ether for packets that have the following characteristics:

- In order to be able to uniquely identify a device, the captured packets should contain the hardware (MAC) address of the device's Wi-Fi interface.
- To make sure that all devices in range are detected, the packets must be sent out regularly by the device.

In addition to the probe request and association request frames that were described in the previous section, our system also detects *reassociation request* frames. A reassociation Request is sent when a Wi-Fi device wants to connect to another access point on the same network. This is the case, for example, when the Wi-Fi device is roaming and has detected an access point with a stronger signal serving the same network. Reassociation requests can even be triggered by any station present on the network. Indeed, by sending out a *disassociation frame*, a station is able to request all associated clients to disconnect from the network, effectively forcing them to reassociate afterwards. Moreover, disassociation frames can be easily forged by a third party [He and Mitchell, 2005], causing every connected device to reassociate.
None of these three types of packets are sent out continuously. We define a *detection round* to be the minimum time interval for which we can be reasonably sure that a device sends out at least one of these three types of packets. Based on our results in Section 2.3, and based on specific lab tests in which we singled out devices, the ideal duration of a detection round was empirically determined to be 130 seconds: each of the tested Wi-Fi devices (including multiple Android devices, notebooks, an iPhone, and an iPad) sends out a probe request at least once every two minutes, regardless of whether it was connected to a wireless network, and regardless of whether it had been in use.
We refrain from using a mechanism which analyzes every possible type of 802.11 packet because capturing and processing all packets would induce a great computational strain on the detectors, while adding little benefit. Indeed, processing every large data packet at the user space level instead of dropping it at the level of the network interface could very easily overload the low-cost, low-power detectors. Furthermore, to ensure completely transparent and non-obtrusive operation, our mechanism does not use disassociation frames to force reassociation of devices. If it is not required that the detectors operate stealthily, disassociation

frames can be used to force devices to reconnect to networks, essentially eliciting an association or reassociation request from the device (causing it to be easily detected). We published a separate paper (not part of this dissertation) that discusses other methods for instigating transmissions from mobile devices [Robyns et al., 2017].

Channel hopping techniques could be used to capture packets on all different 802.11 channels. While channel hopping would allow a detector to detect (re)association requests on all different channels, the fact that the radio is tuned into a single frequency band for only a short period at a time has a negative impact on the number of complete probe requests detected (probe requests are sent on all channels). Empirical tests have shown that the advantages of channel hopping do not outweigh its cost (i.e. fewer devices are detected in total when using channel hopping); it is therefore not used in the proposed solution.

2.4.2 Tracking the location of a device

Using the device detection method described above, a system which tracks the movement of smartphone users can be created by dispersing multiple detectors over the coverage area. The location of a specific device – and thus, its user – can then be determined by keeping track of the different times at which each detector detected a packet originating from the device's MAC address.

An optimal tracking setup considers the placement of the detectors as well as the range of the antennas to cover an area that is as large as possible. The latter can be tuned by opting for directional (beam-type) or omni-directional antennas with a specific gain factor. Detection ranges of individual detectors are allowed to overlap: both detectors could then be used to establish a more precise location of the detected device.

It is a requirement that the clocks on the different detectors are at least loosely synchronized, in order to be able to correlate data from the detectors afterwards. Alternatively, the detectors could have their logging information sent to a server immediately. The server could then be held responsible for correctly synchronizing the data from different detectors.

By correlating data from different detectors over time, a path can be established for every visitor. By taking into account physical properties of the tracked area, such as blocked paths and distance between detectors, more granular paths can be inferred. Moreover, in case of only one entrance and/or exit, it can be determined when a visitor enters or leaves the tracked area. We will use similar techniques in Section 3.3.2, when we will be using data gathered by tracking festival visitors to simulate visitor movement patterns.

Additionally, to further increase location determination precision, the RSSI value – which indicates the received signal strength for a received 802.11 packet – could in theory be used. From this value, an estimation could be made on the distance between the detector and the device sending out the packet. However, empirical tests have shown that the RSSI value is of little use in crowded environments containing a high amount of electronic devices and people due to severe fluctuations and noise in the data sets. Because of these environmental factors,

the RSSI value is currently not used in the detection mechanism. Rather, the overlapping range of the individual detectors provides a similar, but more consistent result. We use this to provide an estimate on the location of a tracked device.

2.5 Implementation

The detection mechanism as described above was implemented on a Raspberry Pi computer. The Raspberry Pi was the hardware of choice because of its low power requirements (3.5W), its small size, and its very low cost: a Raspberry Pi cost under US$35 at the time of our experiments (in 2012), while a current model Raspberry Pi Zero (released in 2015) can be bought for as low as US$5. A USB hub was attached to the Raspberry Pi for power and expansion, and a Wi-Fi dongle supporting monitor mode was used for capturing packets. An external antenna was attached to the Wi-Fi dongle to increase the detection radius. Either a cell phone or an ethernet cable was used for network communication, depending on the facilities at the tracking site.[5] Lastly, an LED was connected to the Raspberry Pi's GPIO pins to provide a status indicator, displaying whether or not the detector was a) scanning, b) connected, and c) aware of the correct local time. The result can be seen in Figure 2.3.
Since Raspberry Pi's do not have a real-time clock (RTC), the time for the Raspberry Pi is reset at boot time. For providing the Raspberry Pi with the correct time, we used `ntp` when a network connection was available. For situations in which no network connection was available, we created a custom-made Android application that could be installed on our own smartphones for informing the detectors in range about the current time. As we described in Section 2.4.1, we limit the detectors to process only probe requests for efficiency purposes. Because of this, the Android app (ab)uses the probe request mechanism to transmit this information to the detectors. It does this by using Android's `WifiManager` to temporarily create a Wi-Fi network with a specific SSID containing an identification string, as well as the current time. This causes the Android phone to broadcast a probe request with this information, which can then be picked up by our detectors.
The Raspberry Pi was running Raspbian, a Debian-based GNU/Linux distribution specifically tailored for use with the Raspberry Pi. The detection software was implemented in Python, making use of the `scapy` packet capturing and manipulation library [Biondi, 2005]. This library was chosen because it allows for filtering of packets at the kernel driver level by making use of the Berkeley Packet Filter (`bpf`), while still allowing for easy interaction with captured packets at the user space level. This combination ensures that the scanner software is as lightweight and speedy as possible, while still being able to extract useful information from packets on-line, at the scanner itself. This is important because the low-cost Raspberry Pi devices have both limited processing power and limited storage speeds, which requires

[5] Note that a network connection is not required for the detector to function correctly. A network connection was used to display a real-time overview of the gathered data and to provide a dashboard to the organization of the festival.

Figure 2.3: The Wi-Fi detector, consisting of: (a) a Raspberry Pi, (b) a Wi-Fi dongle, (c) an external long-range antenna, (d) a USB hub, (e) a Nokia N78 phone, and (f) a heartbeat LED.

a scanner implementation to write as little data as possible, while doing as little processing work as possible. Because of our optimizations, the detectors are able to process more than 4 000 detected Wi-Fi nodes per detection round in real time.

A list of detected devices was sent to a central server after every detection round in order to both provide a failsafe logging mechanism and to be able to gather statistics in real time. These real-time statistics include information such as the crowd density at different detectors, information about visitors' devices (manufacturer, broadcasted SSIDs), and spatio-temporal information about the visitors. An example of the dashboard displaying part of this information can be found in Figure 2.4.

2.6 Experiments

To investigate the feasibility of using the WiFiPi system at various types mass events, we perform experiments in different settings. We do this by looking at the number of devices that the system is able to consistently track, as well as the number of different data points we can gather per device to generate a movement path that is consistent with the actual movements of the device's user.

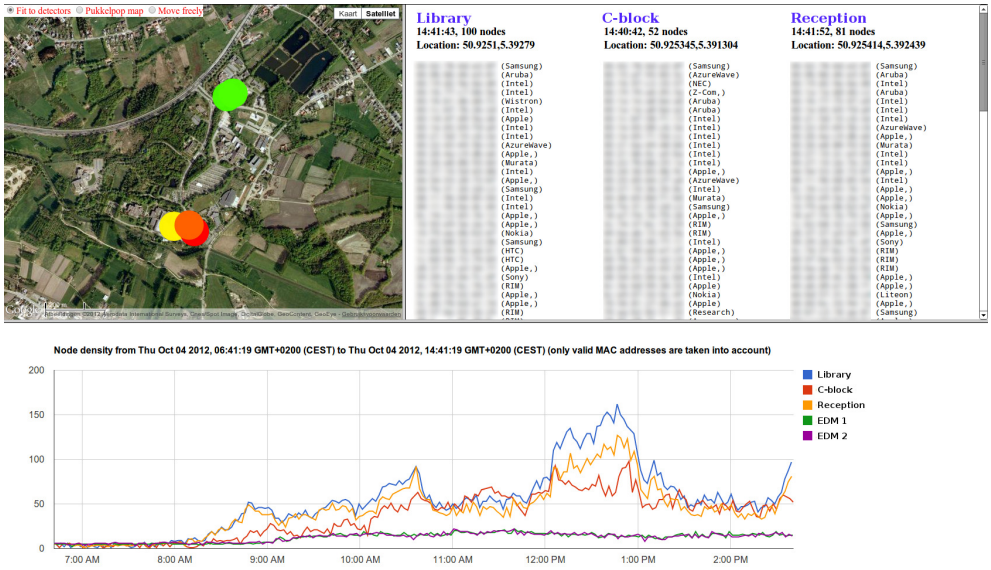


Figure 2.4: The dashboard showing information from the second scenario (university campus). Left: a heatmap showing the density around the different detectors; right: the MAC addresses of the currently detected devices (blurred out), together with the device manufacturers; bottom: a timeline showing the number of detected devices per detector over time.

Table 2.2: Detection statistics for three experiments using the WiFiPi system.

	Pukkelpop 2012	University campus	Pukkelpop 2013
Detection period	3 days	3 months	3 days
Visitors	100 000	3 200 (daily)	85 000
Detected devices	29 296	16 383.4 (daily)	40 815
Detection rate	29.3%	40%	45.7%
Detections per device	10.5	272.1	149.2

At the time of the study, we performed two experiments using the detector software. The first experiment consisted of placing the detectors at a three-day international music festival. For the second experiment, detectors were placed at the university campus, tracking visitors for over three months.

One year later (after publication of this work [Bonné et al., 2013b]), we performed the same experiment using an upgraded version of our detector software and hardware, at the next edition of the music festival. We added the results for this study in Section 2.6.3.

A summary of the detection statistics for all three experiments is available in Table 2.2.

2.6.1 Pukkelpop 2012

Pukkelpop is a Belgian music festival attracting 100 000 visitors[6] every year. The 2012 edition spanned three days, from August 16th to August 18th. During these days, fifteen detectors were placed at strategic locations, ranging from the 8 different stages to important passageways. Three detectors contained a cell phone for real-time monitoring. The total area of coverage was about 400m by 500m.
Combining the data from all fifteen detectors, a total of 137 899 unique devices (MAC addresses) were detected. These detections also include some devices not belonging to festival visitors. For example, some detections might have resulted from devices in passing cars or from devices that are part of the fixed infrastructure. For this reason, we included in the final dataset only those devices which were detected in at least two different locations, at different moments in time. Filtering out the devices conforming to this requirement, we find a total of 29 296 detected devices, giving us a relatively good estimate of the number of people carrying a Wi-Fi-enabled device at the festival (29.3%). The 29 296 devices account for a total of 307 256 data points, spread out over the three days of the festival.
It must be noted that the previous numbers establish a lower bound, not only because some Wi-Fi enabled devices might have had their Wi-Fi turned off for the duration of our experiments, or because it is expected that smartphone usage will increase over time, but also because technical difficulties occurred during the first experiment. Indeed, during this experiment, power failures were common, and both the detector software and the Raspberry Pi operating system still suffered from childhood diseases. These problems caused the detectors to sometimes malfunction, hindering them from detecting some devices. Another reason that these numbers establish a lower bound is our requirement that devices should be detected at at least two different locations. Because of this, rather stationary visitors are not part of the results.
Together with our own Wi-Fi tracking setup, we also deployed Versichele et al.'s Bluetooth tracking setup [Versichele et al., 2012]. As we will explain in Chapter 3, their method provided a coverage rate of approximately 10% of festival visitors, already being surpassed by our own method by a factor of 3 in 2012.

2.6.2 University campus

To demonstrate the versatility of the system, detectors were also placed at the Diepenbeek campus of Hasselt University (Universiteit Hasselt). Besides being an indoor location, this scenario also differed from the one above in the fact that more long term monitoring (3 months+) was done and because the coverage area was smaller with more overlap between detectors to increase accuracy. The Diepenbeek campus building of Hasselt University typically has around 3 200 daily visitors, consisting of students, staff and other guests.

[6]The Pukkelpop 2012 festival attracted 100 000 unique visitors over the course of three days. This number was provided to us by the Pukkelpop organization.

Figure 2.5: The second version of the Wi-Fi detector, installed at Pukkelpop 2013, also containing an LCD display.

For this experiment, four detectors, all of them provided with a network connection, were placed both at the Diepenbeek campus building and at the Expertise Centre for Digital Media (EDM, Hasselt University's multimedia research center). Two of the detectors were placed relatively close to each other. This way, it was possible to cross-check data from those detectors, allowing us to verify that devices were either tracked by both detectors or not detected at all. Over a period of three months, 16 486 devices passed within the range of at least one detector. In total, the devices were detected a total number of 4 486 310 times.
On average, 1 383.4 unique devices are detected at the main campus per working day. Assuming that every visitor carries exactly one switched-on Wi-Fi enabled device[7], it can be concluded that the detectors provide a coverage rate of around 40% for this scenario.

2.6.3 Pukkelpop 2013 and WiFiPi 2.0

In our music festival experiment from 2012, we collected information from smartphones of over 29 000 festival visitors (about 29.3%). To see if this number would increase over time, we repeated the experiment in 2013, with an updated version of both the detector software

[7]Note that a visitor may be carrying more than one switched on Wi-Fi enabled device (notebook, smartphone, tablet), or that he/she may not be carrying a smartphone at all.

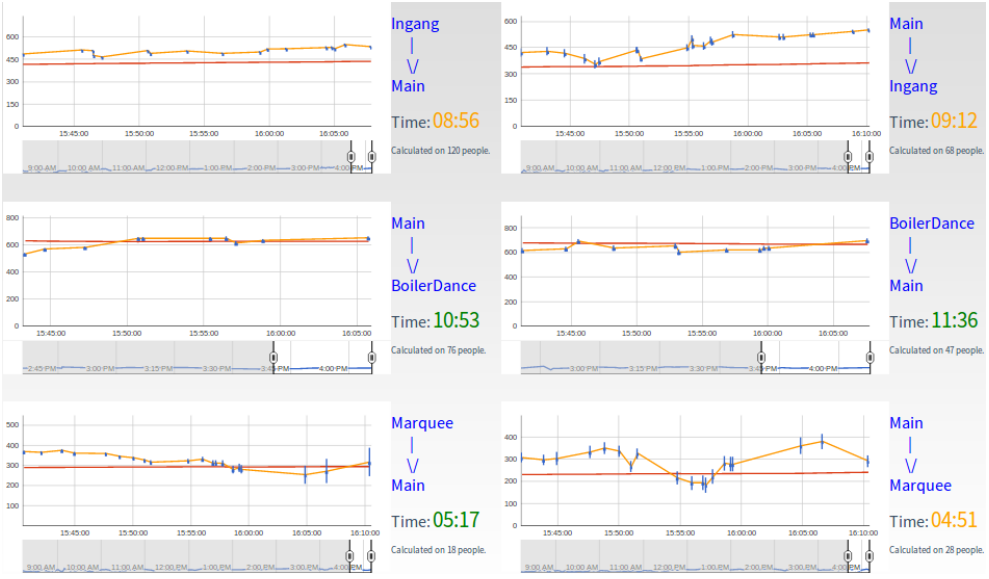
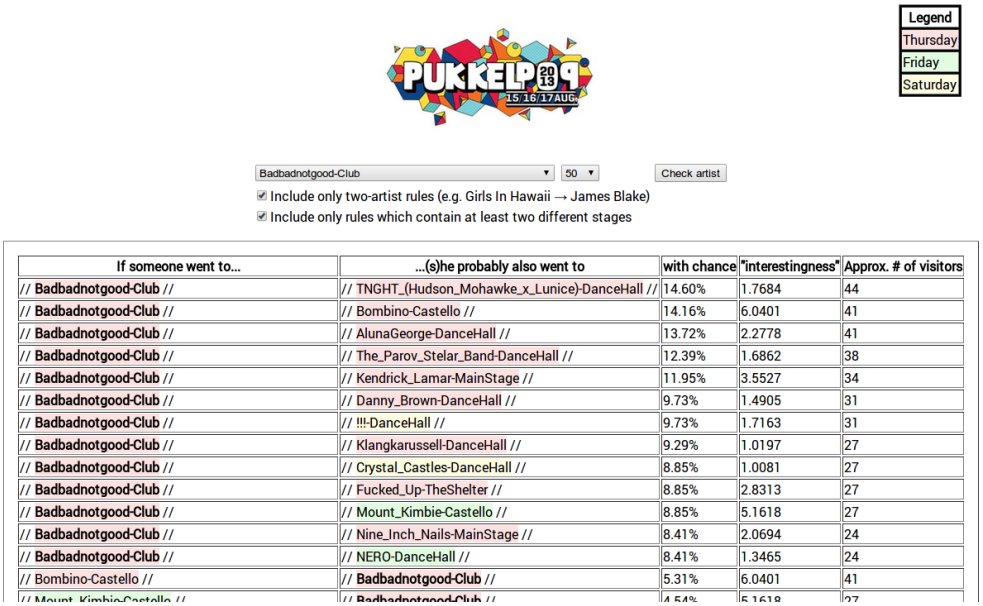


Figure 2.6: The updated version of the dashboard, providing a.o. real-time information about average travel times between the different stages for festival visitors.

and hardware. Among improvements in detection performance, the newer version of our detector (see Figure 2.5) also included an LCD display, providing us with basic information such as the network connection information, the detector's wall clock time and the number of detected devices, allowing us to easily monitor those detectors in real time even when an internet connection was not available.

Apart from the detectors, the dashboard was also updated (see Figure 2.6) to provide additional information. Among others, it provided the festival organization with real-time information about the travel times between different stages, an updated real-time heatmap and different spatio-temporal visualisations of the distribution of visitors between different stages and different points in time.

This repeated experiment provided us with smartphone detections for 38 828 festival visitors (about 48% of the amount of tickets sold). Based on anonymized data we could again infer which music stages a person visited at what time. We were able to cluster visitors based on their musical preferences, and we generated association rules that could be used to infer which artists a visitor was most likely to visit based on other attended performances. An example of this is shown in Figure 2.7, where association rules for the band "BadBadNotGood" are shown. From this figure, we can see that a visitor in our dataset that attended BadBadNotGood had a chance of 14.6% of also attending the band TNGHT, and that approximately 64 visitors attended both bands. "interestingness" is used as a term to describe the statistical



If someone went to...	...(s)he probably also went to	with chance	"interestingness"	Approx. # of visitors
// **Badbadnotgood-Club** //	// TNGHT_(Hudson_Mohawke_x_Lunice)-DanceHall //	14.60%	1.7684	44
// **Badbadnotgood-Club** //	// Bombino-Castello //	14.16%	6.0401	41
// **Badbadnotgood-Club** //	// AlunaGeorge-DanceHall //	13.72%	2.2778	41
// **Badbadnotgood-Club** //	// The_Parov_Stelar_Band-DanceHall //	12.39%	1.6862	38
// **Badbadnotgood-Club** //	// Kendrick_Lamar-MainStage //	11.95%	3.5527	34
// **Badbadnotgood-Club** //	// Danny_Brown-DanceHall //	9.73%	1.4905	31
// **Badbadnotgood-Club** //	// !!!-DanceHall //	9.73%	1.7163	31
// **Badbadnotgood-Club** //	// Klangkarussell-DanceHall //	9.29%	1.0197	27
// **Badbadnotgood-Club** //	// Crystal_Castles-DanceHall //	8.85%	1.0081	27
// **Badbadnotgood-Club** //	// Fucked_Up-TheShelter //	8.85%	2.8313	27
// **Badbadnotgood-Club** //	// Mount_Kimbie-Castello //	8.85%	5.1618	27
// **Badbadnotgood-Club** //	// Nine_Inch_Nails-MainStage //	8.41%	2.0694	24
// **Badbadnotgood-Club** //	// NERO-DanceHall //	8.41%	1.3465	24
// Bombino-Castello //	// **Badbadnotgood-Club** //	5.31%	6.0401	41
// Mount_Kimbie-Castello //	// **Badbadnotgood-Club** //	4.54%	5.1618	27

Figure 2.7: Association rules generated for Pukkelpop 2013, showing the correlation between visitors of different artists. This screenshot shows association rules for the band "BadBadNotGood" (which played at the "Club" stage).

concept of lift, and is calculated as follows:

$$\text{lift}(ArtistA \Rightarrow ArtistB) = \frac{P(ArtistB \mid ArtistA)}{P(ArtistB)}$$

The intuition behind showing the statistical lift as an "interestingness" metric to the festival organizers is that it scales the probability of a specific rule to the probability that a random festival visitor would visit the artists in the second column. This gives rules with less popular artists in the second column a higher "interestingness" value. Intuitively, artists that were more likely to have been visited by any festival visitors (as opposed to only the ones who visited the artists in the first column) correspond to a less interesting case than artists who have been visited by only a small part of overall festival attendees.

The example of BadBadNotGood attendees also attending TNGHT given above in particular was interesting to festival organizers, as BadBadNotGood and TNGHT are very different bands: whereas BadBadNotGood plays jazz that is influenced by hip hop, TNGHT is known for its electronic trap music. What caused these bands to be visited by the same audience is that BadBadNotGood, programmed early on in the day, played a cover version of one of TNGHT's songs, encouraging the audience to attend the concert of the latter later that day.

In our repeated experiment, we did not limit the WiFiPi system to collect only MAC addresses, but also used it to collect lists of the networks that users had connected to in the

past (as explained in Section 2.3). This led to some interesting results: of the 38 828 devices that we detected during this event, 23 103 devices (59.50%) sent out at least one network identifier (SSID). Moreover, 6 609 of these devices (28.61%) broadcasted at least one network that was not broadcasted by any of the other devices, and more than 50% of the devices contained an SSID that was shared among at most 50 other devices. This means nearly a quarter of the people carrying a smartphone could be uniquely identified only by the list of networks in their phone, and that clustering people by workplace or home based on the gathered data is possible. We were able to achieve these results with a simple and low cost setup, which underscores the fact that anyone with limited resources could collect similar data at any particular location. In Chapter 5, we will present a system (called SASQUATCH) that uses this information in an experiment that aims to both assess and increase user awareness about privacy issues in wireless networks.
As we will see in Chapter 3, the detection rates that the WiFiPi system is able to achieve are sufficient for building realistic movement paths of visitors at mass events, both in the context of a music festival as in the more low-traffic context of a university campus.

2.7 Applications

Using low-cost hardware for tracking people offers a wide variety of applications, of which we give some examples in this section. We emphasize that this is only a small subset of many possible use cases.

2.7.1 Real-time crowd management and marketing

An interesting application for organizers of mass events is to use the real-time gathered data for crowd control. Similar to the dashboard (pictured in Figures 2.4, 2.6 and 2.7) that was used for visualizing crowd data during our experiments in real time, one could visualize the real-time data in a way that shows the flows of people moving, and provides additional information on crowd density compared to the maximum capacity in a particular location. Moreover, a visualisation of real-time data could prove to be vital in an evacuation scenario where the goal is to get people to move to – or away from – a certain location as fast as possible.
Furthermore, the data can be used after-the-fact to gather some interesting statistics about visitor behavior at music festivals, for example:

- Which artists or stages are most popular and which artists attract a similar audience?
- Which (unforeseen) crowd movements happen on the terrain over the duration of the festival (due to unplanned events or scheduling issues between the stages)?
- How stationary are visitors? How much time do they actually spend on the festival site?

As an example, we provided some analysis on the first point (correlating different artists based on their visitors, as described in Section 2.6.3), and showed the results to the festival organization. They confirmed the most important trends, and indicated to be interested in a commercial system providing this information.

This gathering of statistics does not only apply to music festivals, but also to places like shopping centers. There, the data could be used, for example, to monitor the shops in which customers spend most of their time, or to oversee queueing times at the cash registers.

After publication of this work [Bonné et al., 2013b], different marketing agencies and cities like London and New York have started adopting similar techniques for tracking their visitors and inhabitants. One example is marketing agency "Renew", which used trash cans in London to track people via their smartphones' Wi-Fi signals. Another example is the company "Navizon" (now Accuware), specializing in tracking devices through a variety of signals [Accuware, 2017].

2.7.2 Mobility models for simulations

Opportunistic, multi-hop networks deal with using ad hoc communication to provide network connectivity among different devices in a local area. A possible application for this technology is a mass event, where conventional cellular networks are likely to become overloaded due to the high amount of visitors.

In simulating opportunistic networks, it is essential that the simulation runs are performed using realistic movement patterns [Aschenbruck et al., 2011; Camp et al., 2002]. Because of this, real mobility data of visitors at a mass event can provide invaluable information for creating a realistic simulation. The dataset acquired at the Pukkelpop festival was converted to two different types of mobility traces: one that could be used by the ONE opportunistic network simulator [Keränen et al., 2009], and one that could be directly used in simulations run by either ns-2 or ns-3 [Henderson et al., 2006].

In Section 3, we will discuss this specific application of involuntary tracking, using data gathered in a similar fashion to optimize routing protocols that can be used for opportunistic communication between music festival visitors.

2.7.3 Ubiquitous computing

In the domain of Ubiquitous Computing, a low-cost Wi-Fi detector could be used to infer which people are currently present within a certain room, and to tune the atmosphere accordingly. A visitor to a room can have preferences for certain types of lighting or genres of music. Furthermore, user interfaces can be tweaked to accommodate a user's preferences, or the user could automatically be logged in to certain services. It must be noted that the maximum detection interval of 130 seconds may be too high in such a scenario. In this case, techniques such as disassociation requests (see Section 2.4.1 and [Robyns et al., 2017]) could

be used to speed up the detection of a device as a wireless infrastructure using access points is likely to be present.
A Wi-Fi detector could also be used to save energy in rooms where no one is present, possibly enhancing other detection methods such as motion sensors, audio detection or security cameras. Assuming that every visitor carries a smartphone, if it is detected that all devices that were previously present have now left the room, lights and other appliances could be turned off automatically. This could increase the accuracy of detection especially when attendees are mostly static.
Lastly, as described by Rose and Welsh [Rose and Welsh, 2010], information from probe requests can be used to infer past location data from users, allowing for user profiling. This user profiling could aid, for example, in automatic language selection, choosing the user's language based on the locations over the world he/she has visited most often.

2.8 Privacy implications

Clearly, tracking people via their smartphones brings about important privacy implications. The most obvious one is that when the MAC address of a person's smartphone is known, it is easy to reconstruct the complete path this person has traveled.
Because MAC address information needs to be shared among different detectors for tracking purposes, it is not possible to anonymize our dataset simply by associating a unique identifier to every MAC address in each individual detector.
A naive solution might consist of creating a one-way hash of every MAC, in order to obfuscate the MAC addresses, while still making sure that the identifier would remain the same over different detectors. However, it would then still be possible to track a certain MAC address by calculating its one-way hash. Thus, care must be taken that the data is anonymized in some other way (e.g. by associating a random number with every MAC address) *after* it has been combined from all individual detectors.
However, even if the MAC address of a person's smartphone is not known, it is still possible to derive information from the captured Wi-Fi probe requests alone. For example:

- The list of known SSIDs is available as part of the probe request. From this list we can derive other networks the user has connected to, which may include e.g. the SSID of the home network, the SSID of places visited, or even the user's personal name (as part of the SSID of the user's home network or mobile hotspot).

- The manufacturer of a person's smartphone, which can be derived from the first part of a smartphone's MAC address, can be used to identify that person. To illustrate how, consider the scenario where the visiting times for a specific person at certain places are known. The list of devices detected at that time can then be reduced to a list of devices matching that person's smartphone manufacturer, which makes it easy to derive the MAC address of the smartphone.

Indeed, we will use both of these points of information in our experiments to increase user awareness in Chapter 5.
Moreover, de-anonymization of the dataset is possible when some other information is known. Indeed, relationship graphs in social networks can be compared to devices traveling together amongst different detectors in order to infer real-world identities from the mobility dataset. This is analogous to previous work done by Narayanan et al., wherein original identities are derived from anonymized social network graphs [Narayanan and Shmatikov, 2009]. Moreover, Bilogrevic et al. showed that an adversary having access to a network of wireless sniffing stations (similar to our detectors) is able to reconstruct social communities based on gathered MAC addresses of participants' mobile devices [Bilogrevic et al., 2012].
As in the work of Rose and Welsh [Rose and Welsh, 2010], the list of SSID's collected from a smartphone can also be used to infer the coordinates of past locations of a smartphone user. As we will discuss in more detail in Part II (more specifically, in Sections 5.2 and 5.3.1), this can be achieved by using the list of SSID's as an input when querying a "wardriving database" such as WiGLE [WiGLE.net, 2016], which contains a list of access points and their real-world locations.

2.9 Conclusion

We have shown that tracking of visitors at mass events can be achieved at a very low cost and – more importantly – unobtrusively and without requiring active cooperation. We achieve this by implementing the scanning software in a way that limits both processing and disk I/O, allowing for detection of up to 4 000 devices per 130 seconds (a 'detection round'). The proposed method can easily be tailored to suit various contexts of use, demonstrated by the two scenarios presented. A number of possible applications have been discussed, along with some privacy implications that should be kept in mind when using the proposed solution. As we noted in Section 2.7, marketing agencies and public institutions have already started adopting similar techniques after the publication of this work [Bonné et al., 2013b].
The privacy implications that this technique entails, combined with the very low cost and ease with which this tracking can be peformed, show the large risk that any third party can abuse this system for nefarious purposes. Indeed, even if marketing agencies and public institutions comply with ethical and legal standards, nothing prevents a malicious actor from using the same techniques to gather information or mount attacks. This ties into research question **RQ1**, showing that the default modus operandi of wireless network already entails important privacy and security issues. We will show how these vulnerabilities can be exploited in more detail in Chapter 5.

The data acquired at the Pukkelpop festival is part of a project in which we aimed to develop a smartphone application which allows opportunistic (ad hoc) communication at mass events such as music festivals as an add-on to infrastructure-based networks. The gathered data

can be used in network simulation experiments, in which the optimal opportunistic routing protocol for use at mass events is determined. We will discuss how these kinds of simulations work in the next chapter.

Chapter 3

A Comparative Simulation of Opportunistic Routing Protocols Using Realistic Mobility Data Obtained From Mass Events

3.1 Introduction 35
3.2 Background and Related Work 35
3.3 Mobility 36
3.3.1 Collecting Data 37
3.3.2 Calculating movement paths 37
3.4 Simulations 38
3.4.1 Simulation Settings 39
3.4.2 Routing Protocols 41
3.5 Results and analysis 42
3.5.1 Metrics 42
3.5.2 Individual Protocol Examination 44
3.5.3 Comparison 45
3.5.4 Candidate Selection 46
3.6 Conclusion 48

3.1 Introduction

In the previous chapter, we showed how smartphone users can be surreptitiously tracked by third parties through Wi-Fi signals that are sent out by their smartphones. In this chapter, we show a legitimate use case for data that is gathered in this fashion. Specifically, we show that mobility data gathered by an opportunistic tracking system at a mass event can be used in simulations that try to emulate these types of events. We again use the "music festival" mass event use case as an example, using the gathered mobility data to optimize and adapt an opportunistic routing protocol that could be used in applications providing ad-hoc communications to festival visitors.

To see why ad-hoc communications at music festivals are an interesting use case for mobility simulations, consider the fact that the existing cellular network infrastructure at mass events is often faced with very large amounts of data. This can result in network outage or serious delays, especially in places where many people gather in a small geographical area. Thus, it would benefit festival attendees if they could exchange small messages (e.g. SMS or WhatsApp) through ad hoc communication at these events, reducing the load and freeing capacity on traditional cellular networks. For this, opportunistic network technology can be leveraged to directly deliver data between devices.

Crowds at mass events follow specific movement patterns, the properties of which can be exploited to better route information between users. In literature, several routing protocols have been proposed that can be used to implement such an ad hoc messaging application on mobile devices. All of these protocols require different parameters to be tuned. These choices range from how long one should wait for the arrival of a message to the number of messages that should be passed on to a neighboring device. Simulation is a common tool in research on computer networks, especially in Mobile Ad Hoc Network (MANET) and Delay Tolerant Network (DTN) research. In this chapter, we use mobility data which was captured using a technique similar to the one described in the previous chapter to perform simulations of these various routing protocols using the Opportunistic Network Environment (ONE) Simulator [Keränen and Ott, 2007], and present our results. Furthermore, by analysing the data, a suggestion for best-suited protocols will be made for an actual implementation.

3.2 Background and Related Work

In the last decade of DTN research the main focus has been on the development of efficient routing protocols. Several surveys can be found in literature [Zhang, 2006; Khabbaz et al., 2012]. A commonly used tool to evaluate these protocols is simulation. There are several available network simulators, for example ns-2 (and its successor ns-3) [Henderson et al., 2006] or OMNeT++ [Varga and Hornig, 2008], each with its own benefits and drawbacks. The Opportunistic Networking Environment (ONE) simulator presented by Keränen

and Ott [Keränen and Ott, 2007], is specifically designed to facilitate DTN related simulations and is used and evaluated in several publications.
Although simulations are a commonplace tool in research, there are typical mistakes and misconceptions. Andel and Yasinsac [Andel and Yasinsac, 2006] point out several of these in the reporting of simulation studies. More specifically, Grasic and Lindgren [Grasic and Lindgren, 2012] have surveyed a number of DTN related papers published in the last decade and describe several issues that should be avoided in future research, with the emphasis on using the correct node density and mobility models for a given scenario. This shows the benefit of using real mobility data.
Aschenbruck et al. [Aschenbruck et al., 2011] provide an overview of the available synthetic and trace-based mobility models that can be used in simulations and emphasize the need for more realistic traces and mobility models. The CRAWDAD database [Yeo et al., 2006a] is a well known repository in the domain and contains diverse wireless trace data sets which can be used to extract mobility information. Numerous publications using those traces to improve synthetic models can be found, for example Lee et al. [Lee et al., 2009] who developed a movement model based on GPS traces. Other publications use the aforementioned traces to evaluate a protocol, like the work of Radenkovic and Grundy [Radenkovic and Grundy, 2011] wherein several traces are used to evaluate a proposed routing strategy.
We have used a similar method to the one presented in the previous chapter [Versichele et al., 2012] to gather a large set of mobility traces at the same international music festival, and have used this data to simulate and investigate existing DTN protocols in a mass event setting without the need to depend on synthetic models. The difference between the method laid out in the previous chapter and the one presented by Versichele et al. [Versichele et al., 2012] is that whereas our method uses Wi-Fi signals to track visitors, their method tracks devices that are in the Bluetooth discoverable state. We compare both methods in Section 2.2. Both methods were deployed at the same time, with our own method as a proof-of-concept, and their method for the actual data gathering. Vukadinovic and Mangold [Vukadinovic and Mangold, 2011] performed an analogous experiment in an entertainment park where GPS traces were collected using mobile devices that were randomly distributed to visitors over five days. However, they did not focus on different routing protocols using these traces.

3.3 Mobility

Simulating the movement of crowds at mass events in a realistic manner relies on the ability to precisely tune the simulation parameters to the context of the event. Traditionally, synthetic movement models – such as the random waypoint model or the city section model – have been used for this purpose [Camp et al., 2002]. Recent studies have shown that using synthetic movement models provides only a limited approximation of real trajectories, and that more realistic simulations can be achieved by using movement data gathered by tracking people within the desired simulation context [Aschenbruck et al., 2011]. A problem often en-

countered with methods for gathering this type of information is that only a relatively small subset of the population can be tracked at the same time [Versichele et al., 2012].
The chosen mobility model has a profound impact on the realism and thus outcome of a simulation. As Grasic and Lindgren [Grasic and Lindgren, 2012] have pointed out, many protocols in the literature are tested and evaluated in unrealistic or irrelevant scenarios. This chapter focuses on a specific real-world scenario and the development of applications for it.

3.3.1 Collecting Data

Despite projects like CRAWDAD, the public availability of wireless traces is rather limited. For this study, a subsample of festival attendees was tracked using the method of Versichele et al [Versichele et al., 2012].[1] By deploying 15 Bluetooth scanners at strategic locations over the festival terrain, proximity-based trajectories of over 10 000 unique devices were registered during the same festival (Pukkelpop 2012) as where our setup from the previous chapter was deployed. The unique hardware addresses (MAC) of Bluetooth chips make it possible to detect and track the movement of a device when combining data from several detectors in much the same way as the Wi-Fi technique. The hardware address can also be used to couple a detected device to a node that is used in a simulation environment. The resulting data contains timestamps of detected devices collected at certain strategically chosen locations on the terrain, such as stages and passageways, while trying to cover as much of the terrain as possible.

3.3.2 Calculating movement paths

As the collected data only includes discrete detection points in time, we need a way to generate movement paths from this data. We do this by letting a simulated node travel from a certain discrete detection location corresponding to a real node and time to the location and time of the next detection of that node. The range of the detectors, depending on the sensor used, is approximately 20-30 meters. Nodes in the simulation are therefore placed at or move towards a random position around the sensor locations in a radius of 20 meters.
If the nodes would simply travel in straight lines from one detector to another, certain obstacles that were present at the festival area would be ignored. To improve realism, a rudimentary shortest path finding algorithm was used to let nodes travel a more realistic path from one point to the next: paths are only created along passageways, since fences and tents on the terrain can not be crossed. To achieve this, additional points were added along paths in order to avoid these obstacles.
In between detections it is possible that certain devices remain undetected for a while. This does not necessarily mean these users left the festival area or turned off their devices: they could simply be out of range of all detectors. The problem is that there is no other information

[1] See Sections 3.2 and 2.2 for a comparison between our method from the previous chapter and the method by Versichele et al.

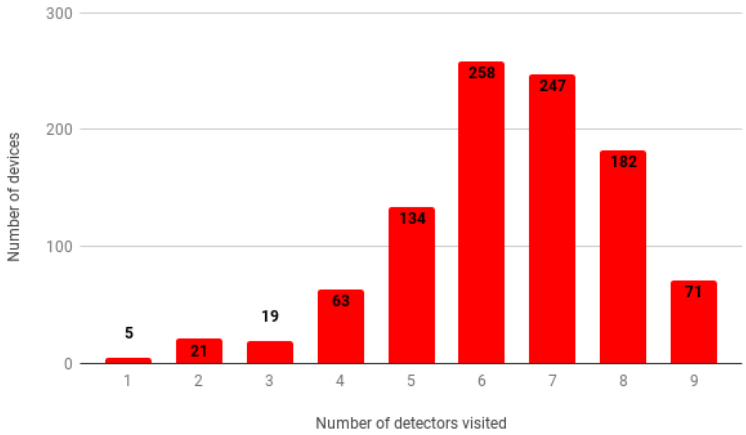


Figure 3.1: Number of detectors encountered by each of the top 1 000 nodes for a single festival day.

about the actual status of the device. The chosen solution to this problem is to disable the node in the simulator after a certain time of not being detected (we will explain how in Section 3.4.1). This time was empirically chosen to be 15 minutes: most of the festival area is covered and it should take less than 15 minutes to move between any two detectors. The node will remain disabled until a detection at a later timestamp is encountered in the data.
The Bluetooth detectors discovered an average of 5 300 unique devices per festival day. This number also includes devices that are only detected a few times (e.g. a passer-by) or are originating from organisational equipment which is static. A set of 1 000 nodes with the best paths (i.e. paths with the most detection points) were selected for our simulations. We perform the simulations with subsets of 50, 100, 250, 500 and 1 000 nodes, selected for 12 hours of a single festival day. This gives us an average of 202.18 detections per node, with an average of 6.40 (out of a maximum of 15) detectors visited for each node. The selected nodes moved from one detector to a *different* detector on the festival area 38.66 times on average (median: 32). A full distribution of encountered detectors, and number of movements per node is available in Figures 3.1 and 3.2, respectively.

3.4 Simulations

The main goal of all evaluated protocols is to enable communication between arbitrary devices in an ad hoc manner. This means that messages must be relayed by other devices to reach destinations that are not in transmission range. This section briefly explains the settings that were used for the simulations, followed by a description of the evaluated protocols.

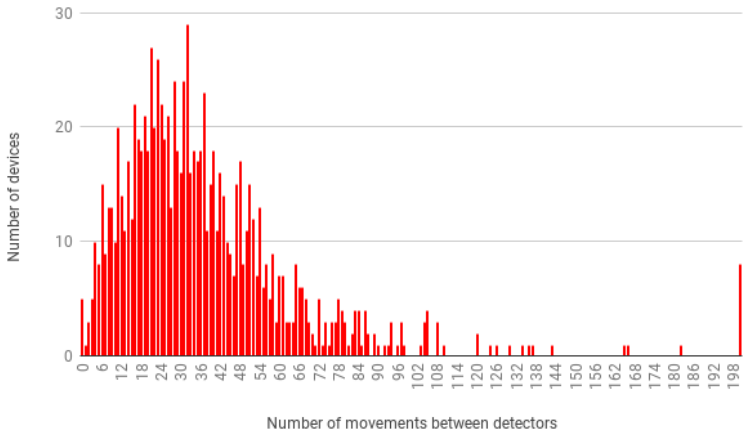


Figure 3.2: Number of times a node moved from one detector to a *different* detector on the festival area, for each of the top 1 000 nodes for a single festival day. The last bar represents devices with numbers ranging from 200 to 527 movements between detectors.

3.4.1 Simulation Settings

The ONE simulator is a network simulator developed to evaluate network protocols in a delay-tolerant environment. It is specifically designed to facilitate delay tolerant networking (DTN) research by abstracting certain elements of typical DTN communication and offering a number of tools for simulation. Visualization is one of these tools; Figure 3.3 shows a screenshot of the visualization component while simulating moving nodes on the festival area. Another tool enables researchers to import mobility data into the simulator. The source code of this component had to be slightly altered so nodes could be disabled when no trace data was available (as explained in section 3.3.2). Messages in the buffers of these disabled nodes are kept there but do age; depending on the routing protocol, it is possible they are dropped from the buffer due to aging as would be the case when devices have left the festival site in a real-world scenario. Each simulation run covers a full festival day (12 hours).

The size of the festival area is circa 400m by 500m. For all nodes in the simulations a communication range of ten meters was chosen using a Bluetooth interface with a transfer rate of 2 Mbit/s. Although many wireless technologies have a much larger range in theory, interference in a dense crowd and other obstacles often hinder communication over a large distance. The ONE simulator does not take into account these interferences; a relatively small transmission range is chosen to compensate for this deficiency, so the results are also relevant for other wireless technologies (e.g. Wi-Fi) that can be used in a real-world application.

The messages in the simulations have a size of 100 kB and the buffers in each node have a capacity of 10 MB. This is an estimation of the requirements of an application on a mo-

Table 3.1: Simulation settings.

Simulation time	One festival day (12 hours)
Simulation area size	400m x 500m
Transmission rate	2 Mbit/s
Transmission range	10m
Transmission protocol	Bluetooth
Message rate	4 messages per hour, per node, on average
Message size	100 kB
Buffer size	10 MB
TTL	15 minutes

Figure 3.3: Simulation of a festival day: the nodes and their transmission ranges are visualized by a circle, the background shows the festival area.

bile device for sending small text messages or pictures like in SMS/MMS. The size of the messages is large enough to support additional security measures such as the use of public-key cryptography. The buffer size is intentionally kept small (a maximum of 100 messages): more messages lead to more processing and transmissions and thus more energy consump-

tion. People who do not return home between festival days have to consider their energy consumption since they often have no way of recharging their devices. This factor should be taken into account when developing and testing routing protocols for scenarios like a festival spanning several days.

Messages are generated by random active nodes in the festival area. A node is assumed to be active when the simulator is certain of the node's location, i.e. when there is a related detection in the real-world trace for that particular node. The destination is also a random node in the festival area. Because nodes can become disabled, it is possible that the associated messages will disappear as well. This is not unrealistic since people may turn off their devices, the application may be shut down or because batteries could be depleted in real life. Using a random event generator in our simulations, every node sends an average of four messages per hour in a simulated festival day.

The time-to-live (TTL) of the messages was set to 15 minutes. This is rather a long time to mimic SMS-like behavior but it ensures that all protocols have a reasonable amount of time to deliver a message. In an actual opportunistic mobile messaging application this TTL should be further reduced since an acknowledgement system should inform the user whether a sent message has arrived at its destination, and long waiting times for this feedback are impractical. An overview of the simulation settings is given in Table 3.1.

3.4.2 Routing Protocols

Six routing protocols were tested in the simulations. Every protocol was run through five iterations of simulation, each iteration simulating a full festival day of 12 hours, with the results being averaged. In each of the five runs a different seed was used for the random number generator responsible for the time of creation, sender and destination of the messages. The protocols used were Direct Delivery (DD), First Contact (FC), Epidemic (EP) [Vahdat and Becker, 2000], Spray and Wait (SW and SWb) [Spyropoulos et al., 2005] and two versions of the Probabilistic Routing Protocol (PRoPHET): the original protocol (PR) [Lindgren et al., 2003] and a slightly adjusted version PRoPHETv2 (PR2) [Grasic et al., 2011]. The implementations of these protocols were made by the developers of the ONE simulator and other contributors. The active community and diverse research contributions have checked and optimized these implementations. Since the goal was to investigate general properties of routing algorithms in a specific environment, the choice was made to use specifically these thoroughly tested and reviewed protocols. A short description of each is provided below.

The first three protocols do not have parameters that can be tuned, so there is only one version to be tested. *DD* delivers the message only directly to the destination, i.e. there is no relaying, and messages are only delivered when the destination is a direct neighbor of the sender. In *FC* only one copy of a message exists at a certain point in time: a message is relayed to the first encountered node unless the message already traversed that node, and is then removed from the message buffer of the previous hop. This happens until the destination is reached or

the TTL is reached. The *EP* protocol replicates messages using a flooding approach so that all messages are constantly being sent to all nodes that are in range.

In *SW*, a maximum of n copies of a message can exist in the network. These copies are 'sprayed' towards encountered nodes, until only one copy remains at the sender. Two implementations of *SW* were used: binary (SWb) and non-binary (SW). In non-binary mode the sender sprays only one copy of the message to each (different) encountered node. In binary mode half of the available copies is passed on to an encountered node, which in turn does the same with these received copies. This happens until only one copy is left. The remaining copy then waits, in both binary and non-binary, to be delivered to the destination as soon as it comes into range. Both the binary and non-binary versions were simulated with different numbers of copies ($n = 6, 12, 18, 36, 72, 144$), based on values proposed in the original work and further empirically determined by preliminary simulations.

The *PRoPHET* protocols try to estimate the chance that a node can deliver the message to another node, based on historical information, and spread messages based on this chance. The transitivity property describes the probability by which a node can deliver a copy to the destination recursively. For example, if node A wants to send a message to node D via B, the transitivity property specifies the probability that B will encounter a node C which can in turn deliver the copy to D. *PR2* uses a slightly adjusted transitivity property and aging of the delivery probabilities. More details can be found in the respective papers [Lindgren et al., 2003; Grasic et al., 2011]. PR and PR2 ran with different values for the transitivity property ($\beta = 0.0, 0.09, 0.25, 0.90$) based on the proposed values in the original and on previous work.

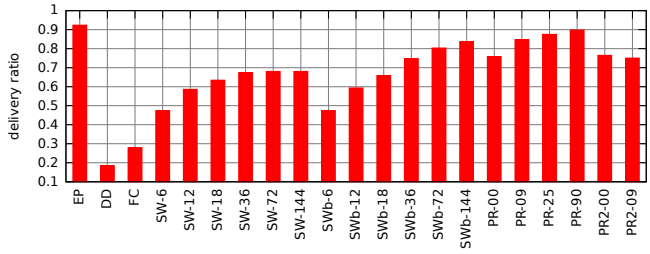
3.5 Results and analysis

This section discusses the results and analysis of the simulations described in Section 3.4. First, the proposed metrics are briefly described. The importance of these metrics is dependent on the application scenario and will be detailed in the analysis that follows. Finally, the most suitable protocols are suggested based on the simulations. The figures in this section show the results of the simulation runs with 500 nodes.

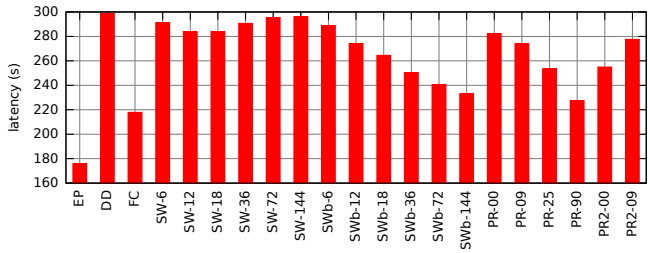
3.5.1 Metrics

1. Delivery ratio: This metric represents the percentage of messages that were created and effectively reached their destination.

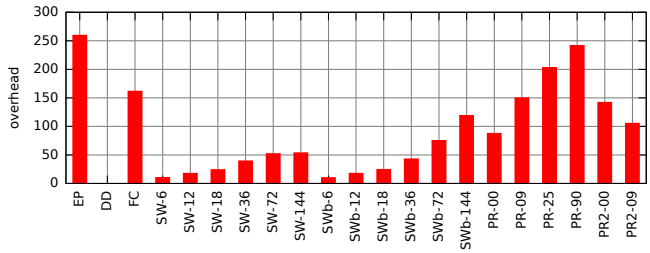
2. Latency: The time it takes for a delivered message to reach its destination. An upper bound on the time a message has to arrive at the destination may be needed in an application to decide whether the message is lost or a resend may be beneficial.



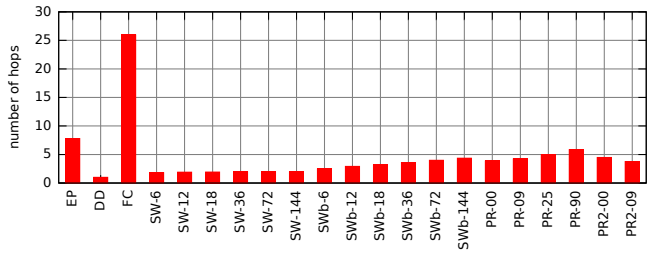
(a) Delivery ratio



(b) Latency



(c) Overhead



(d) Number of hops

Figure 3.4: Comparison between routing protocols based on the indicated metric.

3. Overhead: The overhead is defined as $\frac{\#relayed - \#delivered}{\#delivered}$. It provides an indication of the overhead as transmissions of messages that could not reach their destination are also included.

4. Number of hops: This number describes how many other nodes the message had to pass through to reach its final destination. This metric is often omitted or forgotten in the analysis or reporting of simulations but can indicate how challenging the evaluating scenario was and how well the routing protocol utilizes the network resources [Grasic and Lindgren, 2012].

3.5.2 Individual Protocol Examination

In all of the above protocols, and more in general in the simulator used, control messages are not included in the overhead calculation. The nodes can retrieve certain information, like the history of encountered nodes in the PRoPHET algorithms, without sending actual messages. It is possible to abstract this kind of communication as is the case in the simulator, for instance by assuming the lower layers handle this, but it should not be ignored in an analysis as these messages require additional energy for processing and transmissions and can even cause additional interference.

The results of the simulations are summarized in Figure 3.4. What immediately stands out is the high delivery ratio of the Epidemic routing protocol. However, because of the flooding nature of this protocol, there is a relatively large overhead. It is intuitively clear that when more copies of the same message are sent, the chance of actually reaching the destination increases. The low latency of EP in this figure shows that the underlying mobility trace with 500 nodes (and similarly the 1 000 trace) offers a beneficial environment for this protocol as there is a lot of movement between the groups of people at the festival site and the crowd is almost entirely interconnected.

Another part of the charts that warrants attention is the relatively large amount of hops of the First Contact protocol. The high hop count is caused by the choice of a random neighbor to relay the message, even if the destination is itself a neighbor of the sender. As can be seen on the latency graph, FC also has a relatively low end-to-end latency. The reason for this is that there is only a slight chance that the destination is reached, but if it is reached the coincidentally chosen path will be relatively fast.

The Direct Delivery protocol is not applicable for use within a real-world ad hoc application, since the point is to communicate with each other without being geographically close to each other. It is however useful to check a borderline case of DTN protocols, namely where there is no relaying at all. It is obvious that carrying messages for other nodes is necessary: the delivery property of DD is the lowest of all results. This also means that the probability is relatively small that two random nodes come in contact within 15 minutes (the chosen TTL) in the used mobility trace.

The Spray and Wait protocols try to reduce the amount of data in the network by imposing a maximum number of copies of a certain message. It is again clear that when more copies are allowed, the delivery ratio increases. Since the non-binary version spreads a message by giving one copy to each encountered node until only a single copy remains at the sender, the number of hops is always one (if the sender delivers the message) or two (if the contact delivers it). In the binary version, the maximum number of hops increases with the number of allowed copies because encountered nodes may receive more than one copy, and they can subsequently spread all of these copies. As can be seen in the graphs, the delivery ratio is relatively high while the overhead remains low. Increasing the number of allowed copies will not endlessly improve these statistics: the overhead grows rapidly in the binary case and there are not enough encounters for the non-binary case to keep delivering better results.

The PRoPHET algorithms use historical information to route messages. In simulations that can be found in literature, there often is a warm-up period for history based protocols to build up an initial understanding of the network. In our scenario this is not possible: the simulation is in fact a whole festival day where visitors enter the terrain at a certain moment and actually have no knowledge of the other nodes or their history. Though the delivery ratios are relatively good, these protocols produce a large amount of overhead as can be seen in the figures. While the first version of the PRoPHET protocol performs better in terms of delivery as the β-parameter is increased, the second version responds inversely. Because the results of PRoPHETv2 with $\beta = 0.09, 0.25, 0.90$ performed almost identically these results were omitted from the figures.

3.5.3 Comparison

When performing simulations with less than 500 nodes it becomes clear that the SW protocols are related to EP. Indeed, when enough copies of a message are allowed in SW (and especially in the binary case), messages can spread to almost all nodes like in EP. When fewer nodes are used in the simulation of the festival scenario, the crowd gets less connected and the spreading of the messages will be slowed down. This results in a higher latency for EP that is comparable to SW in the same setting, the latter with the advantage of requiring a limited amount of copies.

By comparing the findings of Keränen and Ott [Keränen and Ott, 2007] to our results, it is again clear that the underlying mobility models play a very important role. While the goal of sending small messages in a DTN environment is the same, they state that disabling the transitivity property ($\beta = 0.0$) in the PRoPHET protocol yields the best results in a map-based and working day movement model, whereas the music festival mobility yields better results (in terms of delivery) when this property is actually used ($\beta = 0.09, 0.25, 0.90$).

Del Duca Almeida et al. [Del Duca Almeida et al., 2012] compared MANET and DTN routing protocols in three scenarios using the ns-2 simulator. They observe that the examined DTN protocols (Spray and Wait, Epidemic and PRoPHET) have almost the same end-to-end

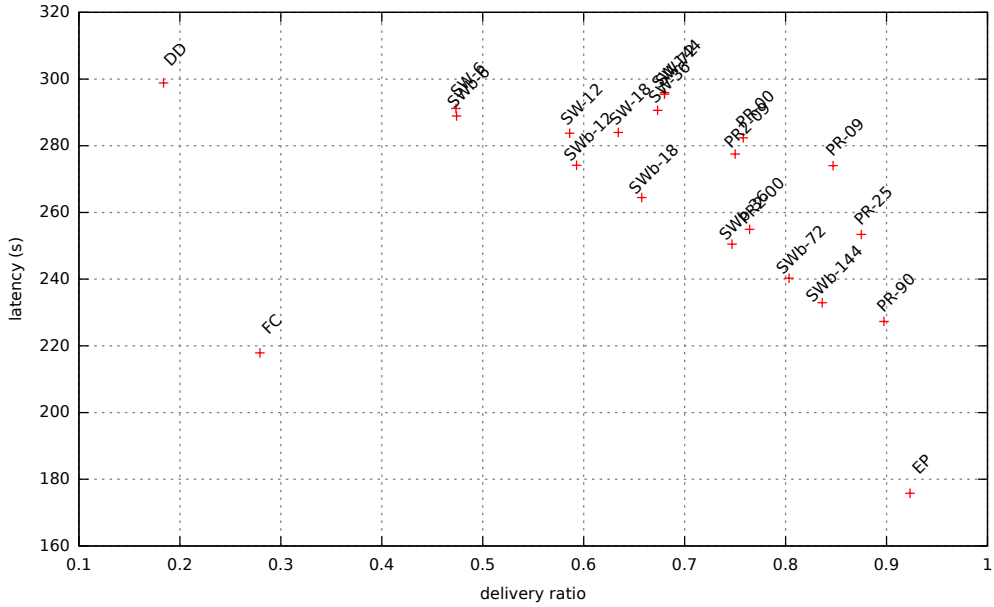
latency, using other mobility patterns with a relatively low amount of nodes. After investigating the power requirements, they report the expected increase of energy usage as protocols send many copies of messages, as in Epidemic, or need a lot of control messages, like in PRoPHET. They only tested the binary Spray and Wait protocol with $n = 6$ which has a much lower energy consumption while having a similar delivery ratio in comparison with the Epidemic or PRoPHET protocols. These findings confirm our assumptions regarding the energy requirements of the protocols.

3.5.4 Candidate Selection

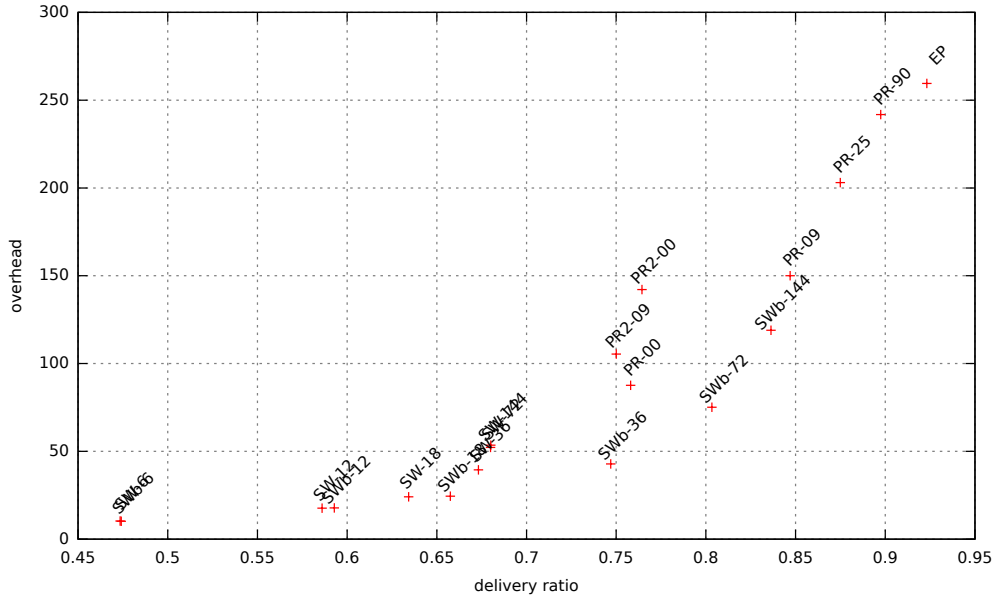
To narrow down the candidate set, two representative scenarios are studied that stress different metrics of the protocols in the results of the simulations. The first is an emergency scenario, in which it is important that small messages reach their destination quickly and with a high degree of reliability. In these conditions, power consumption and overhead are non-essential factors to decide upon the 'best choice'. To clarify the performance of the protocols in this condition, Figure 3.5(a) plots these variables in an X Y fashion. Protocols with desirable performance are ideally located in the lower right-hand quadrant of the chart. Outliers (e.g. FC) are eliminated in these charts for reasons of clarity. In this condition EP is the obvious candidate as it clearly outperforms the others.

A second scenario describes a energy-conscious condition, in which the delivery of messages is less important, but in which users rather are focused on conserving their device's power source. This translates into the minimization of the overhead and optimization of the delivery ratio metrics. Figure 3.5(b) depicts precisely these variables; candidate protocols are found in the lower right-hand part of the chart.

Overall, based on our simulations using realistic node movement, we propose the Epidemic and binary Spray-and-Wait protocol as promising candidates for use at mass events. The main advantage of EP is the high delivery rate and low latency when the crowd is sufficiently interconnected, although the overhead is relatively high. The SWb protocols show similarities to the EP protocols yet limit the amount of copies in the network, thus lowering the overhead, while performing similar to EP in smaller networks. Both protocols can be tuned according to the specific context of use (e.g. the number of messages in the SWb case). Given the fact that the routing protocol and its parameters are implemented entirely in software, such a switch can easily be performed. Indeed, Spyropoulos et al. [Spyropoulos et al., 2005] describe that the Spray and Wait protocol can be tuned online to achieve the desired QoS requirements, providing a practical way for adapting the protocol to the current situation. The EP protocol could analogously be tuned by e.g. adjusting the sending frequency depending on the number of received messages. Furthermore, given the underlying mobility, the latency of the delivered messages is lower as in more advanced protocols like PRoPHET while similar or better delivery ratios are realized. Lastly, the amount of control messages needed for these protocols is relatively low.



(a) Emergency condition



(b) Energy-conscious condition

Figure 3.5: Scenario-based performance evaluation.

3.6 Conclusion

In this chapter we have shown an application of collecting mobility data from smartphone users at a mass event, showing how tracking technology can be used for good by third parties, touching on the second part of research question **RQ1**. We used collected mobility data to compare simulations of opportunistic routing protocols under realistic movement models. We have used four metrics to analyse and describe properties of the routing protocols. By highlighting and analyzing several important properties of the protocols, we have been able to compare the protocols. Based on the results, the Epidemic and binary Spray-and-Wait protocols are proposed as most suitable candidates for use in festival scenarios as they perform similar to more advanced protocols in terms of delivery ratio, while keeping the overhead and need for control messages relatively low. Moreover, it is possible to tune parameters at run time to perform better in certain situations.

As already stated in the previous chapter, results of these simulations can be used when developing a smartphone application which allows opportunistic (ad hoc) communication at mass events such as music festivals. Indeed, in such an application a similar protocol as evaluated in the simulations can be used to route data packets in the network. We will discuss this further in section 11.2.

Chapter 4

Assessing the Impact of 802.11 Vulnerabilities using Wicability

4.1 Introduction **51**
4.2 Capability aggregation **52**
4.2.1 Acquisition **52**
4.2.2 Matching and processing **53**
4.2.3 Presentation **54**
4.3 Case study: prevalence of devices susceptible to active probing attacks **54**
4.4 Datasets **56**
4.5 Conclusion **56**

4.1 Introduction

After discussing how surreptitious tracking of smartphone users is possible, and how these techniques can be used both for nefarious purposes and for good, we show how the impact of this and other techniques and vulnerabilities (such as the use of 'stimulus frames' [Bratus et al., 2008]) can be measured. Indeed, even though these tracking techniques have obvious advantages to researchers (as we saw in the previous chapter), they should be considered as 'vulnerabilities' to the smartphone users themselves, who are tracked surreptitiously without the ability to decline it.

An impact assessment is based on the severity of the vulnerability itself and the number of affected devices. While the severity of the vulnerability is an arbitrarily defined concept that may include properties such as exploitability, remediation level, impact on availability or confidentiality, etc., the number of affected devices can be objectively measured. To measure the number of affected devices, several approaches can be considered depending on whether the vulnerability is caused by an implementation issue (vendor or operating system specific), a protocol design flaw, or a combination of both. In case of a vulnerability in a protocol such as WPS or WPA/TKIP for example, one could sample a number of `Beacon` frames from Access Points (APs) in a nearby city to approximate what percentage of APs supports the protocol. For vendor specific vulnerabilities, e.g. in a specific model of smartphone, it might be useful to look at sales reports[1] to see whether the device is prominent in the market or not. Unfortunately, such reports can be very expensive to obtain. Furthermore, the number of affected devices depends on geographical location and time: a given protocol could become deprecated (e.g. WEP), and some countries will adopt new protocols faster than others.

To help solve these problems, we introduce Wicability[2], an open platform created for researchers that aims to provide insights into the spatial and temporal impact of security vulnerabilities through the analysis of 802.11 Information Elements (IEs). We have performed an initial analysis on our own datasets, gathered through the WiFiPi system described in Chapter 2, and welcome contributions from external researchers. Our tool distinguishes itself from other open databases such as `WiGLE.net` [WiGLE.net, 2016] and CRAWDAD [Yeo et al., 2006a] in that it can be used to determine the percentage of devices that support a given protocol at a certain time and location. In the next sections, we will discuss this platform.

We first explain how the Wicability platform operates, showing how the collected data is transformed and presented by the system. We then present a case study, based on vulnerabilities that can be used to instigate transmissions from tracked devices.

[1] Sales reports can include, for example, Forrester or Gartner reports.

[2] The Wicability website is available at `https://wicability.net/`.

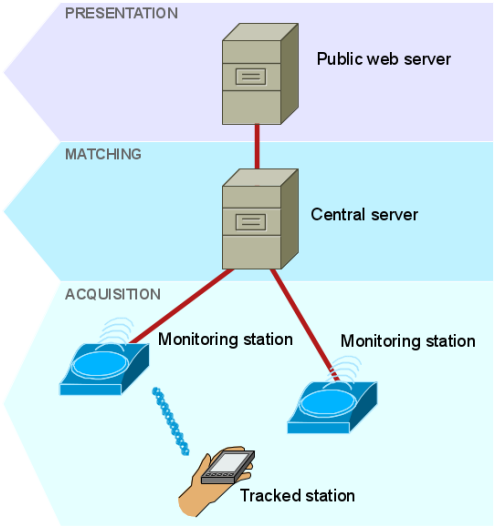


Figure 4.1: The three processing stages of Wicability

4.2 Capability aggregation

The protocols and capabilities supported by different devices are advertised in IEs. Such IEs are exchanged between stations (STAs) and APs prior to association through `Probe Request`, `Probe Response` and `Beacon` frames so that both parties know which protocols, data rates, and crypto suites can be used for communication. Our approach for aggregating this information comprises an acquisition, matching and presentation stage as shown in Figure 4.1.

4.2.1 Acquisition

To obtain a representative set of IEs, we have deployed multiple monitoring devices at densely crowded locations. Here, each monitoring device passively captured all management frames containing IEs using `libpcap` and a wireless interface configured in monitor mode. The captured frames were then forwarded to a central server over an SSH tunnel. No additional processing was performed at the monitoring devices in order to minimize their complexity. Alternatively, a `pcap` file can be provided to the server directly for analysis.

We are aware of the fact that the forwarded messages contain privacy sensitive data such as the MAC address and SSID list. To ensure the privacy of the monitored STAs, we do not collect any data frames, and we make sure that none of the collected MAC addresses or SSIDs are accessible through the public platform.

4.2.2 Matching and processing

In the matching stage, the captured IEs are grouped per MAC address and per dataset. The MAC addresses are only used to distinguish between different devices. Thus, external datasets may be supplied using only anonymized MAC addresses, as long as each real MAC address is consistently mapped to its corresponding pseudonym. For example, we anonymized our own datasets by replacing the 3 least significant bytes of each MAC address with a different value, with exception of "ff:ff:ff" and "00:00:00". For example, the 3 least significant bytes of the first encountered MAC address in our dataset are replaced with "00:00:01", those of the second one with "00:00:02", and so on. As mentioned in Section 2.8, to be able to consistently map each device to the same anonymized MAC address, this anonymization can only be performed after the data collection is complete.
As will be discussed in Section 8.2.2, devices might randomize their MAC address in order to avoid being tracked in the way discussed in Chapter 2. This makes it seem like different packets from this device originated from different devices. Ideally, frames containing randomized MAC addresses should be excluded in order to prevent counting the same device multiple times. We offer two approaches to filter these random MACs. In a first approach, we filter out frames having a MAC address that has either of the following properties:

- The Organizationally Unique Identifier (OUI) part of the MAC address does not correspond to a registered vendor in the OUI list, managed by by IEEE.[3] The OUI comprises the first 6 bytes of the MAC addresss, and corresponds to the vendor or manufacturer of the wireless chip or the mobile device.

- The "locally administered bit" of the MAC address is set. This bit (the second-to-least-significant bit of the MAC address' first octet) indicates whether the MAC address is globally unique and assigned by the manufacturer (set to 0) or whether it is locally administered (set to 1).

Our second approach utilizes MAC layer fingerprinting techniques that are described in a different paper [Robyns et al., 2017] (not included as part of this dissertation) to link similar IEs transmitted by random MAC addresses to their corresponding real MAC address.
Some processing is done to extract the original IEs and their associated IE fields from the management frames. Additionally, the OUI part of the MAC address is resolved to its corresponding "assignee" (vendor or manufacturer), in order to allow users of the Wicability platform to filter by vendor, or to view the vendor distribution for different capabilities and IE fields.
Finally, the dataset is labeled with the location, duration and timestamp of the capture. Datasets can also be filtered afterwards based on this timestamp, to account for longer-running data acquisition phases.

[3] A list of all OUIs registered by IEEE is available at `http://standards.ieee.org/develop/regauth/oui/oui.txt`.

4.2.3 Presentation

After the observed IEs have been matched with a specific device, each IE is parsed, converted to a queryable and human readable format, and stored in a public database. Privacy sensitive data such as the MAC / pseudonym and SSID names are excluded from this operation. The resulting dataset can be queried by researchers using the Wicability web interface. Figure 4.3 shows an example where the distribution of `Supported Rates` values is shown by filtering for that specific IE. Also shown is the vendor distribution for devices that transmit the `Supported Rates` IE. The distribution of each possible field, field value or vendor in an IE can be queried.

4.3 Case study: prevalence of devices susceptible to active probing attacks

To demonstrate how researchers can use the Wicability platform to show the impact of a vulnerability, we provide an example case where we assess to what extent a newly discovered technique for instigating Wi-Fi packets would help in detecting devices.

In 2017, we published a paper (not included as part of this thesis) describing a number of methods for increasing the number of packets sent out by Wi-Fi devices in range of a monitoring station, used for tracking [Robyns et al., 2017]. 802.11 frames that can be used for this purpose are sometimes referred to as 'stimulus frames' [Bratus et al., 2008]. For the purposes of this case study, we use one specific example of a stimulus frame that we describe in the paper, called a Generic Advertisement Service (GAS) request. The technique (or attack, depending on the viewpoint) using GAS request frames causes any device in range supporting the IEEE 802.11u "Interworking with External Networks" standard to respond with a GAS response frame. The described method could be used, for instance, to improve tracking techniques such as the one being described in Chapter 2 by increasing the number of transmissions from devices. This allows not only a larger number of devices to be tracked, but also allows for an increased number of data points per device, allowing for more granular tracking.

To see what percentage of devices would be susceptible to this technique, we can use the data that is available through Wicability. Indeed, since this technique can be used on any device supporting the IEEE 802.11u standard, and since support for this standard is announced by devices in specific IEs, we only need to look at the number of devices that included this IE as part of their `Probe Request` frames.

To get an idea of how many devices are affected, we inspect the "Glimps 2015" dataset, containing data for 28 047 devices collected over the period of three days at a music festival in Ghent. We limit ourselves to only mobile devices from users (filtering out access points) by selecting the "Only STAs" option, and can see directly from the graphs that 7 508 of these

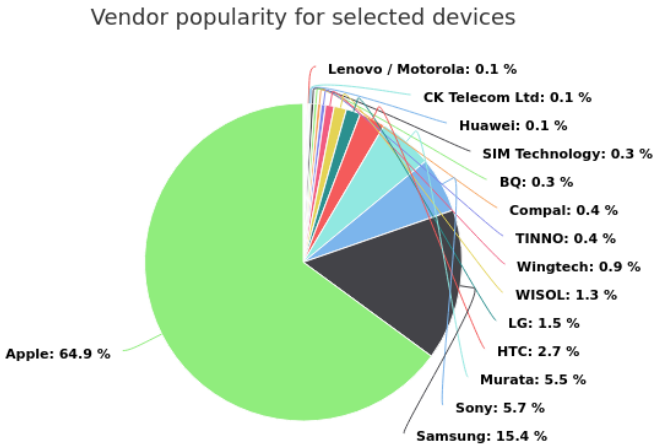


Figure 4.2: Vendor distribution for 7508 devices supporting the "Interworking" capability.

devices support the "Interworking" capability. From this, we can conclude that the described technique would allow us to more easily track 27% of smartphones.[4]

To see which types of devices are most likely to support Interworking capabilities, we can filter the dataset by the presence of the Interworking IE. This shows that by far the most popular vendor supporting Interworking (and thus, being vulnerable to this attack) is Apple, accounting for 64.9% of affected devices. This does not only show that using the aforementioned technique can be a great help in detecting Apple devices, it also shows that using the technique might skew the set of detected devices towards including more Apple devices, revealing a bias that was not clear beforehand. Thus, we conclude that the Wicability platform can be beneficial to researchers when trying to assess the impact of newfound techniques and vulnerabilities.

A similar analysis can be performed for another type of stimulus frames described in [Robyns et al., 2017], called Block Acknowledgement frames. These frames must be supported by any device supporting high throughput (802.11n) standards. However, in our experiments we noticed that only 802.11n devices manufactured by Intel actually responded to these stimulus frames. For this case, and again using the Glimps 2015 dataset, we can get an idea of the percentage of affected real-world devices by filtering the dataset for only STAs broadcasting the "HT Capabilities" IE that were manufactured by Intel. Filtering for manufacturer in this case limits us to only 0.5% of all (21 349) devices broadcasting the HT Capabilities IE, leaving only 101 devices (or 0.004% of all encountered devices) affected by this technique.

[4]Note that this 27% is an upper bound, since some of these devices might only support peer-to-AP GAS messages, and not peer-to-peer GAS messages. We refer to our original paper [Robyns et al., 2017] for more information.

Of less interest to this specific example, but interesting to note nonetheless is that, even though only STAs were selected (excluding access points from the dataset), 8.2% of them had set their Interworking fields "Internet" and "ASRA" to '1', indicating the ability to provide internet access to other devices. This could indicate that these devices were allowing internet tethering to other devices, though these devices also set their "ESR" and "UESA" fields (usually reserved to only access points offering reachability to emergency services) to 1, which might indicate that these devices did not adhere to the standard instead. Indeed, manually verifying the original data shows us that these devices (all of them having Apple as their vendor) have all of their Interworking fields set to '1', suggesting that the standard was not adhered to for some Apple devices.

4.4 Datasets

As of March 13, 2017, the Wicability website offers statistics for four datasets:

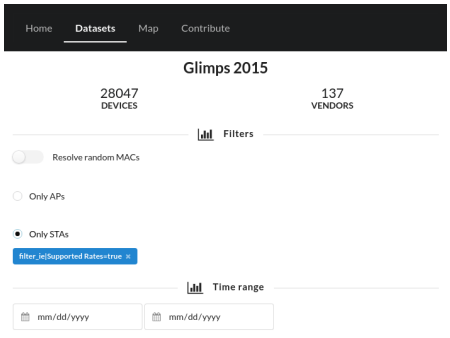
- A dataset of STA probe requests collected by Marco V. Barbera et al. at Sapienza University in Rome, Italy in 2013 [Barbera et al., 2013], which we gathered from the CRAWDAD website [Yeo et al., 2006b].
- A dataset of STA probe requests collected at political meetings by the same team as the previous dataset, again gathered from the CRAWDAD website.
- A dataset of STA probe requests collected by us (in collaboration with The Safe Group) at the Glimps music festival in Ghent in 2015.
- A dataset of metadata from packets originating from both STAs and APs, collected at our own research institute (UHasselt EDM) in 2016.

4.5 Conclusion

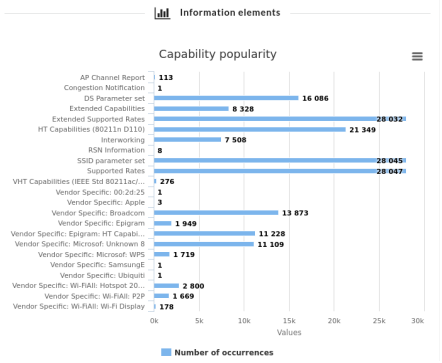
We have introduced Wicability, an open platform that can be utilized as a tool to quantify the impact and remediation rate of protocol vulnerabilities, covering research question **RQ2** and providing a platform to researchers who want to assess the impact of any discovered protocol vulnerabilities in the future. Additionally, the platform can be used to determine the number of devices observed from a specific (chipset) vendor or operating system, along with their supported capabilities. An overview of its core functionality was presented, which comprises the collection and analysis of IEs acquired through passive monitoring. Furthermore, we provided a case study based on our own work that shows how the Wicability platform can be used by researchers to assess the impact of such newfound techniques and vulnerabilities.

To complement our own collected data, we welcome submissions from external researchers to the Wicability platform. These submissions can be provided in the form of anonymized

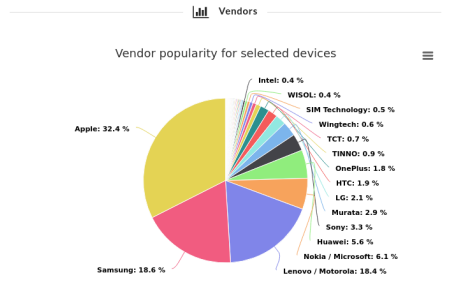
`pcap` files, where the SSID names and MAC addresses have been replaced with pseudonyms. As a result of these contributions, progressions such as the adoption of 802.11w amendment support for protected management frames in response to `Deauthentication` frame Denial of Service (DoS) attacks for example, can be studied in a spatio-temporal manner.



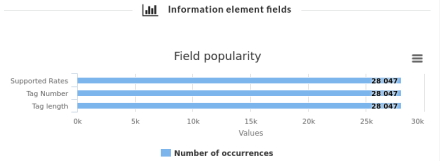
(a) Different filtering options, allowing to resolve random MAC addresses, filter out access points or non-AP STAs, and define a time range.



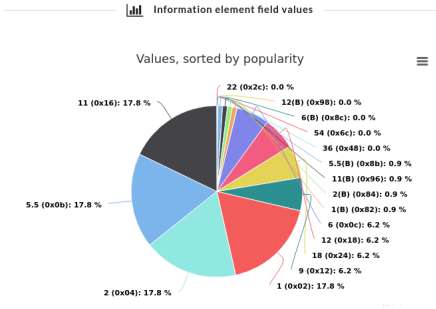
(b) List of IEs sent out by selected devices, allowing for selection of one specific capability which will then be added to the filter list



(c) The distribution of vendors corresponding to all devices included after applying the filters (including selection of IEs).



(d) Fields sent out for the selected IE.



(e) The popularity of the different values for the selected field.

Figure 4.3: Example use case where Wicability is queried for the `Supported rates` IE of non-AP STAs.

Part I: Conclusion

In this part, we demonstrated how easy it is to gather privacy-sensitive information from smartphone users' devices. We showed that people tracking can be achieved at a very low cost and – more importantly – unobtrusively and without requiring active cooperation from the users themselves. We demonstrated this in the context of a three-day long music festival by tracking 29% of the 100 000 festival visitors (and up to 46% in a repeated study). We discussed how, since publishing our work, different marketing agencies and public institutions have already started adopting similar techniques.

To show that such data collection does not need to be used for nefarious (or business) purposes, we also present an extra case study, where we use mobility data gathered during the music festival to optimize opportunistic routing protocols that can be used in the development of an opportunistic communication app. The usage of realistic movement data instead of traditional synthetic models helped to select routing protocols that are suited for use in actual festival scenarios; we concluded that the Epidemic and binary Spray-and-Wait routing protocols are good candidates when developing such an app.

Even so, privacy implications of the ability to use similar techniques to extract privacy-sensitive information are abundant, and are amplified by the fact that they can be executed at a very low cost. Indeed, even if marketing agencies and public institutions comply with ethical and legal standards, nothing prevents a malicious actor from using the same techniques to gather information or mount attacks. We will go into more detail about these privacy and security issues in the next part of this dissertation, where we will look at how we can both measure, and improve, users' awareness.

To aid other researchers investigating security and privacy issues in wireless networks, we introduced the Wicability platform, which can be utilized as a tool to quantify the impact and remediation rate of protocol vulnerabilities. We demonstrated the utility of the system by providing a case study based on our own work that shows how the Wicability platform can be used by researchers to assess the impact of such newfound techniques and vulnerabilities.

Part II

Creating and assessing user awareness

Introduction

As is clear from the previous chapters, many threats to security and privacy exist when connecting to Wi-Fi networks using a smartphone or other mobile device. To gauge how aware users are about this fact, we conduct two studies, both of which confront randomly selected mobile device users with their privacy-sensitive data that is surreptitiously collected. A third study is conducted to explore the reasoning of smartphone users when making other privacy-sensitive decisions. These studies do not only aim to provide an indication of the users' awareness, but also serve as a way for increasing this awareness for the modal user.

The first study uses a system called SASQUATCH, which builds directly on the WiFiPi system from Chapter 2. SASQUATCH is a setup consisting of a network scanner and a public display, which alerts users about privacy issues related to W-Fi networks by displaying private but anonymized information. The study consists of having random passers-by interact with this setup, and asking them about the accuracy with which this information can be used to identify them. It also asks about their awareness about this information being available, with a focus on gauging how worried participants were about this fact.

The second study uses the personal mobile devices of preselected participants to both assess and increase these participants' awareness about privacy and security issues. For this purpose, participants are asked to install an app on their devices for a period of 30 days. This app collects information about which apps connected to the internet over which Wi-Fi networks, together with security characteristics of these connections. Again, participants are confronted with this information at the end of the study with the goal of gauging both their awareness and privacy sensitiveness.

The last study uses a similar methodology to the second one to explore the reasoning of smartphone users when they're making other privacy-sensitive decisions related to apps. This is done by asking 150 participants about this reasoning using in-situ questions, asked at the moment (and with the context) the decision is made. We will focus on decisions related to runtime permissions, where users need to decide on apps' access to sensitive data using context at hand.

Chapter 5

Raising Awareness on Ubiquitous Privacy Issues with SASQUATCH

5.1 Introduction **67**
5.2 Mobile Phones that "Never Forget" **68**
5.3 The SASQUATCH System **68**
5.3.1 Inferring a smartphone's whereabouts **69**
5.3.2 Determining a network's authentication type **70**
5.4 Study **71**
5.5 Results **74**
5.6 System analysis and limitations **77**
5.7 Conclusion **79**

5.1 Introduction

As we saw in the previous chapters, large scale collection of private data is something that is relatively easy to do, even with limited resources. Thus, it is not something that is reserved to large organizations with world-wide networks, as is often thought. To see how aware smartphone users are about these privacy and security dangers, and to increase this awareness in the process, we conducted a study in 2014.

In recent work, Könings et al. propose a model to enhance user-centric privacy awareness [Könings et al., 2013]. Three core questions are put forward in this work that contribute to the understanding of privacy: who is affecting my private data, what is the purpose (why) and how is it being accomplished. Most mobile phones, however, openly share their list of previously accessed networks to increase the comfort and ease of use for the end-users but fail to inform the users about the consequences. A study performed in 2009 by Klasnja et al. shows that users are unaware of important privacy risks when using Wi-Fi networks [Klasnja et al., 2009]. An interesting result in this study is that many of the participants thought about a hacker as a highly skilled attacker who breaks into their computer, and not as someone who can passively collect their data when it is sent over the network. The fact that people are in general very susceptible to Wi-Fi attacks is confirmed by Kindberg et al., who were able to mount a phishing attack on the login page of a public hotspot, tricking 32% of users connecting to their own fake hotspots into entering their mobile phone number [Kindberg et al., 2008].

Other work shows that, when confronted with possible privacy concerns, people are willing to act in the interest of preventing further privacy leaks. A survey of 2254 participants by Boyles et al. [Boyles et al., 2012] demonstrated that 57% of all smartphone app users have either uninstalled an app over concerns about having to share their personal information, or declined to install an app in the first place for similar reasons. An empirical study by Günther et al. [Günther and Spiekermann, 2005] assessing the privacy fears related to using Radio Frequency Identification (RFID) showed a similar trend: 73% of 129 retail consumers preferred RFID to be disabled on checkout. Moreover, results presented by Consolvo et al. [Consolvo et al., 2010] show that a user's privacy awareness can be increased by showing the user personal information that is unwittingly shared. The ubiquitousness of wireless networks and mobile hardware, together with the impact of these practices and the importance that users attach to privacy make that there is a urgent need for raising awareness and making mobile system developers accountable for informing their users.

In this work, we describe a system called SASQUATCH that collects privacy-sensitive information that is sent out inadvertently by people's smartphones.

The main contributions for this chapter are as follows:

1. We explore whether people can identify themselves when we show them information that is contained in a list of previously accessed networks.

2. We report about creating user awareness about potential privacy and security issues when a smartphone connects to Wi-Fi networks.

3. We assess the effort the average smartphone user is willing to invest in securing against privacy leaks.

4. We provide a method to explain to smartphone users what can be done to secure against privacy and security issues.

For this purpose, we combined SASQUATCH, our system that gathers data from mobile phones and impersonates networks, with a public display that pictures the data that is captured. The visualizations used are, of course, anonymized and carefully abstracted.

5.2 Mobile Phones that "Never Forget"

Remember that the WiFiPi system from Chapter 2 collects Wi-Fi signals sent out by smartphones at multiple strategic locations on the festival terrain. By using the unique smartphone identifier (*MAC address*) contained in each of these Wi-Fi packets, and correlating captured signals at different locations, it is able to track visitors over the entire festival area. As we discussed in Section 2.6.3, we also repeated this experiment, collecting additional data in the form of SSIDs (network names for the networks remembered by the visitor's smartphones), which allowed us to uniquely identify nearly a quarter of smartphones based on only one of their remembered networks. In Section 2.8, we already hinted at possible privacy implications of our tracking system.
Similar to results by Klasnja et al. [Klasnja et al., 2009], we noticed that most people are not aware of this information leakage. In a short prestudy in our lab environment we were able to identify a significant number of our own colleagues (researchers with a background in computer science) solely based on the network SSIDs sent out, the manufacturer of the smartphone or the time at which they entered or left our lab building. We confronted 10 colleagues with this information. All but one of them indicated that they were not aware that this information was so easily obtainable. This is not surprising, as many smartphones do not display the list of 'remembered' networks. iPhones and iPads in particular only show a stored network name when it is in range, leaving users in the dark about which information is leaking from their phones (or, that any information is leaking at all).

5.3 The SASQUATCH System

SASQUATCH consists of a single machine capturing all probe requests that are sent out by smartphones in range. The information obtained through this mechanism is used in three ways:

- The SSIDs of the networks broadcasted by a user's smartphone are used to create a profile of the user. This is done not only by gathering the network names, but also by looking up the SSIDs of these networks in the `WiGLE.net` wardriving database [WiGLE.net, 2016], which allows to find the specific locations of access points. These locations are used to make an educated guess (calculated via the method outlined in section 5.3.1) about the user's whereabouts.

- The system executes an Evil Twin attack [Roth et al., 2008], impersonating networks in the smartphone's PNL, in order to identify which networks correspond to open access points. How this method works, and how it can be used to identify open networks is discussed in section 5.3.2.

- The manufacturer of the smartphone is derived from the Wi-Fi packets by deriving the Organizationally Unique Identifier (OUI) from the MAC address in each packet, in the same way as for the Wicability system (see Section 4.2.2 for more information). The system performs a lookup for this identifier in the OUI list, kept up to date by IEEE.

5.3.1 Inferring a smartphone's whereabouts

Since the SSIDs of wireless networks may be used by multiple access points (often located at different locations), there is no one-to-one mapping of networks and locations. To make an educated guess about a subject's visited locations, we combine both types of networks (open as well as secured), and query the `WiGLE.net` wardriving database. This database contains crowdsourced data about networks, among which their GPS coordinates. The data is collected by people (calling themselves "wardrivers") who drive around, scanning for all networks in range, and collecting information about those networks (including the used encryption, the wireless channel, and other parameters that are available by only scanning) [WiGLE.net, 2016]. Our method for deriving a location from a network SSID is as follows:

1. For every network in the smartphone's PNL, query `WiGLE.net` for a possible list of locations, returned as $(latitude, longitude)$ tuples. Assign to every one of these locations a chance of $\frac{1}{\#locations\_for\_network}$. For instance, because the SSID `UHasselt-Guest` returns 19 possible access points at different locations, we assign to each of these locations a chance of $1/19$. This chance only depends on the number of results for the *current network*, and is in no way related to the number of locations returned for other SSIDs in the subject's PNL.

   At this moment, we have no way of telling whether any or all of these locations are results for access points at the same location. This will be accounted for in the next steps.

2. Use Google's *reverse geocoding* API to infer the city corresponding to the $(latitude, longitude)$ tuple returned by `WiGLE.net`. For instance, given that the first result returned for SSID

`UHasselt-Guest` is the tuple $(50.93173981, 5.39291286)$, we infer that this network has a $1/19$ chance of being located in Diepenbeek. We denote this as $p_{(city,network)}$ (or $p_{(\text{Diepenbeek,UHasselt-Guest})}$ for our example).

3. Combine the chances for the cities in the previous step for all of the smartphone's networks to deduce the chances that the user visited a certain city, by assigning every city a value that is the sum of the chances that any of the networks is situated in this city, as follows:

$$p_{city} = \sum_{networks} p_{(city,network)}$$

The resulting value should not be thought of as a mathematical probability, since it is possible that it is larger than 1. Rather, the value gives an indication on the probability that the device effectively visited a specific city. We discuss how this value is used to assess whether a device visited a certain location below.

Even if multiple networks have only a small chance of individually being associated with a certain city, together they can be used to infer that a person went to that city with high probability. For example, assume that network N_1 has a chance of only 0.5 of being located in either city C_1 or city C_2, and network N_2 has a chance of 0.3 of being located in either one of cities C_2, C_3 or C_4. It can then be inferred that the user has a high probability of having visited city C_2. Moreover, networks that return different possible locations may have a majority (or all) of them located at the same approximate location (e.g., because there were multiple hotspots with the same SSID in the same building). This is also accounted for by recombining the location chances in the last step.
SASQUATCH determines a device to have been at a location if the resulting p_{city} value for that location is strictly greater than 0.5. The reasoning behind this is as follows: if a network is available at two or more locations, the maximum possible chance assigned to a specific location is 0.5. Therefore, at least two networks are needed to obtain a chance > 0.5. A similar case can be made for networks that exist at four locations (at least four are needed), and so on.

5.3.2 Determining a network's authentication type

Probe requests only contain the SSID of the network the device wants to connect to, omitting the type of authentication that will be used for this connection. This authentication information is only available as part of the probe response, sent out by the access point.
Because probe requests do not contain an authentication type, SASQUATCH sends out a probe response as if it is an open network for every probe request it received, effectively pretending to be an open network in the smartphone's network list. Thus, when a smartphone tries to connect to the SASQUATCH system (by sending it an *authentication request*), our system can infer that the network with the SSID contained in the probe request is indeed

Figure 5.1: The setup, as it was deployed at our research insitute. The main screen, displaying aggregate information about all smartphones in a range of 50 meters is shown on the left. To the right of this screen is another (smaller) screen, on which a smartphone user can choose to display the information our system gathered from his/her smartphone.

an open network. This method of tricking devices into connecting to an access point by pretending to be a known network is called an Evil Twin attack [Roth et al., 2008].

Our system stops here: smartphones that try to connect to SASQUATCH are prevented from actually establishing a connection by not acknowledging their association request. A person with malicious intent, however, can continue from here and set up a successful attack: as soon as a smartphone connects to one of the spoofed networks, this attacker is an *active man in the middle*. This allows the attacker to capture the smartphone user's data, even when it should be sent over encrypted (SSL) connections [Marlinspike, 2009]. This is especially worrying with smartphones, where many applications continuously check for new information in the background.

5.4 Study

We have performed a field study in which we explored whether people can recognize themselves if bits of information retrieved from their smartphone are displayed, assessed the level of awareness people have about the "open nature" of their smartphones, and evaluated the people's willingness to secure their smartphones against information leakage. Based on this, our hypotheses are that (1) people will be able to recognize themselves when their data (consisting of past networks and locations) is shown, (2) people are unaware about the amount of information being shared by their smartphones, and (3) people are willing to put a minimal effort (e.g. installing an app) in securing their smartphones.

The apparatus we deployed for our study is a public display that shows the data gathered by SASQUATCH. The public display setup consists of two parts: a large display for public usage and a smaller one that can be used individually (see Figure 5.1):

- The large public display shows both the aggregated locations and the insecure (open) networks of all smartphones that were in range in the past 5 minutes.
- The smaller display shows the information that was inferred from a single smartphone. Viewers are invited to give consent for displaying their information by performing an action.

The public display visualisation was designed to trigger a "honeypot" effect [Brignull and Rogers, 2003], i.e. to entice people to start interacting with the screen. The map-based visualisation in combination with a list of names of networks (see Figure 5.2) made it easy for people to recognize (part of) themselves on the display while remaining anonymous.
Participants in our field study were passers-by that were drawn to the screen out of curiosity because of what they noticed on the screen. We also had an assistant that stimulated people to look at the display. We distinguish between *passive participants*, participants that inspected the public display but did not interact, and *active participants* who actively engaged with the setup to view the information that was shared by their smartphones.
We deployed our apparatus at two different locations: our research institute entrance hall and the university's main hall. The setup was active for several days at the research institute and attracted 51 active participants over the course of a week. At the university's main hall we deployed the system for one day and we attracted 31 active participants. We estimate the number of passive participants is three to five times the number of active participants (based on our observations; no counts were done of passive participants).
The public display (see Figure 5.2) is designed in a manner similar to work by Kowitz et al. [Kowitz and Cranor, 2005], in that it aims to strike a balance between notification and privacy. We achieve this by having the display show a list of locations that any of the devices that are in range (approximately 50 meters) have visited, based on the algorithm described in section 5.3.1, as well as an overview of *open* networks these devices have in their network lists. The reason for this is that it is easy for people looking at this information to identify themselves (inviting them to further inspect the setup), while keeping privacy-sensitive information hidden from other people. Indeed, it is infeasible to identify the locations a particular person has visited by looking at the aggregate information. Similarly, while it is possible to view the aggregate SSID information about all devices in range, it is infeasible to determine the networks a specific device connected to. This also caters to possible legal issues: before we started with the study, we sought advice on possible legal issues and ensured we were allowed to show the aggregated information on the public display.
The smaller, private display initially shows the steps a visitor has to undertake in order to view the information his/her own smartphone is leaking (see Figure 5.3). As soon as user consent is given (by connecting to a specific network or scanning a QR code), his/her information

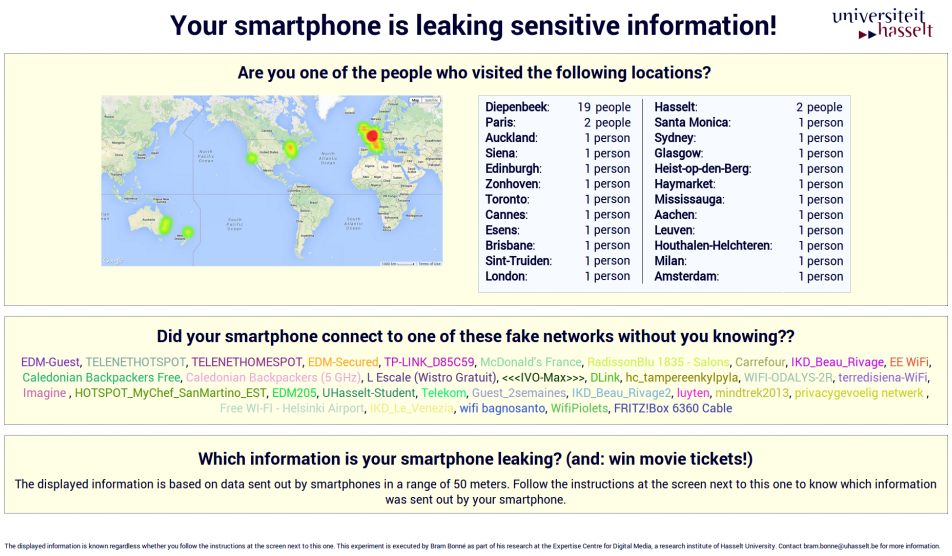


Figure 5.2: A screenshot of the information available on the public display. The top of the page shows the aggregate locations for all devices, together with a heatmap of these locations. The bottom of the page shows a list of open networks devices wanted to connect to, and an invitation to partake in the study.

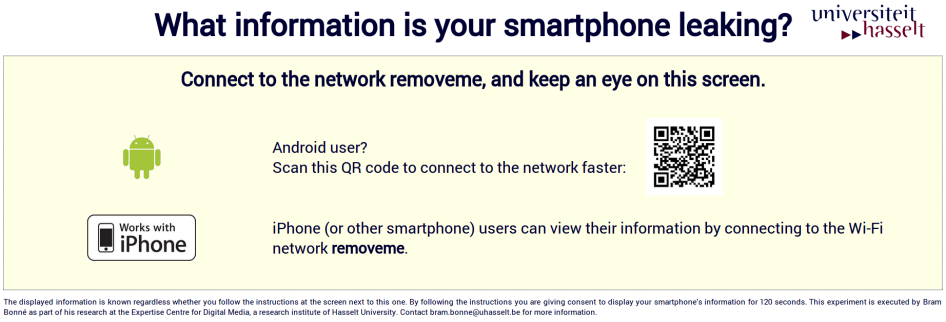


Figure 5.3: A screenshot showing the instructions that are displayed on the private display. The visitor is invited to scan a QR code or connect to a network to display his/her personal information.

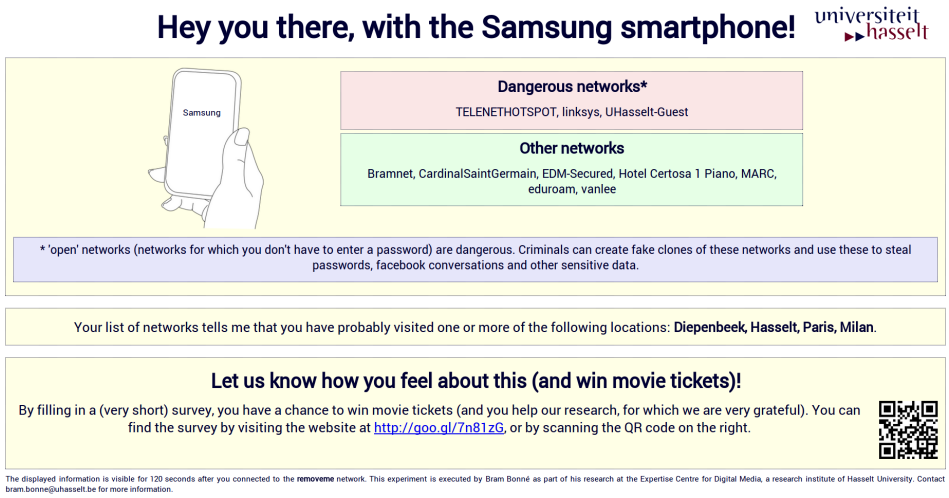


Figure 5.4: A screenshot showing private information for a smartphone. Included are the lists of open and closed networks, the inferred locations, and an invitation to fill out the survey. Also briefly explained are the dangers of connecting to open networks.

is displayed in a specific manner (see Figure 5.4), distinguishing between open (dangerous) networks and closed networks. Also displayed is the list of locations that SASQUATCH inferred to have been visited by the user.
After the participant information is displayed, he/she is asked to complete a short survey. The survey inquires on the accuracy of the displayed information, on how worried the user is about this information leaking to third parties, and on the amount of effort he/she would spend to mitigate information leakage. At the end of the study, the user is given a tutorial teaching him/her how to prevent future leaks.

5.5 Results

During our field study, the setup captured Wi-Fi signals of 1 404 devices. 82 people chose to show their personal data on the second screen of our setup, and 66 people filled out our survey. The survey participants (both male and female) were aged from 17 to 59 years old, all having completed high school or higher.
The packets sent out by smartphones during our study showed that the problem of a PNL containing open networks is widespread: of the 1 404 devices that passed our setup, 628 (45%) contain at least one open network in their PNL[1]. The second problem, where an eavesdropper

[1]This is a lower bound, as most devices will not have broadcasted their complete PNL during the time they were in range of our setup (see Section 5.6). Moreover, a significant portion of networks may have been connected to the university network, which might have prevented them from trying to initiate a connection with our (spoofed)

Table 5.1: Statistics for devices that passed by the SASQUATCH setup

Detected devices	1 404
Devices with open network	628 (45%)
Devices mapped to real-world location	893 (64% of total, 88% of long-in-range)

Table 5.2: Correctness of the data displayed by the SASQUATCH system, as reported by the participants

None of the displayed data was correct	0 (0%)
One point of information (location or network) was correct	12 (21.8%)
Some information was correct	8 (14.6%)
Almost all information was correct	8 (14.6%)
All displayed information was correct	27 (49.1%)
I did not use the system	11 (16.7% of total)

would be able to determine a smartphone's whereabouts based on the networks in its PNL is also real: we are able to relate 893 devices (64%) to at least one real-world location with a certainty of over 0.5 (calculated as in section 5.3.1). If we account for the fact that not everyone's full PNL was captured (see Section 5.6) by only counting smartphones for which we captured at least 3 SSIDs, we find that for these 273 devices, 241 (88%) of them could be mapped to at least one real-world location. An overview of this data is available in Table 5.1.
To check the correctness of both the networks and the inferred location data, we asked the survey participants whether they were able to recognize their own data. 55 of the survey participants had chosen to display their data on the second screen, of which 64% indicated that nearly all or all information was correct, with the other 36% indicating that only part of the information was correct (a full overview is available in Table 5.2). None of the participants indicated that no correct data was shown. This confirms our *first hypothesis*, which says that people are able to recognize their information when the data related to their smartphone is shown.
Survey participants were most surprised about an attacker's ability to spoof networks that were in their PNL: 34 of them (52%) expressed that they were not aware that this could be done. Furthermore, 29% of the participants was not aware their visited locations could be extracted by an eavesdropper, with 30% indicating they did not know their preferred networks could be seen by anyone with basic computer knowledge. In total, only 24% indicated that none of the displayed information surprised them, confirming our *second hypothesis* that most people are unaware about at least part of the data leaking from their smartphones. These results are summarized in Table 5.3.

network, even though they broadcasted probe requests. In Chapter 6, we will show that one in three smartphone users will connect to at least one open network in the course of 30 days.

Table 5.3: Pre-study awareness of participants about different security and privacy issues related to using a mobile device on wireless networks, as measured by the response to the question "Which of the displayed information surprised you?"

None: I was aware that others could see all of the displayed information	16 (24.2%)
My visited locations	19 (28.8%)
The list of networks in my smartphone	20 (30.3%)
The fact that an insecure network can be spoofed (and that my smartphone would connect to it)	34 (51.5%)

Table 5.4: Reponses to the question "For which of the following groups of people would you be worried if the displayed information was available to them?"

I don't care if anyone sees this information	13 (19.7%)
People you know (friends, family, colleagues)	19 (29%)
Civil authorities (police, security service, the NSA)	31 (47%)
Companies (stores, marketing companies)	33 (50%)
Other (random) people	43 (65.2%)

80% of the surveyed people indicated that they were worried about someone being able to view the information that was gathered by the SASQUATCH system (see Table 5.4). Survey participants were the least worried about the information leaking to friends, family or colleagues (29% of all participants), followed by civil authorities (47%), stores and marketing companies (50%), and other (random) people (65%).

Our *third hypothesis*, in which we state that people would be willing to put a minimal effort into making their smartphone more secure, is confirmed by the fact that 62% of the people surveyed indicated they were willing to make this effort, with 15% willing to do "whatever it takes". When explicitly asked whether they were willing to install an app to mitigate the privacy and security issues discussed, 59% answered 'yes'. 30% felt that it was the job of smartphone manufacturers to secure their smartphones against these kinds of attacks, even if some of them indicated that they were willing to put in an extra effort to secure their smartphones themselves. Only 5 participants (7%) did not want to undertake action because they were not worried about information leakage from their smartphone. These results are summarized in Table 5.5. To cater to people willing to make their smartphone more secure, the end of the survey contained a link with instructions about how networks can be removed from a smartphone's PNL. As a result, we noticed that several of the participants removed several networks from their PNL after participating in our survey. To cater to the 59% of participants that indicated they were willing to install an app to prevent the demonstrated security problems, we will introduce Wi-Fi PrivacyPolice in Chapter 8.3. This Android app is designed to prevent exactly the problems that were presented by SASQUATCH.

Table 5.5: Reponses to the question "What effort are you willing to take to make your smartphone more secure?"

Installing an app	39 (59.1%)
Performing a monthly manual check	29 (43.9%)
I am not willing to make an effort, this should be the job of the smartphone manufacturer	20 (30.3%)
Everything necessary, regardless of the effort	10 (15.2%)
I am not willing to make an effort, I am not worried about information leaks	5 (7.6%)

Interesting to note is that 59% of the people who filled in the survey wanted to be kept up-to-date on our new developments to help improve privacy and security for smartphone users: they actively ticked the box that their e-mail address could be used for further updates on this research.

5.6 System analysis and limitations

In Section 5.1, we hinted at the fact that a system for eavesdropping similar to SASQUATCH can be built at a very low cost with minimal effort. We consider a Raspberry Pi to provide sufficient computing power. Because of this, the basic hardware for a system similar to SASQUATCH (a Raspberry Pi and a Wi-Fi dongle supporting monitor mode) can be obtained for as little as US$40 (similar to WiFiPi, see Section 2.5). There are, however, some limitations to building a similar system.

For example, while the use of the `WiGLE.net` database is suitable for profiling a single person, the query limits imposed on users make it difficult to use this database in a system at the scale of SASQUATCH. Indeed, thanks to cooperation of the `WiGLE.net` administrator, we were able to query the database a significant amount of times more than a regular user could. A normal adversary does not have this ability, and is thus more restricted in the amount of locations he/she can collect. Nonetheless, many commercial alternatives to `WiGLE.net` exist[2], requiring only slightly more funds to execute a successful attack. Moreover, a determined adversary or one that aims to gather privacy-sensitive data on only a few people will be able to gather all the location data that he/she desires. Even in the event that the number of networks would exceed the number of allowed queries, profiling a single victim often does not require querying a location database like `WiGLE.net`. For example, many networks may be known or have an SSID named after the business or location where they reside (for corporate networks and small businesses), or after the owner of the network (for home networks). To give an

[2]Some examples of commercial solutions that provide location information based on the information of Wi-Fi access points in range are Combain (`https://combain.com/`), Skyhook (`http://www.skyhookwireless.com/`) and Navizon (now Accuware `https://www.accuware.com/`).

Table 5.6: Statistics for lookups of SSIDs on `WiGLE.net`, throughout all experiments and incarnations for which the SASQUATCH system was used.

Total number of distinct SSIDs broadcasted by all devices	6 540
SSIDs with a matching access point available on WiGLE.net	3 416 (52%)
SSIDs that mapped to exactly one access point (and thus, one location)	932 (14% of total, 27% of all resolved SSIDs)
SSIDs that mapped to multiple access points, all of them located in the same city	1 374 (21% of total, 40% of all resolved SSIDs)
SSIDs that mapped to multiple access points across different cities or municipalities	1 110 (17% of total, 32% of all resolved SSIDs)

idea of the success rate with which we were able to use the `WiGLE.net` for resolving associating SSIDs with real-world locations, we provide some statistics on `WiGLE.net` lookups in Table 5.6. These results are based on the SSIDs gathered during all experiments for which SASQUATCH was used (including installments of SASQUATCH at different conferences).

Similarly, an adversary targeting a single person does not need a highly efficient system spoofing every network in every smartphone user's list. Indeed, this type of attacker can just as well set up a fake access point by configuring his/her wireless home router to use the SSID corresponding to one of the networks in the victim's PNL, effectively causing the victim's smartphone to connect to his/her network, making the attacker an *active man in the middle* between the user's smartphone and the internet.

In our study, we were often not able to gather the full PNL of a smartphone. Because of the high density of smartphones at the university campus, and because our system tried to process data by all smartphones, some packets needed to be dropped to be able to process all data in real time. More importantly, because we were located in the main hall, many smartphones only stayed in range for a relatively short time. This causes that some probe requests were never sent while the smartphone was in range of our system.

For an adversary who targets a limited amount of people, getting the full PNL of the targets is significantly easier. Indeed, if the stream of Wi-Fi packets can be filtered to only include probe requests coming from a small number of devices, a system like SASQUATCH would need only a fraction of the computing power of our setup, and fewer packets would have to be ignored by either the network card or the firmware. Moreover, if only a few people are targeted, it is easier to make sure that these people are within range of the system for a longer time (either by moving the system towards the targets, or by having the targets stay in a single place).

5.7 Conclusion

The first goal of this study was to gauge how aware mobile device users are about the dangers inherent in using their devices on Wi-Fi networks (touching on research question **RQ3**). From our results, we can see that 76% of our participants were unaware about one or more of the issues presented by the SASQUATCH system. The security issue that surprised the most users (52%) is that an attacker is able to spoof wireless networks in the user's smartphone's PNL.

Similar to the security principle of *responsible disclosure*, our second goal was to raise awareness on the data smartphones are leaking due to misconfiguration, insecure implementations or network protocol characteristics. Making users aware of this will encourage them to expect more from smartphone manufacturers with regard to handling privacy sensitive information in their smartphones, and will allow them to take action to make their smartphones more secure.

We achieved the second goal by having people interact with a public display setup that was designed to raise awareness. To stimulate people to look at our screen, we created a "honeypot effect" by having the display show open networks smartphones tried to connect to, as well as locations for all Wi-Fi networks smartphones were looking for. Once users started interacting with the public display more actively by scanning a QR code to indicate their interest in the matter, we informed them about ways in which they can improve privacy and security on their own smartphones.

Revisiting research question **RQ2**, we showed that smartphones leaking privacy sensitive information is a very real and common problem (between 64% and 88% of devices could be mapped to at least one previous physical location) and that a significant fraction of smartphone users are susceptible to the Evil Twin attack. Our study shows that 45% of users is in direct danger, and that 52% is not aware that such an attack was possible. Our approach to raise awareness on these issues was highly appreciated by the users (59% indicated they wanted to be kept up-to-date on our developments to help improve privacy and security for smartphone users) and has proven to be highly effective for informing users (76% of users were not aware of these issues and is aware now), showing that it is indeed possible to inform non-technical mobile device users about the dangers of using their devices on Wi-Fi networks (answering the first part of research question **RQ6**).

Chapter 6

Understanding Wi-Fi Privacy Assumptions of Mobile Device Users

6.1 Introduction **83**
6.2 Related work **84**
6.3 Methodology **86**
6.3.1 Connection monitoring **86**
6.3.2 Exit survey **88**
6.4 Participants **90**
6.5 Results **92**
6.6 Discussion **96**
6.7 Conclusion **99**

6.1 Introduction

In the previous chapters, we saw how third party actors (both malicious and benign) could infer a smartphone user's whereabouts by looking at their device's Wi-Fi signals. We also showed how information from these signals can be used to mount a so-called "Evil Twin" attack, wherein a malicious actor tricks a device into connecting to its network. Both these Evil Twin networks, as well as legitimate Wi-Fi networks offered by commercial or public entities as a service to their customers allow the network operator to view metadata about the (internet-)connections being made by connected devices – and, in case these connections happen unencrypted, their data. Moreover, even networks provided by legitimate parties often lack any form of security: as we saw in the previous chapter, at least 45% of the 1404 devices encountered by the SASQUATCH system were set up to automatically initiate a connection to at least one insecure network (see Section 5.5). This allows not only the network provider, but also others within range of the network to eavesdrop on communications. Having access to this information can allow for third parties to generate a highly accurate profile of mobile device users, which entails inherent privacy risks [Troianovski, 2013].

The problem of security is worsened by the fact that not all apps are using secure methods of connecting to the internet. Indeed, if apps fail to implement proper end-to-end encryption, eavesdroppers are able to see the data sent by these apps on an insecure Wi-Fi network. In 2012, Georgiev et al. showed that even when apps are using secure (SSL) connections, they often fail to validate certificates correctly, opening the door to active man-in-the-middle attacks [Georgiev et al., 2012]. This means that providers of Wi-Fi networks would be able to intercept data, even if encryption is used. The researchers uncovered faulty certificate checking in libraries for cloud computing (e.g. EC2), web services (e.g. Apache Axis), merchant SDK's (e.g. PayPal), and ad libraries (e.g. AdMob).

In this chapter, we want to find out how aware users are about these specific issues, and how this influences their behavior when using Wi-Fi networks. Data from Eurostat shows that 48% of internet users indicated they had been limited or kept from performing an internet activity (e.g. buying goods or providing personal information to online communities) due to security concerns during the 12 months prior to a 2015 survey. However, only 13% of these users had limited their internet use because of security concerns when accessing the internet on a mobile device via a wireless connection from places other than home [Eurostat, 2016].

In the past, researchers have studied the amount of privacy and security awareness of Wi-Fi users, with one of the more notable studies being performed by Klasnja et al. in 2009 [Klasnja et al., 2009], where laptop users were surveyed about their network usage and corresponding privacy concerns. These studies show that user expectations of privacy often do not correspond to the reality, and that a person's stance towards privacy often does not correspond to their actual behavior [Norberg et al., 2007].

Our study expands on this earlier work while updating the methodology to deal with the changing technology landscape. On one hand, it aims to assess whether mobile device users

are aware of the network connections that are being made over Wi-Fi by installed applications (possibly in the background). On the other hand, it tries to find out how comfortable these users are with the fact that this application data is sent over the Wi-Fi network they are connected to, allowing it to be monitored by either the network operator, an eavesdropper (in the case of unsecured networks and connections), or both. We only consider the use of Wi-Fi networks to access the internet by mobile device users, and not the Wi-Fi connections that are used by network providers to access their backbones.

With this, we aim to get an idea of whether the principles of visibility and trusted path, as defined by Yee [Yee, 2002], are satisfied for current mobile device users. The principle of visibility states that "The interface should allow the user to easily review any active actors and authority relationships that would affect security-relevant decisions." The principle of trusted path says that "The interface must provide an unspoofable and faithful communication channel between the user and any entity trusted to manipulate authorities on the user's behalf."

6.2 Related work

This study is mainly influenced by work from Klasnja et al. [Klasnja et al., 2009], in which participants' network usage is monitored. Part of the study consisted of showing participants a list of web sites to which specific bits of personal information were sent unencrypted, asking the participants about their awareness on the transmitted information, and how they felt about it. Klasnja et al. observed that four out of the eleven participants were aware that other people could possibly access their information being transmitted over Wi-Fi, but that this understanding did not raise concerns. Our study works in a similar way, while updating the methodology to deal with the changing technology landscape where mobile devices are rapidly surpassing notebooks in usage [Lella and Lipsman, 2016]. The revised methodology uses smartphones and tablets as the main devices, and envisions to gather more quantitative rather than qualitative data with a participant pool of $N = 108$. Our approach also considers only the connection metadata (such as the originating app and the connection endpoint), rather than the actual transmitted data.

After the study by Klasnja et al., Consolvo et al. introduced the "Wi-Fi Privacy Ticker" [Consolvo et al., 2010]. This tool informs users about sensitive data being sent out over their wireless interface, while indicating whether the connection is secure. The results of their study show that the ticker helped participants to increase their awareness, and that it helped participants form more accurate mental models of the circumstances in which data gets transmitted, eventually contributing to changes in user behavior while on Wi-Fi. In our work, we try to (i) assess the level of awareness (without actively raising awareness) and (ii) determine if this (long-term) awareness has a positive impact on security habits.

A study from 2010 by Swanson et al. [Swanson et al., 2010] reported on the perception of privacy and security when using wireless networks for a group of 11 randomly selected persons. They show that users make security choices based on (often mistaken) analogies to the physi-

cal world, similar to what happens in naïve, or 'folk' physics, and that this leads to users who are confident in their knowledge about security while making unsafe decisions. They call this phenomenon *naïve risk mitigation*, providing examples of participants trusting a connection because they trust the company they are interacting with, or participants believing a malicious actor would not have the time to sort through all the data that could be gathered. Their survey also included an educational component, as it explained the associated risks of such actions to the participants. This study shows the need for concrete examples of network scenarios when surveying device users about security and privacy of their data.

Even when people think of themselves as privacy conscious, their actual behavior often does not match their intentions. Norberg et al. describe "The Privacy Paradox" as the relationship between individuals' intentions to disclose personal information and their actual personal information disclosure behavior [Norberg et al., 2007]. They find that individuals will actually disclose a significantly larger amount of personal information than their stated intentions indicate.

Like privacy stance, there does not seem to be a direct relationship between a person's technical background and the actions they take to control their privacy or increase their online security. In a 2015 study, Kang et al. use diagramming to determine users' mental models about the Internet, and conclude that individuals' technical backgrounds do not influence their privacy or security behavior [Kang et al., 2015]. Our study tries to assess whether these findings also apply to users of Wi-Fi networks, by looking for correlations between technical level and privacy intentions of the participants, and the security of networks they connect to.

In 2012, Chin et al. analyzed the confidence smartphone users had in smartphone security and privacy [Chin et al., 2012]. The study finds that participants are less likely to perform certain privacy sensitive activities on their smartphones than on their laptops, finding, e.g., a difference of 7% vs. 60% of participants not willing to entering their social security number. With the study being performed in 2012, participants cited reasons such as "new phone technology" for not trusting their mobile devices with privacy sensitive information. However, some participants also noted not trusting the Wi-Fi network, or mentioned "potential hackers hanging out in cafes". These results seem inconsistent with a Eurostat survey from 2015, where only 13% of participants had limited their internet use because of security concerns when using the internet with a mobile device via a wireless connection from places other than home (compared to 70% in total) [Eurostat, 2016]. This could indicate that the technology landscape has changed considerably since 2012, with smartphones becoming a more integral part of people's lives.

A more recent study from Clark et al. shows that users of Internet services are often unaware about which of their data is transmitted to the cloud and stored there, using GMail's attachment storage as an example [Clark et al., 2015]. They show that task-oriented users rarely stop to think about the security implications of their actions. We suggest that the same might apply to wireless network users, quickly connecting to a free Wi-Fi hotspot in order

to perform a task at hand (e.g. to get to a website containing information needed at that moment).
With these studies indicating a possible discrepancy between users' privacy attitudes and behaviors between five years ago and now, our goal is to assess to what extent the shift from personal computers and laptops to smartphones has impacted these privacy properties. Moreover, we want to find out whether the Privacy Paradox also applies to mobile device users in 2017. Our main contribution can hence be summarized as the answer to the question: with a security landscape that changed significantly since 2009, did privacy perceptions and practices change with it?

6.3 Methodology

Our study consists of two phases: first, network connections made by apps on the participants' devices over Wi-Fi are logged for 4 weeks, along with information about the networks these devices are connecting to. When connecting to a Wi-Fi network for the first time, participants are given a one-question survey asking them to provide a one-line description for the network. In the second phase of our study, the participants are given an exit survey, consisting of two parts: a survey containing personalized questions based on the gathered data, and a general (non-personalized) survey containing questions about the participant's privacy stance.
For the remainder of this chapter, we will talk about *connections* as the actual (transport layer level-) connections that are made by apps on participants' devices to a server on the internet. Associations between a mobile device and a Wi-Fi access point will be referred to as *networks*.

6.3.1 Connection monitoring

In the first phase of the experiment, participants are asked to install an Android application which does not contain any user interface, except for a welcome screen that tells the user to exit the app and to leave it installed for the duration of the study. This application runs in the background, collecting information about which Wi-Fi networks the participant connects to, and which network connections were made by apps on these networks. More specifically, the data logged for Wi-Fi networks is:

- The network name (SSID)
- Whether the network provides any security in any form (either WEP, WPA, WPA2 or WPA2-Enterprise)

Note that both WEP and WPA have been proven to have major security weaknesses [Tews and Beck, 2009], and their use has been discouraged in favor of using a more modern and secure protocol such as WPA2 for wireless security. For the sake of this study, we do not differentiate between different types of security. Instead, we will focus on networks without

any security enabled, for which we are certain that eavesdroppers are able to capture traffic that is sent in the clear.
Furthermore, for every connection made by any of the apps installed on the phone, the following data is logged:

- The app name (visible to the user) and package name (unique for every app) for the app making the connection
- Whether this app is launchable by the user, as some (system) apps run in the background without being visible to the user
- The hostname (or the IP address, in case no hostname can be resolved) and port of the connection's endpoint
- A timestamp of when the connection was made
- The Wi-Fi network the device was connected to when the connection was made

Only the connection metadata is recorded; no actual communication (i.e. messages, e-mails or any other app content) is gathered by the monitoring app.
The data is gathered by periodically reading and parsing Android's `/proc/net/tcp` and `/proc/net/tcp6` files and matching the user ID's for each connection to a corresponding app on the participant's device, only while the device is connected to a Wi-Fi network. In contrast to how the usual sandboxing model of Android is implemented, apps can view the connections made by other apps by reading these files. Both reading these files and resolving the user ID's or the names of the apps do not require any permissions to be requested by the monitoring app. The only permission that needs to be granted by the user is to view the names and properties of Wi-Fi connections, a requirement of which the participants are informed beforehand.
Since the connection monitor is only periodically logging connections (once every 15 minutes), some of the connections might be missed. Even though this only applies to connections that have been opened, closed, and passed their `TIME_WAIT` timeout of 4 minutes all within the timeframe of 15 minutes, we take our gathered data to be a lower bound on the actual number connections made. Since we only use apps, not individual connections, as part of the study, this should be representative for the actual connections made by the device. Moreover, logging happens using Android's `AlarmManager`, which is set up to log 'inexactly', causing logs to happen together with other jobs (such as another app syncing data). This mitigates the fact that logging does not happen continuously by making sure it happens at the most 'busy' moments.
Apart from logging the aforementioned data, the app also shows a mini-survey whenever the participant connects to a wireless network they never previously connected to. This survey asks the participant to provide a one-line description for the network, aiding the participant in remembering the specific network or situation during the exit survey. An example of such

a scenario is when the participant is on holiday in an unknown city, and connects to the free Wi-Fi network in a bar. In this case, he or she can provide a short summary about the place providing the network (e.g. "small bar with friendly owner next to the train station"), or about the reason (e.g. "needed to look up the location of a restaurant nearby").
Participants are given instructions for installing and activating the app on their own personal devices, and for whitelisting the app from any 'battery saver' or 'memory cleaner' apps that might interfere with its operation. The app is distributed through the Google Play Store, with its availability being limited to only participants of the study. Data collection happens for a period of 30 days for every participant, with the researchers and recruiters following up with participants to make sure the app was functioning correctly. This includes making sure the app is consistently transmitting data, indicating it was not suspended or terminated by any 'optimizer' apps, and that the aforementioned whitelisting is done properly.
At the end of the connection monitoring phase, connections made to third-party advertising servers are excluded from the dataset. The reason for this is that the connection monitor is only able to view the connections made by other apps, without being able to see the type of data that is being transmitted. Because of this, some apps might have only sent non-service specific metadata (such as device identifiers used for advertising). To identify hosts that are used for serving ads instead of content, we use the hosts file that is part of the popular AdAway ad-blocker for Android.[1] This file contains an exhaustive list of hosts that are used to serve advertisements to Android apps and to gather analytics data from users of Android apps. By applying this filter, 10.05% of a total of 5 780 105 connections are removed from the dataset, which excludes 295 distinct $(network, app)$ pairs (1.23% of a total of 24 030).

6.3.2 Exit survey

The exit survey is provided to the participants at the end of the experiment, and consists of two parts: a survey containing personalized questions based on the gathered data, and a general (non-personalized) survey containing questions about the participant's privacy stance.

6.3.2.1 Personalized questions

The first part is based on the data gathered during the first phase of the experiment. For every participant, personalized questions are generated based on a subset of both the connections and Wi-Fi networks. These questions (available in Appendix B.2) are designed to poll the participant about their privacy stance towards the data for a particular app on their phone being accessed by either the network operator or an eavesdropper (in the case of open networks). The process of selecting and generating these questions is outlined here.
For every participant, three networks for which connections occurred during the monitoring phase are selected programmatically. Our program is configured to give a preference to unsecured networks, as they allow for extra survey questions (see below). Moreover, the number

[1] The AdAway hosts file is available at `https://adaway.org/hosts.txt`.

of 'hotspots provided by internet providers', where an internet provider rolls out a nationwide network based on Wi-Fi hotspots, is limited to at most one. As hotspots are the most prevalent unsecured networks based on number of connections made (see Section 6.6), including them all while giving preference to unsecured networks would yield a set of responses that is heavily biased towards these internet service provider (ISP) hotspots. Apart from these two preferences, networks are selected completely at random.

For each of the selected networks, three apps that made a connection to a first-party server (i.e., a server not belonging to a third-party advertiser) while being connected to the network are selected, creating a total of nine $(network, app)$ pairs. Before this selection happens, a few apps are excluded from the list. These apps include web browsers, system apps, and other supporting apps (such as Google Play Services), because a participant might be unable to formulate a meaningful response to posed questions, skewing the results in the process.

Our program is again configured to select apps based on a few preferences. First, if possible, at least one connection involving a messaging app (such as Facebook Messenger, WhatsApp or Telegram) is included as part of the questions.[2] This allows to compare answers for a specific app category afterwards. Second, as was the case with the network selection, a preference is given to insecure[3] connections. For insecure $(network, app)$ pairs, one additional question is added to the survey, asking the participant to what extent they would mind an eavesdropper being able to see app data (cf. question *Q7* in Appendix B.2). Only when no more 'insecure pairs' (apps making an insecure connection on an open network) are available, secured connections are selected. Apart from these preferences, apps are selected completely at random.

Note that the previously outlined selection only applies to the questions themselves. Results listed in later sections will always apply to the full dataset (excluding connections made to advertisement networks), unless otherwise noted.

For every network, the participant is shown a small one-line summary that they provided themselves as part of the mini-survey occurring during connection monitoring (see Section 6.3.1), together with the exact time at which the first connection to this network was made. We display the connection time for the first connection (instead of a later connection) because this is the time at which the participant filled in the mini-survey, and because this is a connection that was actively initiated by the participant; later connections may have been made automatically by the device, while the first connection was made with a specific purpose in mind.

[2] The messaging apps are selected from the list of top free apps in the *Communication* category on Google Play (available at `https://play.google.com/store/apps/category/COMMUNICATION/collection/topselling_free`).

[3] We consider connections made to TCP port 80 to be unencrypted. In principle, an app could create a secure connection to TCP port 80 either by using HTTPS over this port (instead of the default HTTPS port 443), or it could implement its own secure protocol on top of unencrypted HTTP connections. Since network connection monitoring on non-rooted Android devices does not provide access to the actual data on the connection (only the connection's meta-information), and because encrypted traffic over TCP port 80 is highly uncommon [Dainotti et al., 2010], this assumption is deemed to be valid for the purpose of this study.

Even though all of these questions are based on actual connections made by apps installed on the participants' devices, they are not informed about this. Instead, the connections are presented as hypothetical scenarios, asking only about the extent to which participants would agree with this data being visible to one of the aforementioned parties. This ensures that participants' responses are not influenced by the actual data. The last question for every connection asks about how high the participant would estimate the likelihood that the connection actually occurred. In addition to this, the very end of the survey contains the same question in a more direct form, informing the participant about the fact that this connection effectively happened.
The hostname and address of the endpoint the app connected to are only used for statistics. We chose not to include this information as part of the survey because a substantial fraction[4] of these connections are made to a content delivery network or a cloud provider (e.g. Amazon Web Services) where the back end for the app is hosted.

6.3.2.2 General privacy questions

The second part of the exit survey assesses the participants' general privacy stance. For this purpose, the survey contains questions from a 2014 study conducted by Pew Research, which polls participants about their privacy and personal information [Pew research center, 2014]. More specifically, questions Q11 and Q12 from the Pew Research study are used to ask the participant about which privacy-enhancing technologies and habits they know of, and which ones they have used or carried out before. Some examples of such technologies and are "using a temporary username or email address", "encrypting phone calls, text messages or email" and "clearing cookies and browser history".
The privacy questionnaire is included in the exit survey instead of being part of the onboarding questionnaire in order to avoid influencing participants' privacy behavior during the study. Care is taken to prevent participants from knowing beforehand that the study is about privacy, instead framing the study as a more general 'study about wireless networks'. Moreover, asking the questions about the participants' privacy stance is deferred until the very end of the exit questionnaire in order to avoid influencing the answers to questions pertaining to app data.

6.4 Participants

Participants are recruited through an external recruitment organization, with the aim of having a participant pool that is as diverse as possible. The participant pool's diversity is controlled by technical knowledge, education level and demographics, assessed by the questions in Appendix B.1. Participants are offered an incentive of €15 for completing the experiment. As

[4] A cursory search shows that at least 14.92% of the logged connections belong to one of a few popular cloud providers such as `amazonaws.com` or `akamai.net`.

Table 6.1: Participant demographics.

Age group	# participants
18 – 25	31
26 – 35	43
36 – 45	18
46 – 55	7
56+	4
Undisclosed	5

Highest completed education	# participants
Master degree or higher	56
Bachelor degree	33
High school degree	17
Did not finish high school	2

explained in the previous section, the study is labeled as a general 'study about wireless networks' to prevent participants from knowing beforehand that the study is about privacy issues in wireless networks. All communication with the participants is framed accordingly.

After removing participants that did not complete the full study (either because they did not keep the monitoring app installed for 30 days or because they didn't complete the exit survey), the participant pool contains 108 Belgian people aged from 18 to 65 (a full age distribution of participants is available in Table 6.1). 30 participants identified as female, and 77 identified as male. Most participants (56) completed a master education or higher (a full distribution of the participants' education is available in Table 6.1). When asking to rate their technical expertise, most participants (44%) say they have a 'high' or 'very high' knowledge about how computer networks work, with only 13% rating their knowledge 'low' or 'very low'. This is coherent with responses for the question "Would you be able to explain what WEP, WPA and WPA2 are?", controlled by a question to effectively provide the explanation[5], where 66% of participants indicate they would be able to provide this explanation. We found a strong correlation between the declared technical expertise and the participants being able to change a Wi-Fi network's settings (Spearman's $r_s(106) = 0.48$, $p = 1.418 \times 10^{-7}$) and to participants being able to explain how Wi-Fi security works (Spearman's $r_s(106) = 0.60$, $p = 7.896 \times 10^{-12}$).

For sake of completeness, we also controlled the correlation between the declared expertise and a more strict validation of the explanations about Wi-Fi security. After adjusting the category of 15 participants that did not explain Wi-Fi security entirely correct (e.g. by describing the encryption standards as only authentication mechanisms), we can still observe the same trend in the correlation between the declared expertise and whether participants were able to explain Wi-Fi security or not (Spearman's $r_s(106) = 0.46$, $p = 4.454 \times 10^{-7}$).

[5]We manually checked the explanations of the participants, and found that all of the 71 participants who indicated they would be able to explain WEP, WPA and WPA2 were able to formulate an answer that is correct or nearly correct, with 56 of them providing the correct explanation, and the other 15 getting only some details wrong (e.g., describing the encryption standards as only authentication mechanisms).

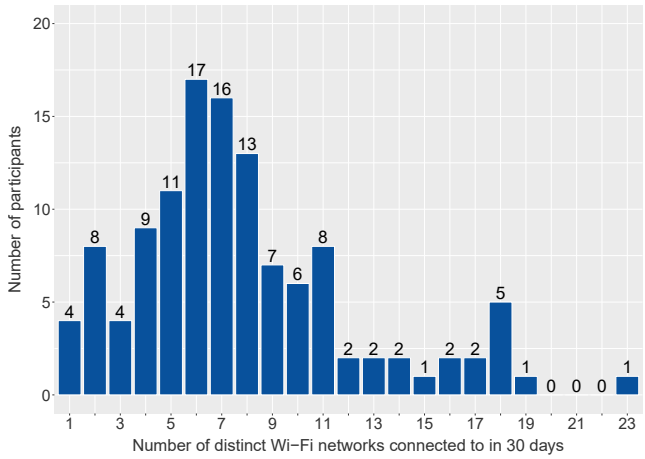


Figure 6.1: Number of Wi-Fi networks connected to by participants during the 30-day study. On average, participants connected to 8 networks, with the vast majority of them (61%) connecting to anywhere between 4-8 networks.

In conclusion, the participant pool is skewed towards relatively young (18-35 years), male, highly educated people with a high expertise in computer networks. In Section 6.6, the impact of this bias on the results is discussed in more detail.

6.5 Results

During the 30-day connection monitoring study, users connected to an average of 8.02 Wi-Fi networks, with the vast majority of users (61%) connecting to anywhere between 4-8 networks (see Figure 6.1). This shows a larger than 100% increase compared to the study performed on notebooks instead of smartphones by Klasnja et al. in 2009, where the average user connected to 4 networks during a period of 4 weeks. 310 of the 866 networks that devices connected to (35.8%) did not have any security (in the form of WEP, WPA, WPA2 or WPA2-enterprise) enabled. All of the participants connected to at least one secured network, which shows an improvement in security compared to Klasnja's study in 2009, where 4 out of 11 participants only connected to unsecured networks. Even so, as mentioned in Section 6.3.1, a network having security enabled does not necessarily mean the network is secure. Indeed, WEP and WPA have been proven to have major security weaknesses. A Pearson correlation test did not show any statistically significant correlation between the expertise of the participants and the number of unsecured networks they connected to during the experiment. Furthermore, our results show there is no statistically significant correlation

Table 6.2: Statistics for the collected data.

Number of...	Amount
Participants	108
Wi-Fi networks	866
Open Wi-Fi networks	310 (35.8%)
Distinct apps	1667
Connections	5 330 660
Insecure connections	68 8930 (12.9%)
Distinct *(network,app)* connection pairs	23 735

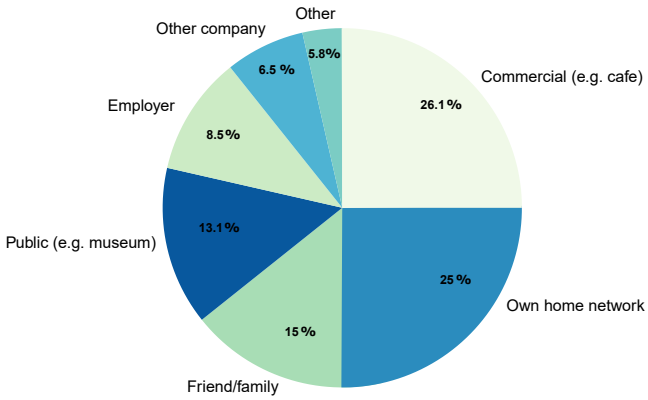


Figure 6.2: Distribution of the different types of networks in the survey. Excluded from this distribution are 51 networks having the "ISP hotspot" type, as these were purposefully limited to at most one (see Section 6.3.2 for more information). The two most prevalent types of networks that were part of the questionnaire were commercially offered networks and own home routers, together accounting for just over 50% of all encountered networks.

between participants' privacy stances[6] and the number of unsecured networks they connected to during the experiment. A general overview of the collected data is available in Table 6.2.

After accounting for the fact that the number of ISP hotspots was artificially limited in the questionnaire (by removing these 51 networks from the list of 311 networks participants were surveyed for), the two most prevalent types of networks that were part of the questionnaire were commercially offered Wi-Fi networks (belonging to e.g. a cafe or a supermarket) and own home routers, together accounting for just over 50% of all encountered networks (see Figure 6.2).

[6]The privacy stance of users is measured by the number of positive answers to the questions defined in [Pew research center, 2014].

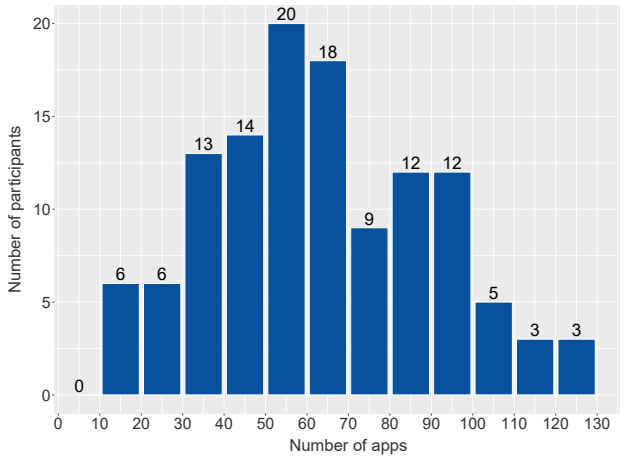


Figure 6.3: Number of apps on participants' phones making a connection to an external server during the 30-day study, grouped in bins of 10. The majority of devices (60%) have between 30 and 70 apps making a connection to the internet over this period.

Participants indicated on multiple occasions that they were unsure about the identity of networks they connected to, indicating this either as part of the general survey feedback (*"My responses on [network X] are not reliable, as I don't know this network"*) or at the moment they are connecting to the network (by giving a one-line summary along the lines of *"No idea"*, *"Unknown network"* or *"Don't know this"*).

The participants' devices had an average of 65.44 apps making a connection to an external server during the experiment, with most devices (60%) having between 30 and 70 apps connecting over this period (see Figure 6.3).

Participants were often surprised about connections being made by apps, indicating that they were unaware about 345 of 928 (38%) connections that were surfaced as part of the exit survey. Excluding messaging apps (where participants indicated awareness for 72% of the connections being made), this number grows to 264 out of 629 apps (or 42%). This also showed in the general feedback given at the end of the study, where two participants talked about a mismatch in privacy expectations and actual behavior. One participant explicitly noted *"It surprises me that Skype transmitted data, as I did not configure the app yet"*, while another user's privacy expectations did not align with what the app was actually doing, as indicated by their feedback *"When surfing anonymously, the 'incognito' mode of Google Chrome that I use regularly was seemingly not taken into account"*. Another participant explicitly indicated that the survey caused them to take action, providing as feedback: *"Interesting survey. I'll certainly remove FileExpert from my device now."*. In this regard, we found a medium correlation between the declared expertise of participants and how aware they were about the

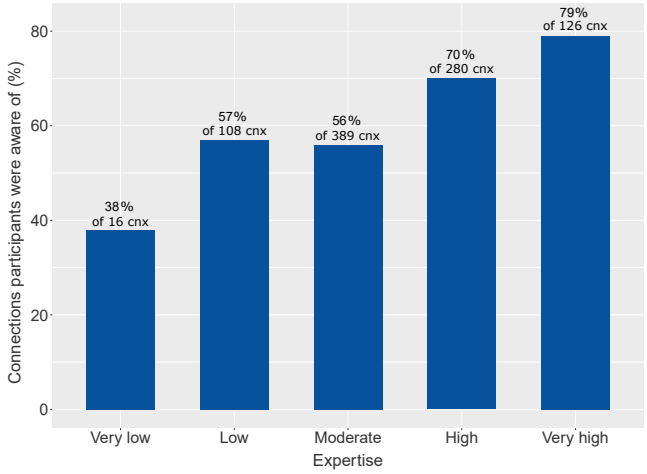


Figure 6.4: Number of times participants indicated to be aware of connections being made by the apps according to their expertise in computer networks. Participants with a higher expertise indicated to be more aware on average about the connections made by the apps installed on their devices.

connections being made (Pearson's $r(106) = 0.34$, $p = 2.09 \times 10^{-4}$). Figure 6.4 shows the distribution of the awareness count depending on the expertise of the participants.

Of the 321 insecure connection pairs ($(network, app)$ pairs containing both an insecure Wi-Fi network and an unencrypted connection made by the app), participants responded for 292 cases (91%) that they would not want a person in the neighborhood of the Wi-Fi network being able to see the data sent over the connection, stating they either *disagree* or *strongly disagree* with the statement "A random person in the neighborhood of <network name> (e.g. someone standing on the street close to the building) is permitted to see all information (see previous question) of app <appname>". This could indicate that participants were not aware about the security risks inherent to connecting to open networks, or that they were unaware about the app making an insecure connection[7] on this network.

Even when asked about the extent to which the network owner would be permitted to see data transmitted by the apps that actually transmitted data over the network, 655 out of 928 connection pairs (71%) were deemed by participants to contain data that would be too sensitive for the network owner, indicated by answering that they 'disagree' or 'strongly disagree' to the statement "The owner of <network name> is permitted to see all information (see previous question) of app <appname>". This number even increases to 88% if we only consider the 191 hotspot networks provided by ISP's, which we hypothesize could be the case because participants may be considering these hotspots as belonging to an unknown person's home

[7]Remember that connections made to advertisement networks were not used for any survey questions, as explained in Section 6.3.1.

network. The data analysis showed a small correlation between the expertise of participants in computer networks and their concern about the network owner seeing the transmitted data (Pearson's $r(926) = 0.07$, $p = 0.04$).

We also analyzed the effect of the type of data (username, password and data such as instant messages, emails or weather information) on the privacy concern level of participants in our study. To measure participant's privacy concern, we use a 5-point Likert scale for the security concern rating, where 1 corresponds to 'Not at all concerned', and 5 corresponds to 'Extremely concerned'. The Shapiro-Wilk and Barlett's tests show that the data violates the normality and homogeneity of variance requirements to perform an ANOVA test. Thus, in this case we use the Friedman non-parametric test for the comparison among the groups (considering that one participant contributes to the sample with multiple records, we use a within subjects design). The Friedman test reveals a significant effect of the type of data on the privacy concern of the participants when using a mobile app ($\chi^2(3) = 240.5$, $p < 2.2 \times 10^{-16}$). A post-hoc test using Mann-Whitney tests with Bonferroni correction shows significant differences between privacy concerns for the username and data, and username and password. Participants rated the privacy concern for the username the lowest ($avg = 1.48$, $SD = 1.06$), followed by the password ($avg = 2.21$, $SD = 1.61$), and gave the data the highest privacy concern rating ($avg = 2.30$, $SD = 1.43$).

Given the wide variety of applications that made a connection from the smartphones of the participants, we performed the same analysis considering only applications in the *Communication* category (i.e. messaging apps). As explained in Section 6.3.2, we selected at least one application of this type (if one was available) when generating the survey questions. The results of the analysis follow the same trend as those observed for the full dataset: the Friedman test reveals a significant effect of the type of data on the privacy concern of the participants ($\chi^2(3) = 241.12$, $p < 2.2 \times 10^{-16}$). A post-hoc test using Mann-Whitney tests with Bonferroni correction again shows significant differences between privacy concerns for the username and data, and username and password, with similar differences (username: $avg = 1.53$, $SD = 1.03$, password: $avg = 2.38$, $SD = 1.61$, data: $avg = 2.63$, $SD = 1.50$).

6.6 Discussion

Our results show some interesting trends, both in terms of data gathered about participants' Wi-Fi usage, as in terms of their privacy and security awareness and concerns. This section provides some general remarks and further insights derived from the results.

First, this research was purposefully limited to connections on Wi-Fi networks because of the many inherent risks involved: these networks are often operated by small businesses, and security can be lacking. However, as mobile data is getting cheaper, more smartphone users are using their cellular network to connect to the internet. This is also indicated by the study's participants, with comments such as *"I'm connecting to (public) wifi predominantly when I'm abroad, as I have a good data subscription domestically and I trust my mobile*

provider more than I trust public networks." and *"This research was about public wifi. I still use this, but not as much as I used to when I had a tablet that only had wifi. Now that I have a data subscription I use [Wi-Fi networks] a lot less. The security of mobile data connections seems more relevant to me now"*. Even so, our results show that mobile device users in 2017 are still relying on Wi-Fi for a significant amount of their internet access, with the majority connecting to 4-8 different Wi-Fi networks in 30 days. Adding to this is that a large part of Wi-Fi usage can be attributed to ISP hotspot networks, together accounting for over 94% of all connection pairs that were logged by the monitoring app.[8]

Important to note is that over one third (35.80%) of Wi-Fi networks participants connected to were insecure, allowing eavesdroppers to monitor metadata about network connections and – when the connections themselves are unencrypted – their actual data. Moreover, it facilitates malicious actors into mounting so-called "Evil Twin" attacks, where an existing (unsecured) network in the victim's preferred network list is spoofed by an attacker, causing the victim's device to automatically connect to the malicious network [Roth et al., 2008]. Combined with the fact that 12.92% of logged connections (excluding servers of advertisement agencies) were unencrypted[9], this presents a real security risk. When considering all networks (even those filtered out for the survey), the biggest offenders seem to be commercial entities (35.75% of open networks) and hotspots provided by the ISP of the home network (23.46% of open networks). This indicates that even in 2017, Wi-Fi network security and privacy should still be considered an important topic in research and in the industry.

Apart from using secure connections for their applications, it is also possible for smartphone users to protect against traffic interception attacks. However, as we will discuss in Chapter 8, VPN's come with their own set of security considerations. While we do expect a substantial number of users to have a VPN installed on their corporate devices, we do not expect the average user to be technically savvy enough to use a VPN service on their own personal devices (which were used as part of this study). It would be interesting to perform a future study, similar to this one, that looks at the number of times a VPN was used to protect any insecure connections made by the device, and how trustworthy the VPN provider is.

As is clear from Section 6.4, the participant pool is skewed towards highly educated people with a high expertise[10] in computer networks. This leads to the expectation of participants being more aware about the apps operating on their devices than the general public, and to their expertise in computer networks having a positive impact on their security habits. Our analysis does indeed show that participants reporting a high expertise in computer networks are slightly more aware about connections being made, confirming expectations. Nonethe-

[8] Because connections are only logged periodically, they provide only a representative subset of all connections made by the device. See Section 6.3.1 for more information on why this is still relevant.

[9] As explained in Section 6.3.2, we consider connections to HTTP port 80 to be insecure for the purpose of this study.

[10] Even though this expertise is self-reported, it was cross-checked with their ability to explain network security. See Section 6.4 for more information.

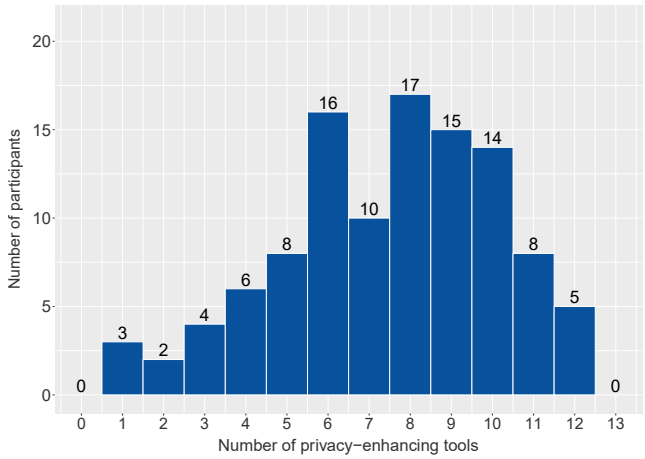


Figure 6.5: Number of privacy-enhancing tools and methods used by participants of the study. The majority of participants (53.70%) had used between 6 and 9 of these tools and methods in the past.

less, even participants reporting a 'high' or 'very high' expertise in computer networks were still unaware about 36.70% of connections made by apps on their device.

More surprising is that this expertise in computer networks does not seem to translate to better security practices: having this expertise did not prevent participants from connecting to as many unsecured Wi-Fi networks as less technically experienced participants. This confirms prior results from Friedman et al. [Friedman et al., 2002], who found that high-technology participants did not necessarily have better security habits than those with a less technological backgound.

Similarly, the privacy stance of participants (measured by usage of privacy-enhancing tools used and actions taken in the past, depicted in Figure 6.5) did not have any significant influence on the number of unsecured Wi-Fi networks they connected to. This indicates that, while users are prepared to install or use privacy-enhancing tools in a one time set-it-and-forget-it manner, they often lack the time or motivation to be continually aware about the privacy and security risks of using public Wi-Fi networks. This is in line with results from Dourish et al. [Dourish et al., 2004] and Klasnja et al. [Klasnja et al., 2009], where it is discussed that task-oriented users do not generally think about these issues when they are going about their work. Rather, they choose to delegate responsibility for security to tools, other individuals or institutions.

This mismatch in privacy stance (i.e. the number of privacy-enhancing tools used in the past) and actual current behavior directly translates to participants' perceptions about transmitted data: when confronted with specific scenarios about unencrypted connections that occurred on insecure networks (worded as a hypothetical scenario, but in reality corresponding to ac-

tual insecure and unencrypted connections that occurred on the participant's device), 91% of participants indicated being worried about the corresponding data being available to eavesdroppers. Furthermore, participants are most concerned about the privacy of actual app data, even more so than they are about the privacy about the app's password. This result seems to confirm the Privacy Paradox, as participants are transmitting a significantly larger amount of data over insecure channels than they intend to.
It would be interesting to see whether the types of apps that are used in more insecure scenarios handle less privacy-sensitive data than those used in high-security scenarios. We did not explicitly label apps in our dataset as having access to 'privacy-sensitive' data, as doing so would be inherently subjective, depending on for example the cultural background of the researchers. While the privacy sensitivity of the data itself was not the focus of this research, we consider it an interesting area for future work to use the permission accesses granted to apps as a proxy for the data they are able to access. This would allow to investigate possible correlations between the amount of sensitive data an app handles, and the security of the app's connections.
Asking the participants about their privacy-sensitiveness to the different types of information that could be transmitted by an app beforehand pushed them to think about what this data could comprise before answering questions about network owners or eavesdroppers having access to this data. Together with the previous results, this demonstrates that using very specific, personalized scenarios (instead of more general ones) might prove to work better to inform mobile device users about security and privacy issues.

6.7 Conclusion

In this work, we sought to understand mobile users' privacy and security assumptions when connecting to Wi-Fi networks. For this purpose, we conducted a study with 108 participants, monitoring the Wi-Fi networks they connected to and the connections made by apps on their device for a period of 30 days. After the monitoring phase, we asked them a number of questions about the gathered data, polling for both awareness and privacy sensitiveness on the data that was sent. With this, we assessed to what extent their privacy and security stances corresponded with actual behavior.
Despite the trend of mobile users using cellular data for internet access, our results show that even in 2017 usage of Wi-Fi networks is very popular. There is even a noticeable increase in Wi-Fi popularity compared to 2009, with users on average connecting to 8 different Wi-Fi networks in a 30-day period. Coming back to research question **RQ2**, over one third of these networks, and 13% of connections made by apps on user's devices, are insecure, which poses a major security risk to the average mobile device user. Moreover, touching on research question **RQ3**, 38% of connections are made by apps without the user being aware.
We show that, even though participants with a higher expertise in computer networks are more likely to be aware about the connections made by apps on their device, this does not

translate to better security practices: technologically savvy participants are just as likely to connect to insecure networks as people who do not have a high expertise in computer networks (providing our answer to the second part of research question **RQ6**). Similarly, the usage of privacy enhancing technologies (such as a browser plugin providing tracking protection) in the past does not have a significant impact on security behavior on Wi-Fi networks. This confirms previous studies which state that users are inclined to see security as a set-it-and-forget-it problem, where they delegate security to a tool or a different entity, not thinking about it when trying to accomplish a specific task later on. It also leads us to conclude that research question **RQ4** should be answered negatively.

This is a problem that is acknowledged by our participants: for 91% of data that was found to be transmitted in an insecure fashion on their devices, participants indicated they were worried about eavesdroppers being able to view this data. We confirm the Privacy Paradox by showing that participants are transmitting a significantly larger amount of data over insecure channels than they intend to, noting that they were unaware about 38% of these connections being made. Furthermore, we expect that there are large benefits in using very specific, personalized scenarios to inform mobile device users about security and privacy issues.

Based on our results, we will provide recommendations to network providers, developers and users in Chapter 8.

Chapter 7

Exploring privacy-sensitive decision making in Android using in-context surveys

7.1 Introduction 103
7.2 Related Work 105
7.3 Methodology 107
7.3.1 Designing the Surveys 107
7.3.2 Recruitment and Incentives 110
7.3.3 Ethical Considerations 110
7.3.4 Limitations 110
7.4 Technical implementation 112
7.4.1 App Installation and Removal Triggers 112
7.4.2 Permission Change Triggers 113
7.4.3 Generating and Surfacing Surveys 113
7.5 App Decisions 115
7.5.1 Data Summary 115
7.5.2 App Installs 116
7.5.3 App Removals 118
7.6 Permission Decisions 120
7.6.1 Permission denials 120
7.6.2 Permission Grants 125
7.6.3 Other influences 128
7.7 Conclusion 128

7.1 Introduction

Apart from the security-related privacy issues that we studied awareness for in the previous chapter, privacy sensitive information can also be collected in a legitimate manner by apps on users' devices. For example, while the SASQUATCH system described in Chapter 5 used peculiarities in the technical implementation of protocols in order to guess the past locations of passers-by, apps on users' devices are able to request direct access to the user's location and collect this information with the user's consent. In this chapter, we study how users make decisions during the lifecycle of an app on their smartphones, including deciding to install an app, making choices about whether or not to give an app access to personal data, and potentially uninstalling the app.

There are many factors that could commingle to bring users to a decision. Part of the thinking around these decisions may involve reasons related to privacy, such as sensitivity to sharing particular types of data, trust in the developer, understanding the value added when personal data is shared, and many more [Harbach et al., 2014; Harris et al., 2016, 2015; Lin et al., 2014]. In order for an app to access personal data, both Android and iOS adopt a runtime permission model, which allows users to decide whether to grant a given permission request at the time when it is first needed within the app. In this chapter we explore users' rationales for decision making during these three parts of an app's lifecycle, but with a focus on how users decide about permissions. Importantly, we study users' rationales at the moment they make their decision.

A large body of work has focused on understanding users' attitudes, comfort and their comprehension about permissions [Almuhimedi et al., 2015; Balebako et al., 2013; Felt et al., 2012c; Kelley et al., 2012]. However, almost all prior studies were conducted by using the permission model in which users had to accept or deny all the permissions requested by an app at installation time, without the possibility to grant permissions individually (for versions of Android before 6.0). A series of notable findings by Felt et al. [Felt et al., 2012c] and Kelley et al. [Kelley et al., 2012] showed that few users pay attention to install-time permission dialogs and even fewer understand them. Furthermore, results from other studies [Almuhimedi et al., 2015; Balebako et al., 2013; Felt et al., 2012c] indicated that users are often unaware of many permissions they have already granted to their apps. Subsequently, researchers started to advocate for a more contextualized permission model that would allow users to control permissions on a more granular level [Felt et al., 2012a; Nissenbaum, 2004; Wijesekera et al., 2015].

Android adopted the runtime permission model starting in version 6.0. There are at least two reasons why runtime dialogs have the potential to improve decision making by providing context. The first is that they often (but not always) clarify to the user why a permission is needed by linking it to the functionality that is triggered, because permissions are requested at the moment the data access is needed. The second is that developers can enrich the informa-

tion shown in the permission request by providing their rationale[1], which can be considered as additional contextual information. While some developers take advantage of this, many still do not.

Given that most prior results were obtained for the old permission model, it is unclear to what extent they are still applicable to the current runtime model. In this work, we conduct the first study, to the best of our knowledge, that examines the reasons why Android users install or remove an app *at the time this happens*, and the motivation behind granting or denying a permission *right after users make their choice*. We are also able to examine users' reasons for each permission group, thus exploring if their reasoning differs when deciding on *location*, *microphone*, *contacts*, and other types of personal data. We capture users' comfort level with their choice both at runtime as well as after the study, which allows us to compare their comfort levels with their decisions both in-context as well as after the fact. Finally, we explore whether other factors, such as demographics, may influence user decision making. Although there exist prior works that studied users' permission choices with the runtime model [Lin et al., 2014; Liu et al., 2016, 2014; Wijesekera et al., 2017], their goals were not focused on users' rationales.

In order to answer these questions, we employed an open-source Android app called "Paco" (Personal Analytics Companion) [Evans, 2016], which is a platform for ESM (Experience Sampling Method) studies, which is a method that has been widely used for mobile privacy studies in the past [Anthony et al., 2007]. We extended Paco to be able to query users about the reasons behind the decisions they make on their Android device related to app installs, permission decisions, and uninstalling apps, and made these extensions available to the broader research community. Paco allows us to capture the rationale behind users' actions in-the-moment, when it is fresh in their minds, therefore preserving as much of the actual context as possible. The 157 participants in our study installed Paco on their personal phones and used it for a 6-week period without any interaction with us. We collected over a thousand permission decisions and the associated reasons. Our study is the first, to the best of our knowledge, to collect such data in the wild.

Our main findings include the following. Many of our participants, when deciding about permissions, are indeed thinking about whether or not the permission is needed for the app or for a specific functionality, and whether the app "should" need it. This suggests that the context provided via runtime permissions appears to be helping users make decisions. Our participants accepted 84% of all permission requests, and among those they indicated they were comfortable (right after deciding) with their choice 90% of the time. The remaining 10% of grant decisions have a low comfort score, which suggests that a form of reluctance can occur when granting permissions. When we asked participants at the end of the six week period about some of their decisions, participants were not at all comfortable with 29% of them. We also noticed that the permission denial rates vary across different permissions.

[1] `https://developer.android.com/training/permissions/requesting.html#perm-request`

For example, microphone permission requests were denied almost twice as often as storage permission requests.
We identify decision rationales for 4 events types (app installation/removal, permission grant/-denial) from Android users and rank them according to participant feedback. One of the most common reasons for denying permissions was that users know they can change it later. We further break down the reasons for denials per permission type and find that the dominant rationale for each permission type can differ – sometimes significantly – across permission types.
The remainder of this chapter is organized as follows. We discuss the related work in Section 7.2, we introduce the methodology of our study in Section 7.3, and we detail the implementation changes to the Paco app in Section 7.4. We present the results about users' rationale for app installs and removals in Section 7.5, and we discuss the findings about permission grant and deny decisions in Section 7.6.

7.2 Related Work

Existing research has explored the space of Android permissions and privacy from two perspectives, that of users and developers.
From the user perspective, research has shown that few people actually read application permission requests and even fewer comprehend them [Felt et al., 2012c; Kelley et al., 2012]. In fact, users were often surprised by the abilities of background applications to collect data [Jung et al., 2012; Thompson et al., 2013], and they were concerned when presented with possible risks associated with permissions [Felt et al., 2012b].
To enhance the user experience, some have suggested providing users with more privacy information and personal examples to improve comprehension [Harbach et al., 2014; Kelley et al., 2013]. Researchers have designed systems to identify privacy violations and to reduce them by recommending applications based on users' security concerns [Almohri et al., 2014; Enck et al., 2010; Gibler et al., 2012; Hornyack et al., 2011; Klieber et al., 2014; Xu et al., 2012; Zhang et al., 2013; Zhu et al., 2014]. Resource requests have been categorized into benign and dangerous requests, so that only the dangerous ones require user approval, thereby reducing the number of privacy/security decisions a user needs to take [Felt et al., 2012a]. Some studies employed crowdsourcing to learn user expectations and to create privacy models [Lin et al., 2012], and others explored creating personalized assistants [Liu et al., 2016].
The research focused on developer behavior has shown that many developers are not deeply knowledgeable about permissions and often misuse them [Stevens et al., 2013]. Intentionally or unintentionally, they are often making mistakes [Shklovski et al., 2014; Smith, 2016] and are not following the principle of least privilege [Wei et al., 2012]. To identify this overuse behavior, tools have been developed that employ natural language processing of application descriptions [Pandita et al., 2013], and static and dynamic analysis of Android apps [Au et al.,

2012; Bodden, 2013; Felt et al., 2011; Spreitzenbarth et al., 2013]. Further research efforts [Enck et al., 2010; Gorla et al., 2014; Sarma et al., 2012] that design methods to generally identify malicious applications have leveraged permission overuse assessments.

To improve the situation, researchers have suggested reorganizing the permission model with better definitions and hierarchical breakdowns [Barrera et al., 2010], or adding fine-grained access control for better policing [Bugiel et al., 2013]. A recent study by Micinski et al. suggests there should be a difference between permission accesses that happen in the background and those that happen interactively (where the access directly corresponds to a user interaction, such as when the user imports their contacts). While the former should be granted explicitly (and regularly notified to the user), the latter should be avoided to prevent user fatigue [Micinski et al., 2017]. Tools have been developed that dynamically block runtime permission requests [Shebaro et al., 2014], or that give users the ability to deny data to applications or to substitute user data with fake data [Hornyack et al., 2011].

We focus on three existing pieces of research that are closest to our work. In their 2013 work on Android install-time dialogs, Kelley et al. [Kelley et al., 2013] examined the extent to which the design and type of information displayed in the dialogs helps users to choose which apps to install. Both our study and theirs ask participants about factors (such as developer, popularity, reviews, etc.) that influence their choice of which app to install. Interestingly, we find different results in terms of the ranking of factors (as shown later in Section 7.5.2). We believe this may come from the different methods of testing, as well as the pre-Marshmallow[2] (theirs) versus post-Marshmallow (ours) permission model. A key difference between their study and ours is that they asked users to choose between pairs of apps for a friend (hypothetical scenario), whereas in our study users choose their own apps, in the wild, on their own devices.

Wijesekera et al. explored permissions in Android in two different studies [Wijesekera et al., 2015, 2017]. These studies explored how a contextualized permission model, based on the principle of *Contextual Integrity* [Nissenbaum, 2004] and work by Felt et al. [Felt et al., 2012a], could improve dynamic permission granting. Both these studies rely on a custom version of Android 5.1.1 (pre-Marshmallow) as the study instrument, that logs every sensitive API access that requires a permission. Their first study [Wijesekera et al., 2015] in 2015 measures how often and under what circumstances smartphone applications access protected resources regulated by permissions. They collected data on phones of 36 people about permission accesses when they happened. At the end of the week, they interviewed people, showed them screenshots of when data had been collected, and asked them if they would have liked to have prevented it (if they had been given the choice). They found that participants wanted to block 1/3 of permission accesses, citing privacy concerns over the data and lack of context about why the app needed the permission to perform its task.

In [Wijesekera et al., 2017] the authors design a classifier to predict users permission decisions. The prediction takes into account context and generates predictions not only on-first-

[2]"Marshmallow" refers to Android version 6.0

use, but also on subsequent uses when the context may be different. They postulate that users may not always elect to make the same decision about a permission each time it is used. They also make predictions as to when a user might change their mind, so that they do not ask on each use, but only on key ones where a user's decision may change (e.g. because of a different context). They used their predictor in a user study with 131 people and showed that they can do a far more accurate job of capturing user preferences this way than with the ask-on-first-use model. ("Ask-on-first-use" corresponds to runtime dialogs in versions of Android 6.0 or higher.) This work is very different from ours in that we do not build predictive models, and we are focused on understanding user *rationales* for decision making in the "ask-on-first-use" model. Our study also differs from all of these previous works in that we capture data "in the wild", meaning our participants used their own phones, their own choice of apps and interacted with their apps whenever they normally would.

7.3 Methodology

To capture users' reasoning when making privacy impacting decisions at the moment these are occurring, we use the Experience Sampling Method (ESM) [Hormuth, 1986; Larson and Csikszentmihalyi, 1983]. This method consists of asking individuals to provide systematic self-reports about their experience at random occasions during the day without the individual expecting it, often aiming to capture candid, in-the-moment experiences, and has been widely used for mobile privacy studies in the past [Anthony et al., 2007]. Our methodology consists of surveying users at the time they are making privacy impacting decisions, by surfacing a survey when the participants install or remove an app, or when they change an app's permissions. We use the Android app Paco [Evans, 2016], which is part of an existing platform for ESM studies, and which can be downloaded from the Google Play store, as our study instrument.

In addition to the in-situ questionnaires, we ask participants to fill out an exit survey. This exit survey was used to gauge participants' privacy behaviors and technology savviness, and their awareness about permissions granted to apps on their devices. It also assesses how comfortable participants are with the permission decisions they made in the past.

Similarly to Wijesekera et al. [Wijesekera et al., 2015], and our own study in the previous chapter, we avoid priming participants beforehand by publicizing the experiment as a study on app interactions in order to limit response bias. No mention of privacy is made at any point during the study, except in the exit survey.

7.3.1 Designing the Surveys

We now describe the process we followed to design our in-situ and exit surveys (provided in full in Appendix C).

7.3.1.1 In-Situ Surveys

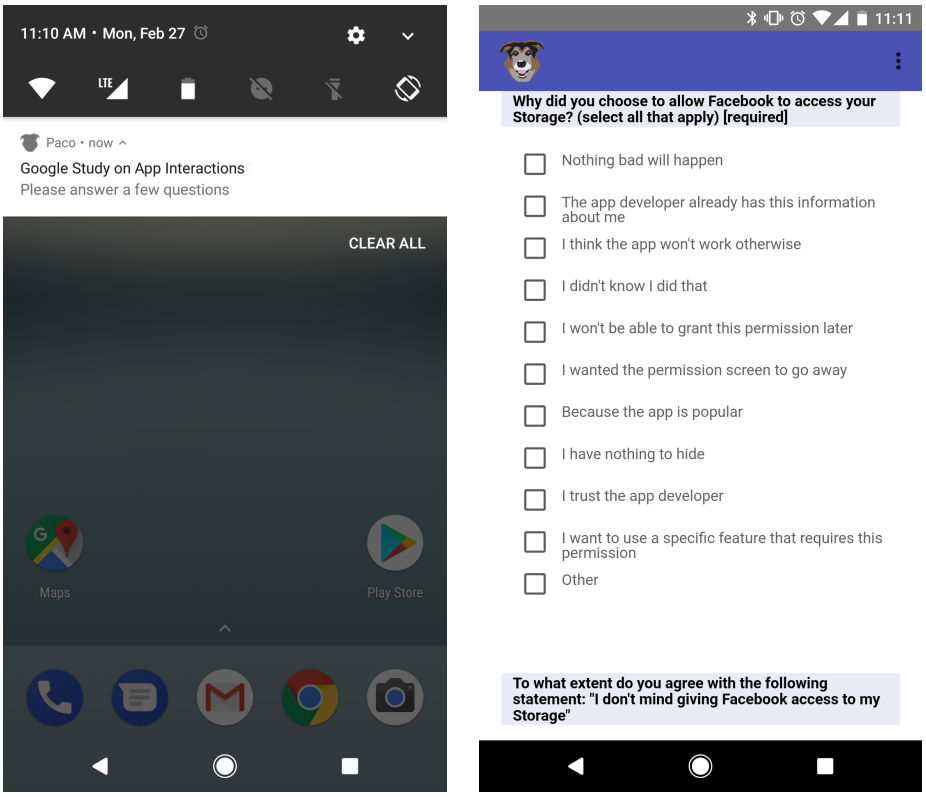
The in-situ surveys are surfaced when one of the following four events occurs: the participant installs an app, removes an app, grants a permission to an app, or denies a permission to an app. In each of these cases, the participant is asked a question about his/her rationale for performing the action. In two cases, the participant receives a second question. After permission grant events, our second question aims to assess the participant's *reluctance* when allowing the permission, by asking to what extent they agree with the statement "I don't mind giving <app> access to my <permission>". App installation events also cause a second question to surface (after asking about rationales) that asks about the factors – such as app rating or popularity – that influenced their decision to install the app.

To capture the participant's decision rationales, we designed multiple-choice questions with the option to select multiple reasons, and with an additional "Other" choice allowing a free-form response. To ensure we have an exhaustive list of possible reasons, we first performed a short pre-study through the Google Surveys platform (GS), formerly known as Google Consumer Surveys (GCS). For each of the in-situ questions, we ask a random sample of 1000 participants about their reasons for performing a recent action. For instance, we asked "The last time you <did X>, what were your reasons for <doing X>?". We coded the different responses as follows. Initially two coders each coded half the responses and then cross-checked their responses. With over 90% overlap, they then independently completed the rest. The third coder independently coded responses using labels from the first two. Complete agreement was reached by all coders. Finally, we grouped answers with similar labels, and extracted the most representative answer from each of the top-10 largest groups.

Figure 7.1(a) shows how a participant is alerted that there is a question to answer, and Figure 7.1(b) shows a sample question for a permission grant request. In order to remove positional bias in the answers, we randomized the order in which the answer options were shown - with the exception of the "Other" option, which is always placed last. In order to reduce participants' response fatigue, we limit the number of questions that are surfaced to at most 3 permission events, 2 app install events, and 1 app removal event per day, with a maximum total of 5 events per day.

7.3.1.2 Exit survey

In the exit survey, we question participants about their privacy behaviors, by asking about which privacy-enhancing practices they have employed in the past (compiled based on a Pew research survey [Madden and Rainie, 2015]). Additionally, we ask participants to rate themselves on a 5-point scale from early to late adopters of new technology. Apart from these general questions, the exit survey also contains a personalized component. In this part, we ask participants about how comfortable they are with certain apps on their devices having access to a specific permission. These <app, permission> pairs are generated for each participant individually, by inspecting what permissions have already been granted for apps on their



(a) Notification informing that a survey is available.

(b) Survey question soliciting the reasons for granting the Storage permission to an app.

Figure 7.1: Example of an in-situ survey in the Paco app.

devices. These apps are not limited to the ones for which a permission is granted or denied during our study; they also include apps that were installed prior to enrolling in our study.

The personalized questions are worded as hypothetical scenarios, asking for example "How comfortable *would you* be with the <app name> knowing who is calling you". Moreover, the questions do not directly ask about the permissions, but rather about specific data access that this permission entails. For example, instead of asking about how comfortable the participant is with an app having storage access, we ask how comfortable they would be with the app being able to access pictures on their device. When answering such a question, participants are not informed that we selected a <app, permission> pair that exists on their devices. For each of the four permissions – Location, Contacts, Phone and Storage – we select a random app for which the permission was enabled (if available), and generate the corresponding question.

7.3.2 Recruitment and Incentives

Participants were recruited via our company's external U.S.-wide participant database and were sent a screening survey via email. We screened for participants using a device running Android version 6.0 or later, with their device locale set to "English - United States". (The latter requirement is needed because of the way we implemented our changes to Paco, see Section 7.4.2.) Participant diversity is controlled for gender, age, education and employment. Participant demographics are available in Table 7.1. After the recruitment phase, participants were informed that they would be required to install the Paco app. They were made aware about the fact that this app monitors their device usage to show survey questions, and were shown a list of all the data collected by Paco. Participants were told that for each of the 6 weeks they participate in our study, they would earn $10 and that submitting the exit survey would earn them an additional $20.
We recruited a total of 193 participants. Of these 193, 34 never finished the setup process and 2 voluntarily dropped out, so they are not included. The other 157 participated for the entire 6 weeks. Thirteen out of the 157 participants did not answer the exit survey, and have been excluded from parts of our analysis relying on exit survey data.

7.3.3 Ethical Considerations

In compliance with ethical training guidelines at Google (where this study was performed), we ensured that participants' anonymity and privacy were respected. We thus carried out the following. First, all researchers that participated in this work have been trained in ethical user research prior to this study. Second, there was an informed consent process where the participants were informed of all the types of data being collected before they start the experiment. Third, we deleted all the participants' personally identifiable information after the data collection period and thus did not use any of it in our analysis. Fourth, respondents had the option to exit the study at any point in time. Fifth, only the data from participants who completed the entire 6 week study is used in our analysis (data from the 2 who stopped participating is discarded). Lastly, as will be explained in Section 7.4, we implemented end-to-end encryption on top of Paco to make sure that all gathered data would be available only to the participants and the experiment organizers (and not, for example, to operators of the Paco service or other parties).

7.3.4 Limitations

Our analysis is based on participant self-report data, which is subject to biases such as social desirability and recall. Participation in our study requires installing our study instrument (Paco) and enabling *accessibility* and *app usage* permissions (see Section 7.4.2), hence our results could be skewed towards participants willing to do so; those unwilling to do so may have characteristics we did not discover. We try to limit such an effect by recruiting a diverse

Table 7.1: Participant demographics

Gender	Participants
Male	79
Female	78

Age	Participants
18 - 23	29
24 - 30	44
31 - 40	35
41 - 50	23
51 or over	26

Education	Participants
Up to High school	15
Some college (1-4 years, no degree)	40
Associate's degree	28
Professional school degree	5
Bachelor's degree	51
Graduate Degree	18

Employment	Participants
Arts & Entertainment	8
Business & Finance	6
Education	8
Engineering	12
Health Care	12
Human Resources	2
Information Technology	14
Management	19
Miscellaneous	15
Religion	3
Retail & Sales	17
Retired	5
Self-Employed	6
Student	18
Undisclosed	5
Unemployed	7

participant pool (controlled for gender, age, education, and employment) and by explaining upfront about all the types of data collected. Only 2 participants, out of 193, voluntarily dropped out of the experiment expressing concerns around the accessibility permission usage, so the effect is indeed limited. In order to limit the leading effect of our in-situ questionnaire

towards participants' future actions on permission decisions or app installs, we imposed upper thresholds for the number of such questionnaires, which averaged at only 30 surveyed events per user over a 6-week period.

7.4 Technical implementation

Our main survey instrument, the Paco app [Evans, 2016], acts as a behavioral research platform, which allows researchers to survey participants either at predefined intervals or whenever a specific action (such as an app install or permission-related decision) occurs. The advantage of using such an app is that we do not require participants to possess a rooted Android device.

Since Paco did not provide triggers for app installation or permission change events at the time of our study, we extended its code to provide such functionality. Moreover, to ensure that the participants' data is protected while in transit between the device and our servers, we also added end-to-end encryption to Paco. All code changes to Paco were submitted and accepted to the main project, and are now available to other researchers and the general public (Paco GitHub at `https://github.com/google/paco/`).

In addition to extending the Paco platform itself, we also modify the way in which surveys are shown to the participants by making use of Paco's scripting functionality. We discuss these implementations below.

7.4.1 App Installation and Removal Triggers

To identify the moments when a participant installed a new app, or when they removed an app from their phone, we listen for `ACTION_PACKAGE_ADDED` and `ACTION_PACKAGE_REMOVED` intents broadcast by the Android system's package installer, while making sure that these events are not part of a package update (by checking whether the `EXTRA_REPLACING` parameter is set). For both events, we store both the package name of the app and the user-friendly app name (henceforth referred to as app name). The package name is a text string unique to each application on the Google Play store, and is useful for our analysis, whereas the app names are more identifiable and are used in generating survey questions (see Section 7.4.3). An example package name is `com.rovio.angrybirds` and its app name is *Angry Birds*.

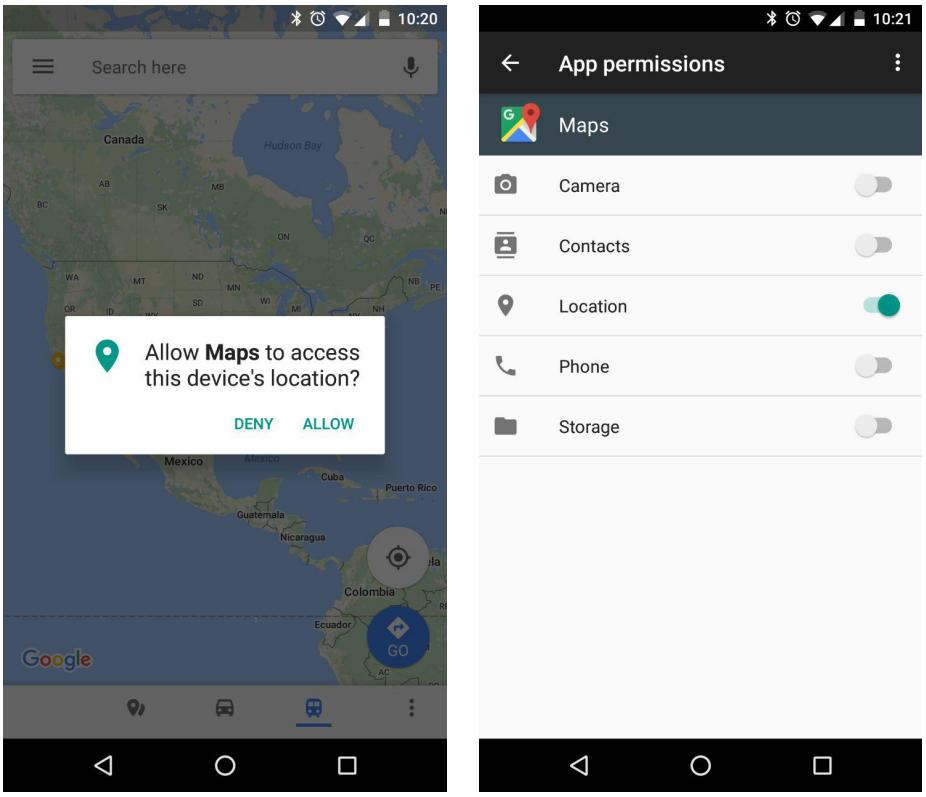
In case of an app installation event, the app name is available by querying the Android's package manager using the package name of the app. Since information about removed packages is no longer available in the package manager after an app is removed, we also manage a separate cache of package names and their corresponding app names. This allows us to access app names even after an app has been removed.

7.4.2 Permission Change Triggers

For permission change events, no intent is broadcast by the Android system, requiring us to monitor these permission changes ourselves. One obvious way to perform this would consist in periodically checking which permissions are granted to each of the apps installed on the user's phone, and looking for any changes in this information. This could be done by polling the Android package manager's `getInstalledPackages()` method and passing the `GET_PERMISSIONS` flag. However, a problem with this approach is that we would only detect permission *changes*, missing the case where the user has made a decision to remain in the same state as before. For instance, a user could deny a permission when it has not been granted before (permissions are set to deny by default when installing an app).
Because of the previous limitation, the permission change trigger is implemented as an accessibility service, which is used in Android to provide services (such as screen readers or special input devices) to people with disabilities. Because an accessibility service is able to inspect all text and user interface (UI) dialogs that are presented to the user, implementing such a service allows to analyze the text that is currently on the screen. We implement our own accessibility service to listen for events that correspond to the UI elements used for changing permissions. We then extract the text from these dialogs to determine the type of the permission and the app. We limit the accessibility service to only capture events from the `com.google.android.packageinstaller` and `com.android.settings` packages (which covers both the runtime permissions dialogs and the permission screen in the Android settings menu). This makes sure that our service does not needlessly slow down the system, and that it respects the participant's privacy by not collecting data beyond what is needed.
To identify the app for which a permission change event occurred, we query Android's usage statistics manager (this requires the *app usage* permission), determining the last active app that could have triggered a permission dialog to be shown. Because background services in Android are not allowed to request a permission, a permission dialog must always belong to the last active foreground app (if the package installer itself is excluded).
Two different cases of permission change events are considered. The most common case is the one where an app requests a permission at runtime, either when it is first started or when the user wants to use a specific feature requiring the permission. An example of this case is depicted in Figure 7.2(a), where the "Maps" app requests the Location permission. The second case is where the user actively changes an app's permission, by navigating through the Android's settings menus to either the screen containing all permissions for an app (see Figure 7.2(b)), or to the screen containing all apps that request a specific permission.

7.4.3 Generating and Surfacing Surveys

Paco allows to override the way in which surveys are generated and shown to participants, by providing experiment organizers with the ability to write scripts that will be used for generating both the notifications and the actual survey. For this study, we extensively make use of this



(a) The "Maps" app requesting the Location permission at runtime.

(b) Permission toggles for the "Maps" app in Android's settings.

Figure 7.2: Android's different methods for modifying an app's permissions.

functionality to dynamically generate questions. First, Paco's scripting functionality is used to comply with the study requirements for the in-situ questions outlined in Section 7.3.1.1. This includes overriding how often (and for which events) the user is notified, and randomizing the order of all survey responses except the "Other" option.

Furthermore, instead of relying on a predefined set of static questions, we generate them dynamically in order to provide more context to the participant (since the generated survey questions could be answered after a short time gap). For example, instead of asking "Why did you choose to allow the permission just now?", the participant is asked "Why did you choose to allow Maps access to your Location?".

Finally, the exit survey is also offered through Paco. This survey, too, depends heavily on dynamically generated questions. As discussed in Section 7.3.1, users are asked about how comfortable they are with their apps having access to data associated with a specific permission. These questions are generated for different <app, permission> pairs, where the

Table 7.2: Type and frequency of the different events considered by our study, and the number of events for which a participant was surveyed. See Section 7.3.1.1 for an explanation on survey limits.

Event Type	Occurrences	Surveyed
App Installs	3118	1913
App Removals	1944	775
Permission Grants	2239	1605
Permission Denials	437	272
Total	7738	4565

permissions have already been granted for the app by the participant. For this purpose, the Paco app is extended with the functionality to pass on a list of all apps and their associated permissions to the script that is generating the surveys. This script selects one app for each of the four chosen permissions and generates the questions accordingly.

7.5 App Decisions

7.5.1 Data Summary

We track four events in our study: app installs, app removals, permission grants, and permission denials. The total number of events that we recorded in our study are shown in Table 7.2. As mentioned in Section 7.3.1.1, we enforce limits on the number of events we survey each day. As a result, not all recorded events are surveyed. Our 157 participants triggered 3118 app install events (of which 1913 are surveyed), and 1944 app removals (of which 775 are surveyed). The apps could have come from either the Google Play store or from other sources. On average each participant installed 20 apps and removed 12 apps during the 6 week period. We note that a participant can install and remove the same app multiple times, and each of these actions would be recorded as a separate event. An app removal event could have occurred for an app that was installed prior to our study, and thus does not necessarily correspond to one of the app install events we observed.

We clarify that the Paco tool recorded all events (not only those surveyed) for all of the 4 event types that occurred on participants' phones during the 6 week period. Based on the complete set of user permission decisions, we observed an overall grant rate of 84% and a denial rate of 16%. Due to our self imposed limits on the number of surveys shown per day, we ended up asking survey questions for 72% of the grant events and 62% of the denial events. For the surveyed responses, we find the grant rate to be 86% (with corresponding denial rate of 14%). Thus the grant and denial rates of our surveyed (i.e., sampled) events is very close to the rates for the total occurrences. Out of the 157 participants, 144 answered the exit surveys.

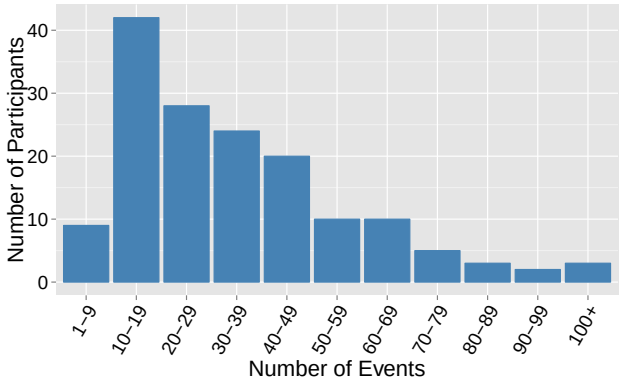


Figure 7.3: Event distribution across Participants

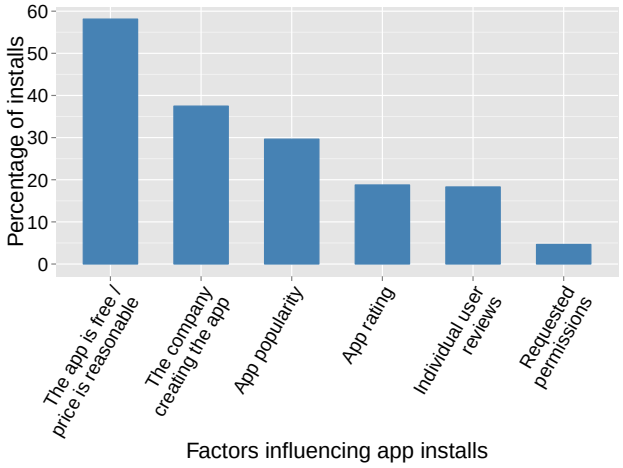


Figure 7.4: Factors impacting app installation (multiple responses per installation event are possible)

In the rest of the chapter, we present results for the surveyed events to ensure consistency with results about participant responses.
In Figure 7.3, we show the activity level of our participants with our surveys. Most answered at least 10 surveys, and some have answered many more.

7.5.2 App Installs

After installing an app, our participants were asked to select which factors (all that apply) influenced their decision to install the app. These results are shown in Figure 7.4. As expected,

we observe that price is the dominant factor. What is somewhat surprising is that the company creating the app (i.e. the developer) is the second highest factor, even more important than an app's popularity. Among these six factors, permissions occur the least frequent, and only directly affect 5% of app installation decisions. This is not surprising, because with the runtime permissions model participants do not see the permission requests during the installation flow[3], and thus users are unlikely to think about permissions at that moment. However, these install events – when participants selected permissions as a factor – came from 33% of our participants; this indicates that permissions influenced one third of our participants at least once during app selection. Note that app ratings and reviews can be influenced by privacy concerns around permissions, and thus this 5% metric should actually be treated as a lower bound in terms of its ability to capture the relevance of permissions for app installation.

Our observation about the influence of permissions at installation time corroborates the finding in [Kelley et al., 2013], where permissions ranked 8th out of 11 reasons. However, our findings about the influence of reviews and ratings differ significantly from those in [Kelley et al., 2013] (see Figure 2 therein). They found that ratings, reviews and cost were most important (in that order) and of similar importance, whereas in our study developer and popularity were factors cited more frequently than ratings and reviews. This could be due to different study methods. They asked 366 MTurkers to rate factors on a 5-point importance scale, whereas we asked participants to select all that apply. Moreover, the MTurkers in [Kelley et al., 2013] were asked about their general views, whereas our participants were asked about specific apps right after installation. This suggests that an interesting avenue for future research would be to understand if and why the influence of reviews and ratings are evolving.

Table 7.3 shows the reasons why users install particular apps. For each reason, the percentages indicate the proportion of install events (total events counts in Table 7.2) it was selected for. The reason "I want to try it out", that may capture curiosity, dominates the list and is selected in 50% of installations as a reason. The other popular reasons "The app is useful" and "The app is cool or fun to use" stress that the app's functionality plays an important role as well. We found that only 14% of the installs had social influences such as family and friends. Only 34 times (2% of the surveyed installations) did participants indicate that they compared the number of permission requests across apps before installing. However, these 34 instances originated from 15% of our participants. We hypothesize that permissions may not be a key reason at moments of installation because Android users are aware that in the runtime permissions model they can make decisions about permissions later when using the app. In Section 7.6.1, we see this partly confirmed since for 40% of instances when denials occurred, participants said they did so because they can grant these permissions later.

Table 7.3: Reasons participants checked for app installation (multiple responses per installation event are possible)

App Install Reason	Number of Occurrences (% of install events)
I want to try it out	954 (49.9%)
The app is useful	579 (30.3%)
The app is part of a product/service that I use	500 (26.1%)
The app is cool or fun to use	400 (20.9%)
I trust the app or the company making the app	310 (16.2%)
My friends/family use it	276 (14.4%)
It was the only app of its kind (no other apps provide the same functionality)	160 (8.4%)
Other	129 (6.7%)
I was required to install it	126 (6.6%)
I was offered something in return (e.g. credits, monetary rewards, discount)	79 (4.1%)
The app has fewer permissions than other apps like it	34 (1.8%)
I don't know	34 (1.8%)

7.5.3 App Removals

The reasons our participants remove apps are shown in Table 7.4. As expected, the most common reason is that the participant no longer uses the app. The second most common reason, device performance, influenced 28% of app removals. In Section 7.5.1 we saw that participants are uninstalling apps at an average rate of 2/week. We were surprised by this as we assumed that when users stop using an app, they simply leave it ignored on their device rather than actively bothering to remove it. We see from these rationales that users are often removing apps for performance reasons and this contributes to the removal rate. We note that the "Other" bucket is large. Upon examination of the open ended feedback for the 128 app removal events in the "Other" option, we found that it mostly included additional details

[3]Some older apps that do not target an Android API level of 23 (Marshmallow) or above, and that are not yet updated to use the new permissions model, could still show a list of requested permissions at install time.

Table 7.4: Reasons participants checked for app removal (multiple responses per removal event are possible)

App Removal Reason	Number of Occurrences (% of removal events)
I no longer use the app	307 (39.6%)
To free up space or speed up my device	216 (27.9%)
I didn't like the app	208 (26.9%)
Other	128 (16.5%)
The app is not working as expected	120 (15.5%)
The app is crashing / very slow	48 (6.2%)
Because of advertisements in the app	42 (5.4%)
Because of in-app purchases	35 (4.5%)
The app required permissions I wasn't comfortable granting	32 (4.1%)
I don't know	16 (2.1%)

clarifying one of the already selected options. Some of the remaining responses suggested issues related to privacy or mismatched expectations. Examples include:

- Permission abuse: "The application is abusing the permission for location that I granted it. Uninstalling for this abuse of GPS." (P7)
- Negative publicity: "Read that the app is stealing private information about the phone and sending it back to China." (P31)
- Expectation mismatch: "It didn't have the information I was expecting it to have according to the description box."(P64)

Not all negative press cycles result in uninstalling apps, but for the participant above (second quote) it did. The reason "App required permissions I wasn't comfortable granting" is among the least influential here, however that option was triggered by 15% of our participants for 32 removal events. Note that if this 15% is extrapolated to the Android user base, that includes over 2 billion active devices, then the order of magnitude for devices uninstalling apps due to permissions would be in the 10s of millions.
In April 2016, the Google Play store started to require all developers to prominently disclose if their app included ads and in-app purchases. Among our participants, we see that only 10% of all uninstall events were influenced by ads or in-app purchases. This low fraction may be due to this extra transparency that helps manage people's expectations.

7.6 Permission Decisions

In this section, we discuss the reasons participants provided when accepting or denying app permission requests. Our participants granted 86% of the surveyed permission requests, indicating that they were 6 times more likely to grant a permission request rather than deny it, on average. It is noteworthy that the 14% of permission requests that were denied came from 49% of our participants. This indicates that nearly half of our participants denied a permission at least once in a 6 week period. We also observed that 95% of all decisions were made via the runtime dialogs as opposed to from inside the Android settings menu. The permission grant ratio for decisions made at runtime is 86%, whereas it is only 71% for decisions made via the settings menu, implying that users are more likely to deny a permission through the settings than when deciding at runtime. One plausible explanation is that users, especially those concerned with privacy, may seek to turn off access to personal data when they are not using an app.

7.6.1 Permission denials

Table 7.5 shows the reasons participants had for denying permissions. Participants could pick as many reasons as they wanted for each decision, and overall the average number of reasons per denial decision was 2.3. The top two reasons imply that the majority of decisions are being made by focusing on the functionality of the app, and whether or not it really needs the particular permission. This corroborates previous findings by Wijesekera et al. [Wijesekera et al., 2015], who observed that relevance to app functionality is a dominant reason for blocking permissions, though we find different fractions of participants who select this reason. Wijesekera et al. found that 53% of their participants wanted to block a permission because it seemed unnecessary for the app functionality. If we use our top two reasons as a proxy for their "unnecessary for app functionality" reason, our data reveals that 34% of our participants fall into this category. A potential explanation for why our study observes fewer participants denying permissions because they felt it was unnecessary is as follows. In [Wijesekera et al., 2015] the participants were shown (at the end of the study) a handful of permission accesses that had occurred during the prior week and asked if they would have liked to deny them and why. This captures their attitude. In our study, we capture participants actions (i.e., behaviors) and their associated rationale. In essence this gap reflects a type of difference between privacy attitudes and behaviors and thus it is not surprising that the privacy behavior occurs less often than the stated attitude.

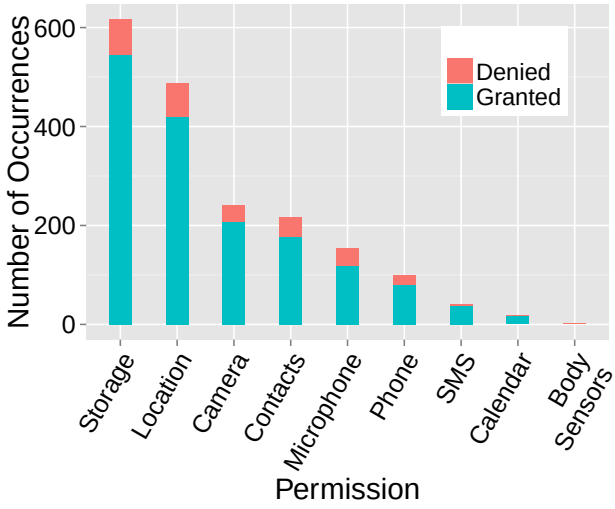
It is interesting to note that the reason "I can always grant it afterwards if I change my mind" is very prevalent among our participants (essentially tied for second place), indicating that users are aware about the fact that permissions for an app can be changed at any time (via Android's settings menu). Providing this answer for a permission denial could indicate that the user is denying the permission initially to see if the app still works, and undoing this

Table 7.5: Reasons participants checked for denying a permission to an app (multiple responses per deny event are possible)

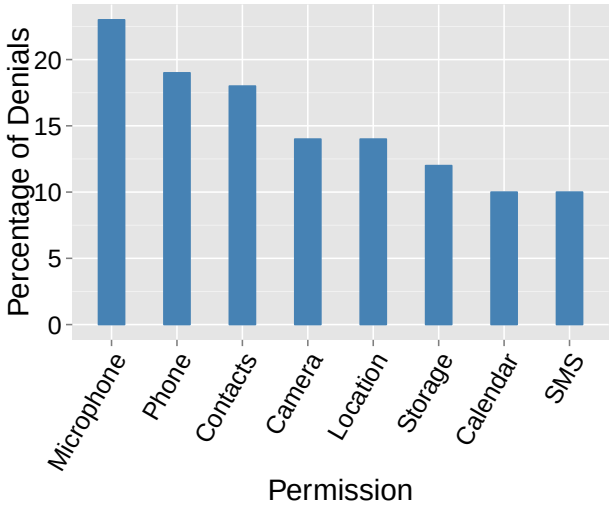
Permission Deny Reason	Number of Occurrences (% of deny events)
I think the app shouldn't need this permission	111 (40.8%)
I expect the app will still work without this permission	110 (40.4%)
I can always grant it afterwards if I change my mind	110 (40.4%)
I do not use the specific feature associated with the permission	95 (34.9%)
I consider the permission to be very sensitive	57 (21%)
I don't trust the developer enough to provide this information	42 (15.4%)
I wanted the permission screen to go away	36 (13.2%)
Other	28 (10.3%)
I think something bad might happen if I provide this permission	15 (5.5%)
I didn't know I did that	7 (2.6%)
I don't know	6 (2.2%)

decision later if necessary. This may indicate that the participant would prefer to use the app in a more private way and tests that possibility.

There were 57 instances where our participants denied a permission because they explicitly considered it to be very sensitive. It is striking to see that this was a more significant reason than not trusting the developer. Among these 57 instances, only 22 also picked "don't trust the developer" option. This implies that the remaining 35 instances (coming from 18 participants) correspond to scenarios were the participants do not distrust the developer but nevertheless consider the permissions sensitive and do not want to share the data. This suggests that although trust is necessary, it may not be sufficient to convince users to share data. This is of course a complex issue that requires further study because it is hard to know exactly how participants interpreted the "trust" option in our surveys.



(a) Number of permission changes per permission.



(b) Percentage of permission decisions (either through dialogs or through the settings) that led to a permission being denied.

Figure 7.5: Participant Permission Decisions

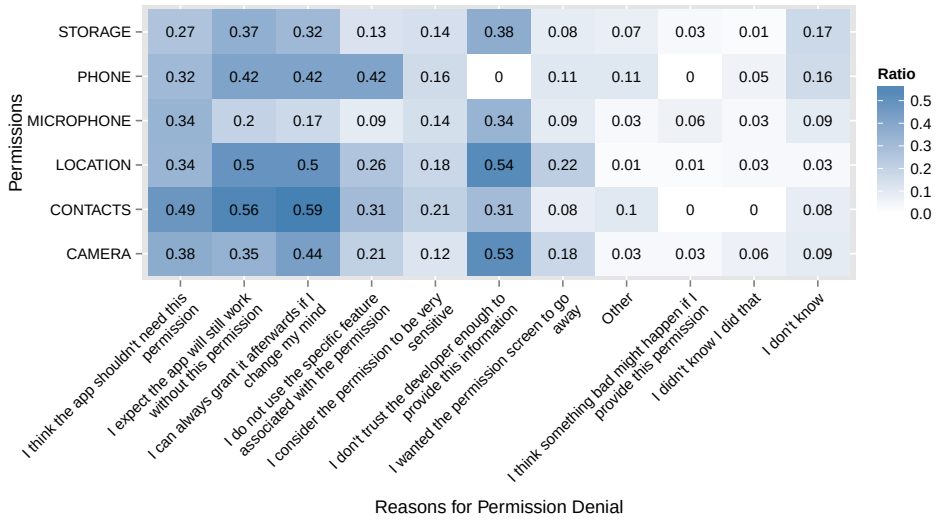


Figure 7.6: Reasons participants checked for denying each specific permission (multiple responses per deny event are possible). Each entry in the heatmap expresses the ratio of number of times that reason was given for the permission, over the count of all denies for the permission.

We now examine decision making with respect to permission types. In Figure 7.5(a), we see that the largest number of permission decisions occur for Storage and Location permissions. For each permission type, Figure 7.5(b) shows the fraction of requests that were denied. As is clear from this plot, the Microphone permission has the highest percentage of denials, followed by the Phone and Contacts permissions. It is interesting that Camera access did not exhibit a similar denial rate as Microphone; we posit that this might occur because the Camera permission sometimes only entails taking still photos (without audio and video). Although Location is perhaps the permission that users are most aware of, it does not appear among the top three most denied permissions. One possible reason is that users might have experienced some sort of habituation effect [Bouton, 2007] for the Location permission, where a repeated exposure to such a permission request could have reduced their level of sensitivity or concern when granting such a permission, similarly to what has been reported in another study on pop-up dialogs [Bravo-Lillo et al., 2014].

To determine whether some decision rationales are more influential for specific permission types, we broke down our participants' reasons for permission denials according to the permission type. Figure 7.6 illustrates this via a heatmap. We have removed 2 permission types, SMS and Calendar, because there were fewer than 15 denials for these permissions.

Overall, we observe that the top two or three reasons for each permission type can differ. For example, for Location and Camera the top reason for denying is "I don't trust the developer".

This reason has little significance for Phone and Contacts, where the dominant reasons are "I can always grant it afterwards" and "The app will still work without this permission". This shows that users make decisions about each of the permission types according to different rationales. We hypothesize that for Phone and Contacts, our participants might be trying to not share them initially at all (and only doing so later if really needed) - thus issues of functionality are top of mind. However for Location and Camera, it is possible that the reason why the data is needed is often more clear and thus the primary rationale is based on trust.

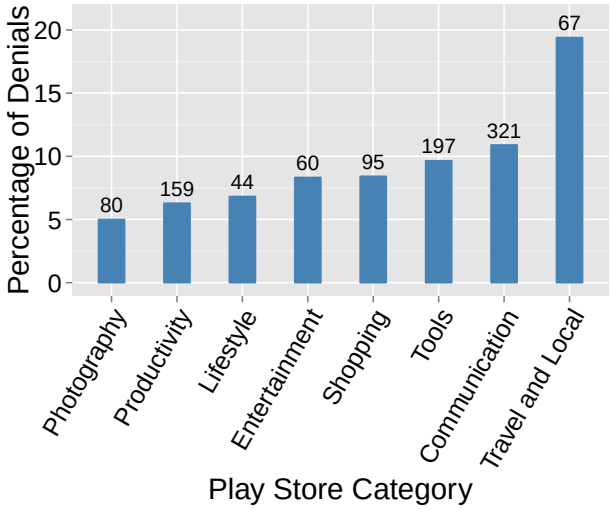


Figure 7.7: Percentage of permission denials across apps belonging to different Play store categories. The numbers on the bars indicate the total number of permission decisions in each category.

Next, we assess whether the permission denial rates are different across different app categories. For each of the 624 apps that registered a permission grant or denial event in our study, we identified its Play store category and considered it as an indicator of the app's functionality type. We recognize that some Play store categories, such as 'Productivity', are very broad and cover a wide range of app functionalities. However, app category was the only readily available functionality indicator.

Among the 624 apps, 41 did not appear in the Play store and seem to be device manufacturer apps that come pre-installed on the Android device or apps that have been downloaded from other Android app stores. For the remaining 583 Play apps, we aggregated the grants and denials across apps in each Play category. There were just 8 categories that had more than 20 apps, and the denial rates for these categories are shown in Figure 7.7. We also overlay

Table 7.6: Reasons participants checked for granting a permission to an app (multiple responses per grant event are possible)

Permission Grant Reason	Number of Occurrences (% of grant events)
I want to use a specific feature that requires this permission	1095 (68.2%)
I trust the app developer	515 (32.1%)
I think the app won't work otherwise	382 (23.8%)
I have nothing to hide	289 (18%)
Nothing bad will happen	225 (14%)
The app developer already has this information about me	208 (13%)
I wanted the permission screen to go away	164 (10.2%)
Because the app is popular	150 (9.3%)
Other	39 (2.4%)
I didn't know I did that	36 (2.2%)
I won't be able to grant this permission later	22 (1.4%)

the number of permission decisions within each category as the number on top of each bar. Denial rates vary between 5% - 19% across these 8 app categories. Moreover, the same permission can have different denial rates across different app categories. For example, 'Travel and Local' had a 43% denial rate for the Location permission, whereas 'Communication' registered only a 11% denial rate for the same permission. This reaffirms the influence of app functionality on users' permission grant or deny decisions.

7.6.2 Permission Grants

We now examine the reasons why users agree to grant permission requests. Table 7.6 shows that the dominant reason is "I want to use a specific feature that requires this permission", which suggests that users are agreeing because the request is in line with their expectations. As suggested by Felt et al. [Felt et al., 2012a], a goal of using runtime dialogs is to improve the permission decision making and to avoid undermining users' expectations; our results thus indicate progress on that front. The second most important reason is trust in the developer. As discussed earlier, follow up work is necessary to fully understand how trust influences

permission choices. Nonetheless, this result underscores how important it is for developers to gain a trustworthy reputation among (potential) users.

In a similar way as we did for the denials case, we checked whether some reasons are more influential for specific permission types, but found the distribution of reasons to be similar across permission types.

Next we look at the question of whether or not participants grant permissions willingly. Recall that after our participants granted a permission, we asked them to indicate if they agree or disagree (5 pt scale) with the statement "I don't mind giving <app> access to my <permission>" (Q2 in Appendix C.1.3). Surprisingly, we found that 10% of the time, participants indicated that they "Disagree" or "Strongly disagree" with the statement (see Figure 7.8). This could occur if participants believe an app won't work without the requested permission and so they agree, albeit reluctantly. This can be associated with the phenomenon of "learned helplessness" [Warshaw et al., 2015], which covers scenarios when participants convince themselves they agree with something (e.g., data sharing) because they did not really have a choice.

To see whether this comfort level changes over time, we asked participants in the exit survey to rate their comfort level with permissions they had granted to apps on their phones in the past (Q19 – Q22 in Section C.2; we included "I don't know the app" as an additional option). When asking these questions, we made the permissions more specific. For example, if the participant had granted the Storage permission, we ask whether they were comfortable with the app accessing photos on their device storage. These questions were intended not only to revisit comfort with prior decisions, but also to illustrate more explicitly to the participants the implication of their decision. These prior decisions may have occurred any time during our 6 week study or even earlier as explained in Section 7.3.1.2.

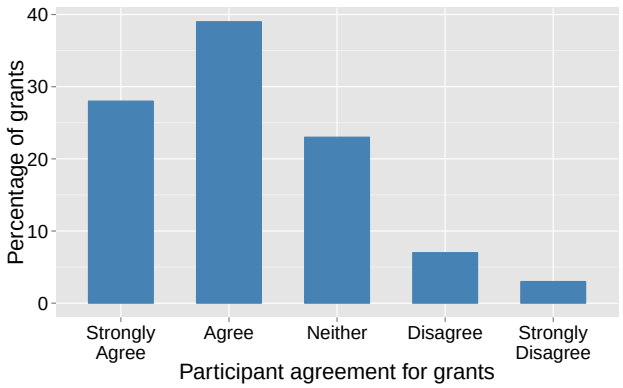


Figure 7.8: Participant responses to the statement: "I don't mind giving <app> access to my <permission>", right after granting that permission.

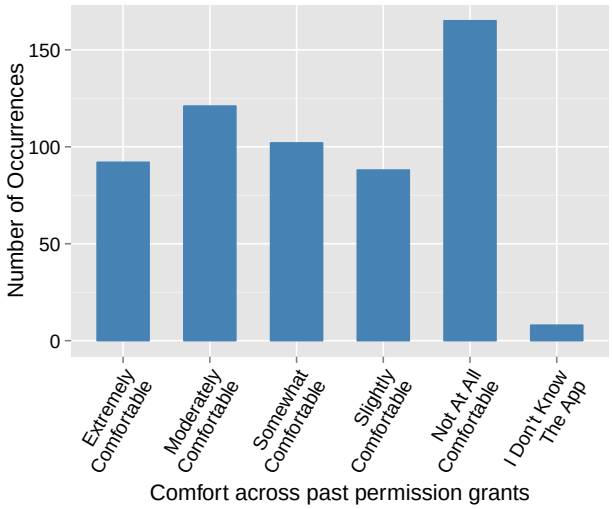


Figure 7.9: Participant comfort for permissions that were granted in the past, in response to the exit survey question "How comfortable would you be with the <app name> app knowing <information available through the permission>".

In a surprisingly high number of situations (see Figure 7.9) participants were not comfortable with their prior decisions. In 29% of scenarios presented to the participants, they indicated they were "Not at all comfortable" with the data access that was allowed to the app. If we include the cases where users were "Slightly comfortable", then we see that in 44% of the cases our participants are not feeling comfortable about their past decisions. These discomfort levels vary based on the permission: on a scale from 1 to 5, where larger numbers indicate a higher discomfort, the Storage permission entails an average discomfort of 3.41, Phone has a discomfort of 3.33, Contacts has a discomfort 3.11, and Location has a discomfort of 2.77.

Participants were not comfortable about permissions they granted in the past and this may be occurring because they do not always understand what a permission entails, and only realize this after it is made explicit. Consider, for example, the Storage permission: this permission might be understood by a user as allowing the app to store data on the device, only to be refuted by our question stating that the app now has access to pictures on the user's device. This explanation is supported by previous work [Harbach et al., 2014; Swanson et al., 2010] that has shown how users need to be confronted with a specific scenario before being able to correctly reason about privacy and security.

It is interesting to contrast the 29% discomfort long after decision making, to the 10% reluctance that existed at the moment of decision making. This 29% statistic could be said to capture privacy attitudes; the exit survey captures what people say or think about sharing data

when they are being questioned but not making a real life decision. However in practice, in only 10% of grant decisions did users say that they minded sharing the data right after granting. The gap between these numbers approximately captures the difference in participant's attitudes and behaviors, in the context of Android permissions.

7.6.3 Other influences

We check whether the participants' demographics are associated with their grant/denial behavior. We used Pearson's Chi-squared test (with Yates' continuity correction when needed) to check the dependence between participants' age and gender, and their denial behavior. We control for age (gender) when gender (age) is being tested. Due to small sample sizes, we did not test for independence across education and employment demographics. We notice that women across age groups 18-23 ($\tilde{\chi}^2$ = 10.7, df = 1, p-value = 0.001068) and 31-40 ($\tilde{\chi}^2$ = 16.3, df = 1, p-value = 5.396e-05) are three times as likely to deny permissions than men. On average over all age groups, women deny twice as often as men, with a 20% denial rate for women compared to 11% for men ($\tilde{\chi}^2$ = 25.6, df = 1, p-value = 4.11e-07). Comparing men across different age groups, we notice that men's denial rates differ significantly ($\tilde{\chi}^2$ = 31.2, df = 4, p-value = 2.841e-06); participants in age ranges 18-23 and 31-40 have denial rates around 5% whereas the other age groups have denial rates of 15% or higher, about three times higher.

Lastly, we checked associations between participant responses to questions in the exit survey (Q1–Q18 in Section C.2) and their permission denials. We did not find any statistically significant correlations or dependencies.

7.7 Conclusion

There are a couple of important takeaways herein for Play store developers. First, we saw that in terms of app installs and uninstalls, permissions were not a dominant reason compared to other reasons. However, 15% of our participants uninstalled apps due to permissions. Extrapolating this statistic to the set of Android devices (over 2 billion), indicates that this could affect tens of millions of devices. This result could motivate developers to reconsider requesting certain permissions at all or to make runtime requests more contextual – for example by only asking for permission access when the user opts to use certain functionality within their app rather than at first run.

Second, the vast majority of rationales for decision making around permissions are related to app functionality, whether the app needs the permission, whether it "should" need it, and whether the user needs the functionality entailed by it. Thus, participants are more willing to grant permissions when the reason for asking is clear. This should motivate developers to provide sufficient and clear explanations for their requests. Android provides a utility

method (`shouldShowRequestPermissionRationale()`) to help identify situations where users might need an explanation.

In summary, we observed an overall denial rate of 16%. These denies came from half our participants which indicates that there exists one or some scenarios for many people in which they will deny a permission. The scenarios when participants deny permissions are very varied. This is implied by the findings that i) denial rates vary from 10% to 23% according to permission type, and ii) denial rates vary from 5% to 19% across app genres (Play store categories). Among our participants, we also saw that women denied permissions roughly twice as often as men.

We found that even though the overall grant rate is quite high, at 84%, there is a portion of decisions (10%) in which users grant permissions reluctantly. Moreover, users were surprisingly uncomfortable (29%) when revisiting their prior decisions at the end of our study. This indicates a gap between behaviors and stated attitudes.

Our participants' rationale for denying a permission in 42% of denial instances, was because they knew they could change the permissions afterwards. We hypothesize that this might be happening because participants want to test out whether or not the app will work in a more privacy preserving way (with less user data). Exploring this would be an interesting avenue for future research.

It is interesting albeit hard to understand how users' comfort levels and understanding of permissions have evolved after the introduction of runtime dialogs. In [Wijesekera et al., 2015] (pre-runtime), the authors state that 80% of their participants wanted to deny at least one permission. In our study, we recorded that 49% of our participants denied permissions at least once. We found that 16% of permission requests were denied. This is about half the rate reported in [Wijesekera et al., 2015], though the latter study asked participants to allow or deny access many times a permission was used, instead of only on first use as in our study. These two studies differ in their interactions with users, and both involve limited populations, yet these metrics hint that users may be getting more comfortable granting permissions using runtime dialogs. It would be interesting to explore this hypothesis in future research that makes a more direct comparison.

Catering to the discomfort of 29% of our participants about permissions they granted in the past, either because they did not fully understand the implications of granting the permission, or because they did not recollect granting the permission, we will present an initial proposal for a "Privacy API" in Chapter 9. This API aims to inform smartphone users, and other users of internet services, about the data that is being collected by the providers of these services.

Part II: Conclusion

In the second part of this dissertation, we were able to gauge the prevalence of the security and privacy issues that we presented in the first part. We showed that in 2017, usage of Wi-Fi networks is very popular, with users on average connecting to 8 different Wi-Fi networks in a 30-day period. We demonstrated that between 64% and 88% of mobile device users can be mapped to at least one previously visited physical location, only based on the signals broadcast by their smartphones.

Over one third of the Wi-Fi networks that smartphone users connect to do not have any encryption enabled, which means that connections can be monitored by eavesdroppers in range of the network. Even more worrying is that 45% of these users are susceptible to Evil Twin attacks, allowing an attacker to set up a fake access point that their devices will connect to automatically (offering the attacker the position of an active man-in-the-middle), without user intervention. Combining these results with the fact that 13% of connections made by apps on users devices are insecure (and the user being unaware about 38% of them), we can state that privacy and security issues in smartphones are something that affects a large amount of smartphone users.

We then continued to show how aware smartphone users are about the aforementioned issues. Gauging how aware users were about the ability of attackers to extract information from probe requests sent out by their devices, we showed that 76% of the surveyed users were unaware about one or more privacy or security issues induced by this fact, with 52% of users being unaware that this causes an attacker to mount an Evil Twin attack. Furthermore, users indicated to be worried about 91% of the insecure connections that were made by their devices, confirming the Privacy Paradox. We also showed that, unfortunately, an increased awareness does not always translate to better security practices.

Analyzing users' decision making regarding permissions for apps on their devices, we show that smartphone users are relatively comfortable with the information access they are granting to apps: in our study, participants indicated they did not mind granting 84% of permission requests at the moment the access was allowed. However, when questioned about their comfort around already granted permissions at the end of the study, and with explicit examples of the exact data that is available to the app developer, the level of discomfort climbed to 29% of

cases. To cater to this discomfort, caused either by the fact that they did not fully understand the implications of granting the permission, or by the fact they didn't recollect granting the permission, we will present an initial proposal for a "Privacy API" in Chapter 9. This API aims to inform smartphone users, and other users of internet services, about the data that is being collected by the providers of these services.

In performing our studies on Wi-Fi awareness, we also increased awareness among our participants and other smartphone users, demonstrating the value of using very specific, personalized scenarios. This approach proved to be highly effective: 76% of participants in the SASQUATCH study indicated they weren't aware about the presented issues before the experiment. From their responses, we found that participants indicated they were willing to try out different solutions to mitigate these issues. We will discuss possible solutions in the next part of this dissertation.

Part III

Towards improving security and privacy for mobile devices and users

Introduction

As we already discussed in Chapter 5, existing work shows that people are willing to act in the interest of preventing further privacy leaks [Boyles et al., 2012; Günther and Spiekermann, 2005; Consolvo et al., 2010]. This is confirmed by our own work: 62% of participants in the SASQUATCH study indicated they were willing to make an effort to make their smartphones more secure against the demonstrated attacks. In the last part of this dissertation, we provide an overview of methods and techniques that can help mobile device users in improving their security and privacy, while keeping in mind the difficult trade-off between usability and security [Whitten and Tygar, 1999].

When the participants in our SASQUATCH study were asked about the actions they are willing to undertake to protect against these issues, an overwhelming majority of users say they would like to be able to install an app that prevents different Wi-Fi attacks. Indeed, this set-it-and-forget-it approach also surfaced in the study about Wi-Fi privacy assumptions, where users preferred delegating security to a tool they have to set up only once. For this reason, we will also describe the implementation of an Android app that automatically protects against different Wi-Fi attacks, called Wi-Fi PrivacyPolice.

We also provide an overview of other ways for improving Wi-Fi privacy on smartphones, from a variety of contexts. These solutions are presented as suggestions to changes in technology, habits and legislature to different stakeholders, including developers, ISPs and lawmakers.

We go further than discussing only solutions to the technical aspects of smartphone security and privacy by also considering the privacy of information flows of apps that are installed on users' devices. For this, we introduce the concept of a privacy API that allows us to cover the privacy of user data after it has been captured.

We end the final part of this dissertation with a discussion of the parallels between issues that were handled in this work and those that exist in Internet of Things (IoT) devices today. In the last chapter, we will make the case that the same issues exist for both types of technology, and that the Internet of Things space is now in the same place as mobile devices were at the start of this PhD work. We briefly discuss the impact this has.

Chapter 8

Technical solutions to smartphone privacy and security issues

8.1 Introduction **139**
8.2 Recommendations on smartphone security **139**
8.2.1 What the smartphone user can do **139**
8.2.2 What a developer / manufacturer can do **141**
8.2.3 What network providers and ISPs can do **143**
8.2.4 What security and privacy researchers can do **143**
8.3 Automatically solving privacy issues with Wi-Fi PrivacyPolice **144**
8.3.1 Preventing network leakage **145**
8.3.2 Preventing evil twin attacks **145**
8.3.3 From proof-of-concept to consumer app **146**
8.4 Comparison to other mitigation strategies **147**
8.5 Conclusion **148**

8.1 Introduction

After our studies in the previous chapters, we provide some solutions to the aforementioned privacy and security issues on mobile devices. In this chapter, we deal with technological improvements, both in the form of recommendations to a variety of stakeholders, and in the form of an app that allows the set-it-and-forget-it approach that was discussed in Chapter 6.
In the first section, we describe a set of guidelines on how to prevent future privacy leaks. These are presented first to the smartphone users themselves, to show how they can protect themselves in the short term. However, we also present a set of recommendations to developers, manufacturers, network providers, and researchers. We do this with the aim of solving these issues in the long term without requiring user intervention, benefiting all smartphone users instead of only those concerned about their privacy.
Afterwards (in Section 8.3) we describe Wi-Fi PrivacyPolice, an app designed to solve a subset of the described privacy and security issues in an automated fashion. Specifically, it prevents the names of preferred networks from leaking to third parties, and it protects against Evil Twin attacks, which are designed to lure smartphones into connecting to a clandestine network.

8.2 Recommendations on smartphone security

In this section, we describe some countermeasures that can be taken by different stakeholders, based on the results from our studies in earlier chapters. First, we describe strategies that allow smartphone users to mitigate network attacks mostly in the short term. We then show how these attacks can be solved at a technical level, by giving some recommendations to developers; this category of stakeholders includes app developers as well as smartphone manufacturers and operating system vendors. Some of these techniques will be applied in a real-world app in Section 8.3. We then continue with recommendations to ISPs and network providers on the opportunities to secure their connections. We close with ideas on how academic research could help improve smartphone security.
As described in section 5.2, usability is an important factor to consider when increasing the security of a system [Whitten and Tygar, 1999]. For this reason, we aim to provide only solutions that have a minimal impact on user convenience and usability.

8.2.1 What the smartphone user can do

As a stopgap solution, users can largely secure themselves against profiling and Evil Twin attacks by removing networks from their preferred network list (PNL) when they are not needed. While removing networks from the PNL is easy on both Android and Windows Phone mobile devices, iOS devices lack the capability of removing networks when they are not in range. Thus, iOS users wishing to remove networks from their PNL need to either be in

range of the original network (which is often not possible), or need to actively spoof the network themselves when they want to remove it. Because the SASQUATCH setup described in Chapter 5 impersonates all networks requested by smartphones in range, participants in our experiment and other iOS users within Wi-Fi range of our setup were also able to remove these networks from their PNL. This solution, together with instructions on how to remove networks from a smartphone's PNL, was also displayed to participants that finished our survey. We also included the remark that iOS users had the option of staying close to our setup in order to remove insecure networks from their list.

Similarly, iOS users are able to protect against the Evil Twin attack by enabling the option "Ask to Join Networks" in the Wi-Fi settings. Enabling this option will cause the iPhone or iPad to never automatically connect to a known network. Instead, it will ask the user for confirmation every time a network connection is made, making this a typical trade-off between convenience and security. In Section 8.3, we will describe a solution (implemented as part of an app), that allows a third – intermediate – option: having the smartphone ask confirmation whenever a new access point corresponding to an already trusted network is encountered.

Smartphone users can also secure their own managed networks (e.g. their home network) against the previously mentioned attacks by choosing a common SSID (e.g. `linksys` or `dlink`, which are among the highest ranking SSIDs in our dataset based on occurrence), and securing it with a non-common key. This effectively thwarts profiling by SSID because an eavesdropper has no way of knowing where the particular network the smartphone is referencing is located. Furthermore, Evil Twin attacks are prevented because an attacker has no way of knowing which key should be used when spoofing the network. Since this method requires the smartphone to have a common SSID in its network list, a disadvantage could be that the smartphone continually discovers networks it thinks are the home network of the user. This would lead the smartphone to try and fail to connect to each one of these networks, possibly draining the battery.

Virtual private network (VPN) services allow users to route all their network traffic through an intermediate server in an encrypted fashion, and could be used to ensure a secure connection even when the network itself is compromised by an attacker. While these services are indeed able to prevent an eavesdropper (or the network owner) from intercepting traffic, they present some other challenges. First, VPN services can be difficult to use for technically less educated users or users depending on services that actively block the use of a VPN. Second, and more importantly, VPN services are difficult to vet, with a recent study showing that 75% of Android VPN apps contain a third-party tracking library, and over 38% of them contain some type of malware [Ikram et al., 2016].

8.2.2 What a developer / manufacturer can do

A technique for preventing tracking of Wi-Fi devices over time that has gained some traction in the past years is MAC address randomization. This technique essentially uses a pseudo-MAC address when the device is in an unauthenticated state (such as when scanning for new networks), preventing devices such as the one described in Chapter 2. This technique has been implemented in recent versions of both Android (since version 5.0) and iOS (since version 8). However, multiple studies have shown that either this countermeasure was completely absent, not implemented correctly, or easy to subvert [Martin et al., 2017; Robyns et al., 2017].

As mentioned in Section 2.8, preventing third parties from knowing the MAC address of the device is not sufficient. Indeed, a third party attacker can still infer information about the smartphone from the list of SSIDs broadcast by this device. Indeed, as a short prestudy for the SASQUATCH study in Chapter 5 has shown, we were able to identify a significant number of our own colleagues (researchers with a background in computer science) solely based on the network SSIDs sent out, the manufacturer of the smartphone or the time at which they entered or left our lab building. Thus, it is important that this SSID leakage is prevented *in addition* to techniques like MAC address randomization.

A simple solution for preventing leakage of all SSIDs while providing the same existing convenience and ease-of-use to the user would be to not have the smartphone send out probe requests for known networks. However, this method is unlikely to be adopted by any major smartphone manufacturer, as it would require smartphones to continuously scan the Wi-Fi spectrum for beacons sent out by access points (known as 'passive scanning'), necessitating the Wi-Fi radio to be enabled at all times.

A better option would be to use only *broadcast* probe requests. This type of probe request does not contain an SSID, and invites all access points in range to respond with a probe response containing the network identifier of the access point's network, enabling the smartphone to pick out its preferred networks. However, this approach would not work for networks with hidden SSIDs, as they require the client (in this case, the smartphone) to actively show its knowledge of the network beforehand. This should not be considered a major problem, as these types of networks are considered not to add any security, and the use of them has been discouraged for some time [Davies, 2005]. Lindqvist et al. [Lindqvist et al., 2009] describe a privacy-preserving method for access point discovery allowing for the use of hidden access points using cryptography.

Limiting the broadcasting of probe requests would also mitigate the problem of connections that are being made to impersonated (Evil Twin) access points. Indeed, if no specific SSIDs are mentioned in probe requests, an adversary has no way of knowing which networks to spoof. However, an attacker will still be able to spoof generally available networks (e.g. McDonald's `Wayport_Access` network). To further protect against Evil Twin attacks, a smartphone could allow connections to a network only at locations where it is known to be in range (e.g. beacause the first connection was made at that location). Using the exact location, however, may cause the battery to drain at a faster rate because the smartphone's GPS is used.

As an alternative to using the exact location, connections to a network can be restricted to envrionments where a known set of other networks is also in range.

All of the previous solutions rely on smartphone manufacturers and mobile OS developers for a universal implementation. However, operating systems like Android, Windows or GNU/Linux allow for applications to read and/or modify the Wi-Fi configuration, and to enable or disable networks in the PNL. This would allow for applications that either warn the user about forgotten access points (i.e. access points that have long not been used), that enable wireless networks on a per-location basis or that even only enable networks as soon as they are known to be in range (e.g. based on probe responses to broadcast probe requests). Disabling networks when not needed effectively stops the smartphone from sending out probe requests for these networks, mitigating both of the problems mentioned. Section 8.3 will discuss an automated solution that allows only connections to SSIDs when the access points themselves are known (based on the access points' MAC addresses), providing the benefits of a location-based approach without the battery impact.

Having security between the mobile device and the network, preventing eavesdroppers from monitoring metadata about network connections, is only part of a complete solution. With most participants from our Wi-Fi survey (71%) having a high concern of their data being available even to the network provider, app developers need to secure against all possible forms of man-in-the-middle attacks. In practice, this means making sure that all connections between the user's device and the app provider's servers are end-to-end encrypted, and that the libraries and implementations used to achieve this are secure and up-to-date (see [Georgiev et al., 2012]). We recommend using some form of certificate pinning, where the public key certificate of the other endpoint is already embedded in the app package instead of relying on the system's certificate chain to validate the other endpoint's key. As 12.92% of the connections logged during the study from Chapter 6 (excluding servers of advertisement agencies) were insecure, app developers in general have some room of improvement.

Based on our study on Android permissions in Chapter 7, there are some steps app developers can take in order to improve understanding for their users, and to increase the install base for their apps. First, requesting permissions at run time (instead of at install time) and making these runtime requests more contextual (by requesting them at the time they are needed) prevents apps from being uninstalled by users. Second, when requesting a permission, the rationale provided by the app should be clear, as users' permission decisions are heavily influenced by whether the user understands the request. Android provides a utility method (`shouldShowRequestPermissionRationale()`) to help identify situations where users might need an explanation.

A specific recommendation can be made to distributors of the Android operating system, too: while the fact that Android's `/proc/net/tcp` and `/proc/net/tcp6` files are readable for every installed application greatly helped performing the experiments in our Wi-Fi survey, they also pose a potential privacy risk to Android users. Indeed, the app developed for our study was able to infer the connections made by all other apps on users' devices without

requiring any special permissions, leading to users giving feedback akin to *"it's creepy what your app is able to see"*. Barring any technical limitations the authors are not aware of, we recommend restricting access to these files from apps, making them available only to the operating system itself.

8.2.3 What network providers and ISPs can do

As our results in Section 6.5 show, over one third of the Wi-Fi networks that smartphone users connect to do not have any security enabled. This shows that even in 2017 more attention needs to go to pushing network operators to secure their Wi-Fi networks. As mentioned in Section 6.6, the biggest offenders are commercial entities and ISP hotspots. While convincing all commercial entities to upgrade their networks to provide security might prove difficult, convincing a few ISPs to do the same will yield an almost equally good result. Moreover, as these ISP hotspots are widely distributed (with each network name corresponding to a large number of individual access points), devices will often automatically connect to one of those networks, posing a large security risk. Indeed, they accounted for over 94% of all connection pairs in our study. Thus, we recommend ISPs to eliminate any insecure hotspots from their network, instead providing only encrypted hotspots to their customers. If this proves to be difficult, e.g. because the ISP also wants to provide internet access to non-subscribers through a captive portal, we recommend offering two separate wireless networks: a secured network for subscribers, and an insecure network that allows the use of a captive portal. In addition, we envision that implementing the 802.11u wireless roaming standard and Hotspot 2.0 (also known as Passpoint) could provide benefits in this regard.

One consequence of cellular data replacing (or supplementing) Wi-Fi for mobile device users is that trust shifts from the Wi-Fi network providers to the mobile network operators. This makes the operators the middle man between the user's device and the other endpoint, and gives them the same responsibilities: they need to make sure the network is upgraded to the latest generation so that users are not susceptible to eavesdropping attacks. While LTE provides significant security advantages over older generations of networks, it is not immune to attacks [Shaik et al., 2016]. Whereas at the moment of writing these do not pose an immediate threat to users in the form of eavesdropping attacks, these could still enable an active attacker to force a victim's device into using 2G or 3G rather than LTE networks, which can in turn make it possible to mount 2G- or 3G-specific attacks.

8.2.4 What security and privacy researchers can do

As was already indicated in Section 6.6, users tend to use a set-it-and-forget-it approach when dealing with security and privacy issues, limiting their security awareness when trying to complete a task at hand. This makes that even generally privacy-aware users are often unaware about security issues at the moment they occur.

To cater to the set-it-and-forget-it approach, users need to be able to delegate their security approach to a tool, setting it up at a time that is convenient to them. Such a tool can take the form of, for example, the Wi-Fi Privacy Ticker [Consolvo et al., 2010], informing the user at the exact moment privacy-sensitive data is being transmitted by their device, and allowing them to prevent it if desired. As far as the authors are aware, such a tool is currently not available for mobile devices, and developing it could prove to be an interesting area for future work.

Another approach, similar to what we discussed in Section 8.2.2, could be to have either a tool or the operating system handle possible insecure situations differently, modifying the user interface to nudge users into making good security decisions as described by Balebako et al. [Balebako et al., 2011]. For example, the operating system could be modified to make it more difficult to connect to unsecured networks, either by making this option less accessible in the user interface or by having the user explicitly dismiss a warning about the dangers of connecting to such a network. This is already the case on some operating systems, and it also corresponds to user interface changes that were made in specific applications in order to improve users' security behavior. For example, the Google Chrome browser uses SSL warnings to warn users about the dangers of unsecure connections, based on an extensive study of possible designs, and attributing improvements to these dialogs to 'opinionated design', using visual cues and specific threat scenarios to explain the risk to users [Felt et al., 2015]. In a similar vein, unsecured networks could be prevented from being included in the list of 'preferred networks' the mobile device will connect to automatically, still allowing the user to complete their task (by allowing a connection to an insecure network when needed), but preventing the device from connecting to a similar network later on. This is similar to the approach taken by the "Wi-Fi PrivacyPolice" tool, discussed in the next section, which can be set up to only allow connections to access points that were encountered before.

8.3 Automatically solving privacy issues with Wi-Fi PrivacyPolice

In order to solve the privacy issues associated with mobile Wi-Fi usage in an automatic manner, we created an Android application, first as a proof-of-concept but later published as a real application. This application, called Wi-Fi PrivacyPolice, is designed to prevent the attacks described before. We describe the functionality of the application in two parts: the prevention of network leaks, and the prevention of evil twin attacks. We tested both mitigation strategies on our personal devices for a period of 6 months, and using different types of wireless networks and access points. We noticed no degradation of the user experience.

8.3.1 Preventing network leakage

The leaking of SSIDs from the smartphone's PNL is prevented by using the application to disable all networks in the PNL by default. This prevents the 'active scanning method' described in Section 2.3 from broadcasting directed probe requests. Instead, the application sends out a *broadcast* probe request, which does not contain any specific SSIDs, but requests all access points in range to respond with a probe response. If this results in a probe response being received for a network in the smartphone's PNL, the application re-enables this network, allowing the smartphone to connect.
Broadcast probe requests are currently already used by smartphones to find networks that are not in the smartphone's PNL. They request all access points (instead of only access points corresponding to a single SSID) to reply with a probe response.
There are, however, two types of networks that do not respond to these types of requests:

Hidden networks In these networks, the access point will not signal its activity unless explicitly requested (often called *network cloaking*). However, these networks have been considered not to add any security, and the use of them has been discouraged for some time [Davies, 2005].

Networks with probe responses disabled These only signal their activity using beacon frames, which will still be picked up by the smartphone even if our strategy is used. However, it is possible that detection of these networks will take longer, since the Wi-Fi radio has to be enabled at the time the beacon frame is sent.

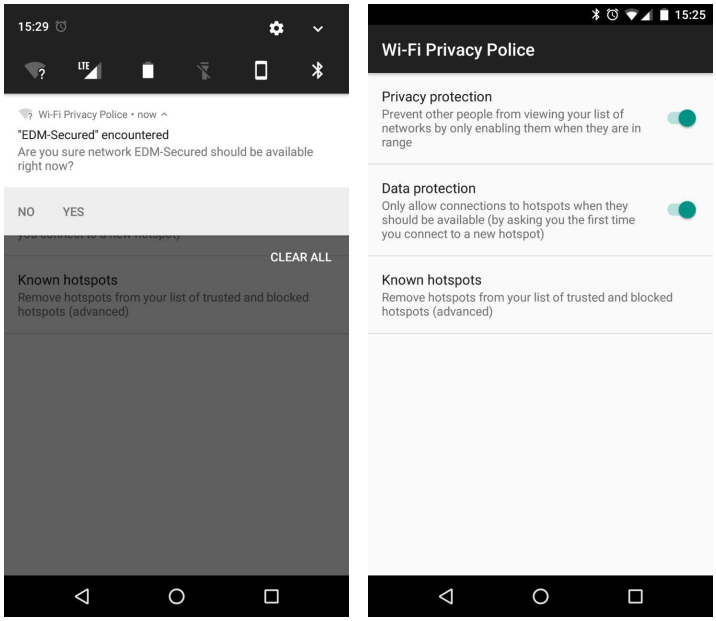
Apart from these exceptions, our experiments did not show any degradation of connection speed or quality. We modified the logic used by the app to keep hidden networks always enabled when the app was released to the public (see Section 8.3.3) as to not degrade usability for non-technical users.

8.3.2 Preventing evil twin attacks

Evil twin attacks are already partially mitigated by the strategy described in the previous section. Indeed, when this strategy is used, the attacker has no way of knowing which networks to spoof in order to have the victim connect to the Evil Twin access point.
However, an attacker could still try to spoof popular Wi-Fi networks, such as McDonald's `Wayport_Access` network in the US, or `BTWIFI` internationally.[1]. If the popular network is in a victim's PNL, its smartphone will still automatically connect to the evil twin access point. To prevent this from happening, PrivacyPolice also remembers the MAC address of every access point in the smartphone's PNL. If the access point uses a MAC address that does not correspond to the MAC address of an access point previously connected to by the user (as might be the case when roaming), he/she is asked to explicitly confirm that the network

[1] A list of popular SSIDs can be found at `https://wigle.net/gps/gps/main/ssidstats`.

is expected to be available at the current location through a notification (see Figure 8.1(a)). When the user allows the connection, all access points having the corresponding SSID that are in range are added to a whitelist, allowing the device to connect to them automatically afterwards.
This strategy does not completely prevent an attacker from mounting a successful attack: if the attacker is able to guess the exact access point the user has previously connected to, he/she can also spoof the access point's MAC address. However, this requires the attacker to have a priori knowledge about the victim.



(a) Notification asking the user about a newly encountered access point. Note that no Wi-Fi connection to the encountered network is made before the user confirmed that it should be available.

(b) The settings screen, allowing to enable either network leakage prevention or Evil Twin attack prevention separately.

Figure 8.1: Screenshots of Wi-Fi PrivacyPolice in action.

8.3.3 From proof-of-concept to consumer app

After a limited test on our own devices and those of other researchers in the field, we made Wi-Fi PrivacyPolice available to the general public on October 30th, 2014 through Google's Play Store and through the F-Droid open source app store (aimed at more privacy-conscious users that would be hesitant to install a closed source app requiring access to the list of Wi-Fi

networks). Together with the public release, we organized two Q&A sessions on Reddit[2], to inform smartphone users about the security risks in Wi-Fi use and to help them with any questions they have. The code for Wi-Fi PrivacyPolice was made available through GitHub[3]. Over the course of the past 2.5 years, Wi-Fi PrivacyPolice has been updated to fix bugs reported by users, to add requested functionality, and to work with later versions of Android.[4] Furthermore, the app now contains some simple configuration screens, allowing the user to enable either network leakage prevention (see Section 8.3.1) or Evil Twin attack prevention (see Section 8.3.2) separately (see Figure 8.1(b)). As of March 8th, 2017, the app has been installed by 95 476 users through Google's Play Store[5], who provided an average rating of 4.1/5 stars. The app has been translated into 7 different languages, and has received contributions from 8 different contributors.

8.4 Comparison to other mitigation strategies

In Section 8.2, general solutions to Wi-Fi attacks (such as using a home network with a common SSID, but a unique key to thwart evil twin attacks) were already discussed. Apart from these solutions, some other mitigation strategies have been proposed, ranging from improvements to the 802.11 discovery protocol [Lindqvist et al., 2009] and MAC address randomization[6] to location-aware Wi-Fi probing [Kim et al., 2013].
While the first category of solutions requires modifications to the protocol (or the implementation thereof), the second category is interesting because it allows users to protect themselves by installing an application (thus, without the cooperation of the smartphone vendor). There are, however, some differences between the proposed solutions and our own mitigation strategies.
First, location-based solutions require information from the GPS sensor. This makes it required for devices to have GPS enabled when scanning for wireless networks, which may be infeasible because of the battery impact, or because the device does not possess a GPS chip at all. Alternatively, the device could use the names of other networks in range in order to

[2]The Q&A sessions on Reddit can be found at `http://www.reddit.com/r/Android/comments/2uyw50/wifi_privacypolice_prevents_your_smartphone_or/` and `http://www.reddit.com/r/androidapps/comments/2u2ww0/dev_wifi_privacypolice_prevents_your_smartphone/`.

[3]Wi-Fi PrivacyPolice's source code is available at `https://github.com/BramBonne/privacypolice`.

[4]A notable example of a change to comply with newer Android versions is the following: since Android version 6.0, released in October 2015, the Android permission model has changed, requiring Wi-Fi PrivacyPolice to request 'location' permissions in order to view Wi-Fi networks in range. Because of its predominantly privacy conscious userbase, PrivacyPolice had to be modified to thorougly inform users about why the app suddenly required an extra privacy-sensitive permission, even then causing some users to uninstall the app out of privacy concerns.

[5]Install numbers for the F-Droid app store are not available.

[6]Apple reported that, starting with iOS 8, iOS devices will randomize their MAC address when scanning for networks, as is explained in `http://devstreaming.apple.com/videos/wwdc/2014/715xx4loqo5can9/715/715_user_privacy_in_ios_and_os_x.pdf?dl=1`. This randomization is enabled only when both mobile data and location services are turned off.

approximate its location. However, it is then easy to consider the case where a network is enabled only when the name for that network is in range, essentially corresponding to our own proposed method.
Furthermore, similar to our method for preventing evil twin attacks, location-based solutions need to explicitly ask for the user's consent every time a new access point containing a known SSID is found, which may be less than satisfactory for networks that allow roaming, or when multiple access points provide access to the same network. This is insurmountable, and can be considered a policy decision. However, our solution allows the prevention of network leakage independent of the prevention of evil twin attacks, allowing the user to switch off the second functionality while still preserving his/her privacy.

8.5 Conclusion

In this chapter, we provided some technical solutions and mitigation strategies for improving smartphone security and privacy, aiming to tackle research question **RQ7**. We started by giving some general recommendations to developers, manufacturers, network providers, and researchers that can lead towards mitigating the issues described in previous chapters in the future. With this, we showed that possibilities exist for overcoming the discussed privacy and security issues at the level of the manufacturer, without having to make usability compromises. We also provided more short-term recommendations to users that want to secure their mobile devices in meantime.
As our studies confirmed earlier works in that smartphone users might see security as a one-time (as opposed to a continuous) issue, and that they tend to prefer a set-it-and-forget-it approach over continuous vigilance, we developed a tool that automatically mitigates Wi-Fi attacks on smartphones. This tool, called Wi-Fi PrivacyPolice, can be used to prevent both the leakage of privacy-sensitive information contained in the network identifiers sent out by users' devices, and to prevent Evil Twin (or 'network spoofing') attacks. It does this by (ab-)using Android's `WifiManager` to dynamically enable or disable networks depending on whether they should be available. The tool was first developed as a proof-of-concept, but was afterwards released as a consumer app through official channels, with positive results concerning ratings and number of installations. Years after its inception, PrivacyPolice is still running on thousands of devices, including our own, solving smartphone privacy issues without degrading performance or usability.
The solutions presented in this chapter only provide a part of the puzzle: they aim to solve technical privacy and security issues, mostly caused by unforeseen side-effects of protocols used by mobile devices. However, privacy sensitive data is also part of legitimate data collection done by service providers (such as social networks or search engines). This data collection, while legitimate, is often opaque to users of the service. To overcome this problem, we will approach the privacy issues from a more legal perspective in the next chapter.

Chapter 9

The Privacy API: Facilitating Insights In How One's Own User Data Is Shared

9.1 Introduction **151**
9.2 Definitions **152**
9.3 The API in practice **153**
9.4 Methods of enforcement **154**
9.5 Considerations and limitations **156**
9.6 Related work **156**
9.7 Discussion **157**
9.8 Conclusion **158**

9.1 Introduction

In the previous chapter, we looked at solutions to privacy and security issues from a technical perspective, providing recommendations to users and developers alike to increase mobile security. However, not only technical specificities lead to privacy issues for mobile device users. Indeed, over the past few years, software applications have increasingly been offered in the form of services, often without requiring a direct payment. Instead of – or in addition to – directly asking users to pay for the service, many internet services have shifted to using advertisements or selling user data to third parties as their source of revenue. Historically, these services have been limited to applications such as social networks, though recently non-free service providers (such as mobile operators or hardware manufacturers) have also been using these methods [Lee, 2016; Troianovski, 2013] as a way to increase their revenue, using privacy-sensitive data as a form of currency. In this chapter, we will look at the intersection between legal and technical solutions, aiming to offer a strategy that will allow users to make better privacy decisions.

A problem with paying for services with data instead of money is that, while it is relatively easy for a user to assess the value of the service itself, it can be difficult to assess the value and scope of the provided data. This value is not only determined by how much the service provider profits from it directly (e.g. because it allows the provider to train machine learning models with it, because it allows to profile the user, or because it allows to provide a better service or generate customer value), but also by what third parties can do with it. Moreover, implications of sharing this data with either the service provider or third parties are often unclear. While there have been initiatives and court cases that try to estimate the value of personally identifiable information or users' interests [Steel, Emily and Locke, Callum and Freese, Ben, 2016; Gliman and Glady, 2015], this information is often only a guess at best, and is based on what the average user shares instead of on the data of a specific person.

Furthermore, service providers often provide some form of information inferred from the data (using data mining techniques) to third parties. The user providing the data to the service provider more often than not does not possess the same means (in the form of technology or infrastructure) as the service provider, leaving the user with no idea about which information can be inferred from their data [Reiman, 1995; Solove, 2008]. This has led to users distrusting service providers, opting instead for more 'guerilla' techniques of thwarting data mining algorithms by providing falsified or obfuscated data [Brunton and Nissenbaum, 2013; Brodkin, 2017a]. Similar to users, it can be difficult for consumer organizations to assess in what way service providers are handling user data, or to provide metrics for comparing different service providers in the area of privacy.

To add to this problem, users are often unaware about the data that is collected by the services they are using. Indeed, as we saw in Chapter 6, users are often unaware about the connections being made by their devices. Furthermore, as we demonstrated in Chapter 7, users are often unaware about which data access they granted to apps on their devices in the past.

We suggest that the only way a user can decide whether a service is worth the data he/she has to hand over in order to use it, is by providing him/her with some way of knowing exactly which data is used, and in what way it is used, to generate revenue for the company providing the service. We propose an API, offered by the service provider to every user, allowing to query the exact same data third party marketing companies and advertisers have access to (but limited to information about the specific user accessing the API). The aim is to tackle the Legibility principle of Human-Data Interaction [Mortier et al., 2014].
This chapter does not aim to provide a catch-all solution to all privacy issues, but rather proposes a technique that can be easily implemented for existing systems, providing a valid intermediary step on the path to a complete privacy framework such as the one proposed by Su et al. [Su et al., 2016].

9.2 Definitions

In order to be able to provide a clear explanation of the information flow in the next sections, the different actors are defined first:

Service provider The entity providing an (internet) service, requiring the user to provide some form of personal data in order to use the service. The provided service can be (but is not limited to) a social network (such as Facebook), a webmail service (such as Google's GMail), an app (such as Snapchat) or an Internet-of-Things device or platform (such as a Samsung Smart TV or Google's Brillo).

User The person using the (internet) service, whose data is being monetized by the service provider, e.g. in order to keep the service free or cheap, or to provide extra functionality.

Data broker The entity paying or otherwise offering incentives to the service provider in exchange for users' data. This can be, for example, a third-party analytics or marketing company, an advertising agency or a direct data consumer such as an insurance company gathering information about its clients.

Trusted third party (TTP) A third party (organization) that is trusted by the user, with different users being able to trust different third parties. This can be an institution founded with the purpose of defending consumer rights, such as the Electronic Frontier Foundation, Consumer Watchdog, the American Civil Liberties Union or the Consumer Federation of America. This third party can be either non-profit or for-profit. For most cases, the TTP can be considered to be a consumer organization. However, the term 'trusted third party' (or TTP) is used in this text to include other organizations not strictly fitting the definition of a consumer organization. For example, an organization raising awareness on online privacy could also be considered to fit the definition of a TTP.

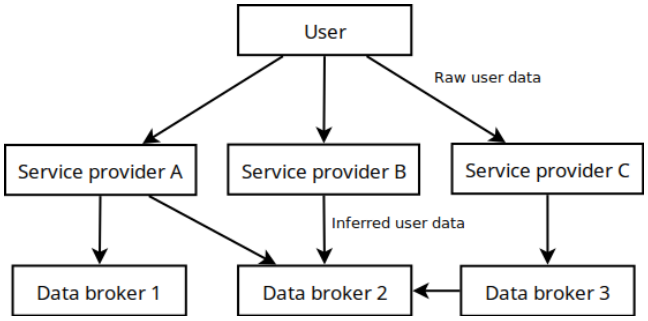


Figure 9.1: Current information flow

A visualization of how information flows between these different parties is available in Figure 9.1. Note that this visualization does not yet include the "trusted third party" as described above; this party will be introduced as part of the new information flow in the next section.

9.3 The API in practice

The implementation of the privacy API aligns with existing API's offered to third party data brokers. These data brokers currently already access data inferred about the users of the service. The privacy API would expose exactly the same interface to the user as is currently exposed to data brokers, with the only difference being that instead of aggregated information, the API only exposes information about the currently logged in user.

The actual responsibility of visualizing the data and its implications to the user is in the hands of trusted third parties. These organizations are able to provide a service to users, allowing them to see exactly which data is available to data brokers through the services they are using. TTP's effectively act in the same way as data brokers, with the exception that users give them permission to handle their data beforehand. These TTP's are then able to aggregate user data from a variety of services and inform the user about which data is shared by which service provider. As will be discussed in Section 9.5, the "trusted third party" is not strictly required: a user could also access the privacy API directly instead of relying on a TTP to visualize their data. However, we envision that most less technically inclined users will rely on a third party organization to visualize and interpret the data. This information flow is visualized in Figure 9.2.

Access control – defining which data will be available through the API – is handled in the same way as other APIs offered by the service provider to third parties. Users can log in directly at the service provider, generating a token that can be passed on to the TTP allowing access to only this user's data via the API.

Note that offering the privacy API does not expose any proprietary information about the service provider or its algorithms: it essentially provides information as a black box and on

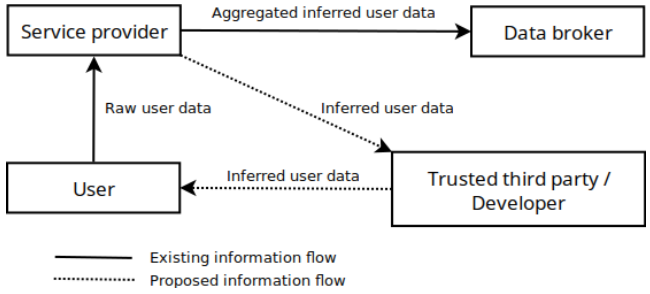


Figure 9.2: Proposed information flow

a need-to-know basis to the user, without revealing how inferences on the data were made. Furthermore, the API does not offer any more information to competitors than is already available to them if they would buy user data from the first party as a data broker, or if they would buy this data through a third party.

The existing API (and thus, the privacy API) can take the form of a regular programming interface, allowing data brokers to query the data directly, but it can also take the form of a more informal 'data dump' containing aggregated user information or statistics, delivered regularly to the data brokers. An API for the second method would work in much the same way, allowing the user (or a TTP acting on behalf of the user) to get an up-to-date data dump of the part of their personal data that is used to generate the information sent to data brokers. Implementing the privacy API in this way shifts the need for user trust from the service provider and its third party data brokers to a "trusted third party". To see why this shift is needed, consider the motives of both these parties. In the case of the service provider or the data broker, the main motive is to keep as many people using the service as possible, which might conflict with the goal of providing transparency. However, for a TTP such as consumer organization, the main added value to the consumer is exactly this providing of information. Moreover, since providing transparency is now effectively decoupled from providing the service, the user is now able to choose between different organizations analyzing the data from privacy APIs. Similarly, it is easier to vet well-known consumer organizations than it is to vet unknown third party data brokers. As a last benefit of this approach, more technically inclined users are now able to build their own application upon the privacy API endpoint, not relying on any third party, and allowing for a diverse ecosystem.

9.4 Methods of enforcement

A crucial part of the implementation of a privacy API is that it should operate in exactly the same way as the API that is provided to data brokers (providing the exact same interface). This requirement should be enforced to ensure that service providers do not stray from the original goal of this API.

The European Union already has strict privacy regulations in place for companies that manage user data and that want to operate within the EU [Council of European Union, 1995, 2016], requiring these companies to provide users with an interface to get a complete overview of all collected user data, and enforcing these requirements by starting lawsuits against non-compliant companies [Belgian Commission for the Protection of Privacy, 2015]. Different companies have been implementing this requirement in different ways, with e.g. Google providing a privacy dashboard containing controls for managing which data is saved in a Google account and an overview of all data generated for that account.[1] While we applaud the efforts made by different companies to be more transparent about their data collection and retention practices, there is no standardized way of presenting this information, requiring the general public to rely on the goodwill of the service providers to provide a clear and complete overview. Moreover, there is no uniform way of presenting this data, making it difficult to compare different services.

A limitation shared by most of these approaches that implement the EU's regulations is that the user can only see which data the service provider has about them, without knowing which part of that data is shared with third party data brokers. Moreover, the service provider may be using data mining techniques to infer extra knowledge from this data, which could also be shared with data brokers. For example, consider the case of a user who only provided the service provider with their purchasing habits (e.g. by using a loyalty card at a store), from which the service provider inferred that this user is a parent. Existing solutions would only show the user an overview of their purchasing habits, while the privacy API would show exactly which inferred information was passed on to the data brokers.

A privacy API could be made mandatory by lawmakers in much the same way as the discussed regulations, but can also provide a more efficient alternative for the service providers themselves. Indeed, implementing a privacy API might require substantially less effort than providing a privacy dashboard, giving service providers a more low-cost way of complying with these regulations. Moreover, this could prove to be a valid alternative to more involved regulations aiming to limit collection of user data by service providers, as the privacy API shows exactly the amount and type of data that is provided to third parties, eliminating guesswork about which data is used only to improve the service itself.

An alternative to enforcement by law is to approach this problem in a similar way to how security audits work. In this case, third party auditing companies could offer certificates of compliance to companies that pass an audit standardized by the industry. This maps directly to standards such as ISO/IEC 27001, where companies are audited to make sure they adhere to security best practices.

Even without being enforced by lawmakers, or without being incentivized by auditing companies, service providers can implement the privacy API to grow trust and goodwill among

[1] Google's activity controls and activity overview are available at `https://myaccount.google.com/activitycontrols` and `https://myactivity.google.com/myactivity`, respectively.

their (potential) users, showing that transparency about data collection and monetization is one of the company's values.

9.5 Considerations and limitations

Note that the main difference between the privacy API and the API that is being offered to data brokers is that the privacy API might disclose more identifiable or personal data about the specific user than the aggregated data that is available through the data broker API. This is a desired side effect of the proposed approach: as data brokers tend to apply deanonymization techniques to the gathered data [Schneier, 2015], in which anonymized or aggregated data is tied back to the original user from whom the data originated, having the most specific information available is important to see which data could in principle be inferred by the third party data broker itself. However, this also means that user data is now managed by an extra party (the TTP), effectively creating a larger attack surface for malicious actors trying to obtain this data and requiring the TTP to be 'trusted' not only with the data itself, but also with securely managing it.

One limitation of the proposed approach is that it is retroactive: users need to be using the service before they can assess which data is shared. As such, this approach can not be used directly to help users decide beforehand whether they want to use a service (in contrast to proactive approaches like P3P). However, since data aggregators and marketeers care mostly about recent data, this could be a valid trade-off for the user to make. Similarly, (anonymized) data about other users of the service could be used by consumer organizations (acting as the trusted third party for multiple users) to paint a general picture about which data is provided and to compare different service providers in the area of privacy.

Secondly, the proposed approach only works for service providers that pass on user data to third parties, which larger service providers are not likely to do, as their data is usually their value. Service providers can still use all user data internally (either to improve their service to users, or to mine this data for interesting patterns). To cater to this, we count on security regulations and audits instead of providing an API (which would put the responsibility in the hands of the user instead of the service provider).

9.6 Related work

Previous approaches to solving this problem have tried to create ontologies or formalized languages for describing privacy-sensitive data, like P3P [Cranor et al., 2002] which never saw widespread adoption despite having been implemented on 15% of the top 5,000 websites [Cranor et al., 2007] and despite being supported in Microsoft's Internet Explorer and Edge browsers. Common criticisms to these approaches is that they often fail to capture the semantics and relationships of the data, that they are difficult to understand for less technical

users, that they make it difficult to arrive at an agreed-upon vocabulary and that the user requirement of defining privacy preferences beforehand can lead to misdirected settings [Perera et al., 2016].
Techniques like TaintDroid [Enck et al., 2010] inspect the information flow going from a user's (Android) device to the internet, specifically tagging personally identifiable information like IMEI numbers and location data being sent by apps to advertisement servers. This provides great insights in how information is shared by mobile apps to third party data brokers. However, it only shows data shared by the app directly, and cannot distinguish between data that is sent to the service provider for internal use (e.g. for improving the service) and data that is sent to the service provider to be shared with data brokers afterwards.
Similarly, the privacy API differs from the 'right of access' requirement in the EU's General Data Protection Regulation (GDPR) [Council of European Union, 2016] mentioned in Section 9.4 in that instead of offering the data that has been provided by the user to the provider, it offers the data that is available to data brokers. This may exclude certain data that will not be passed on, and may also include any extra inferred data that is passed on to data brokers.
Very recently, a related proposal has been made by Su et al. [Su et al., 2016], where the (privacy-sensitive) flow of health data is formalized. They propose to separate the data operator from the data source, mediating between the actual data collector and the data sinks. Our approach differs from the aforementioned one in that it does not require any modifications to current implementations and contracts between data collectors and third party data sinks, pushing for more transparency within the existing process instead. Indeed, our work rather proposes an intermediate step towards a complete privacy framework, offering a solution which can be easily implemented on top of existing systems without requiring large modifications.

9.7 Discussion

The privacy API proposal was presented at IEEE EuroS&P 2017's workshop "Innovations in Mobile Privacy & Security (IMPS)", aiming to provoke a discussion about service providers' data sharing practices. The idea was well received by both the reviewers and the conference attendants, stating that we are long overdue in holding service providers responsible for how they are sharing data with third parties.
Conference attendants noted that a major risk with introducing trusted third parties is that a secondary market might emerge, where these TTP's would be the ones reselling user data to marketing companies. Indeed, the TTP can be seen as a service provider too, being able to share user data with third parties, requiring them to expose a privacy API themselves if they choose to do so. An important distinction is that the visualization of the data is now effectively disconnected from the service that is provided, offering consumers a broader choice. Nonetheless, since the TTP now bears the cost of visualizing information streams between the service provider and data brokers, it also needs to be able to cover this cost.

For this reason, we consider trusted consumer organizations – which already provide similar services, and which rely on consumer trust – to be a good TTP candidate.
It was also mentioned that even without a privacy API, we could require service providers to include the data shared with third parties as part of their privacy dashboards. This shows an interesting side effect of presenting our proposal, which is that the 'third party data sharing' was considered to be an important omission in current approaches. Even without the implementation of a privacy API, legislators should consider including this information as part of their requirements.
The general consensus was that a technical solution in and of itself is not sufficient; legal aspects, and aspects related to regulation are required to be able to implement a solution such as the privacy API. The implementation of the privacy API relies on service providers being required to expose exactly the same information through this API as they do to third parties.

9.8 Conclusion

In this chapter, we formulated an initial proposal for a privacy API which allows complete transparency about the user data that is passed on by service providers to third parties, providing a more long term solution to research question **RQ7**, and catering to collection of privacy sensitive information that happens with the consent of the users, as discussed in Chapter 7. We showed that implementing such an API can have benefits for all parties involved: service providers (in the form of easier compliance and increased user trust), consumers (in the form of more transparency and comfort, and improved information for making decisions), legislators (in the form of regulations that are straightforward to implement and audit) and consumer organizations (in the form of better metrics).
We propose to perform further research based on this concept, implementing a proof-of-concept privacy API for a small set of service providers handling user data. Thus, this chapter can be considered as a call to action for service providers interested to cooperate with researchers in the fields of computer science and privacy. This gives innovative companies a way to show they are serious about privacy, while allowing researchers to assess the feasibility of a privacy API.

Chapter 10

A look at the future: the Internet of Things

Since the start of the research presented in this dissertation, another category of devices has gained popularity: "smart devices", devices that have their functionality extended by adding sensors or internet connectivity to them, have showed up as consumer products. Some examples of these devices include smart refrigerators (allowing to keep an eye on the contents from outside of the house, or to order groceries online), smart lightbulbs (allowing your smartphone to control them), smart thermostats (learning from their owners to automate their schedule), smart locks (allowing to be unlocked by your smartphone) and smart plugs (allowing to switch on or off the power for any device in the household). Other types of connected devices have existed for a longer time, with examples being surveillance cameras or printers connected directly to the internet, or home routers offering an interface to the public internet. These smart devices have not been limited to households either, with industries such as healthcare and public safety also using sensors and internet connectivity to extend the functionality of their devices or to provide some form of automation. The rise of smart devices has lead to the concept of the "Internet of Things" (often shortened to IoT), where everyday devices are provided with extra functionality by connecting them to the global internet.

As we discussed in the introduction to this dissertation, mobile devices often provide a new context for existing vulnerabilities, re-surfacing them or making them more acute. The same comment can be made for IoT devices; these devices are running embedded software, providing complex functionality (in the form of e.g. a web server), while the manufacturers of these products are often not software companies, leading to vulnerabilities in the software of these devices [Barcena and Wueest, 2015]. Adding to this is that manufacturers have little incentive to patch or update these devices after they have been sold to consumers, leaving them vulnerable for the rest of their lifespan. Problems with IoT devices are abundant [Fernandes

et al., 2016; Hunt, 2017], spawning even Twitter accounts[1] created with the sole purpose of pointing out issues with these devices. While we limit ourselves to the discussion of mobile devices in this dissertation, we provide this short addendum about IoT devices to discuss how the same problems will surface for IoT devices.

To see why these security issues are not always obvious, consider the case of a connected lightbulb, allowing its owner to control the lights in his/her home through a smartphone app. If an attacker is able to capture data transmitted to or from this lightbulb (e.g. through the same Wi-Fi attacks that have been possible for decades), he/she might be able to reason on the current occupation of the home, or the state of the members of the household. This could enable burglars who can determine the best moment to enter the home, by using this data to get an indication of how many people are in the house, or to assess whether the members of the household are asleep. Note that even encrypting the datastreams would not completely mitigate this threat: being able to see the only the existence of datastreams might be sufficient to make a calculated guess.

Active attacks on IoT devices can lead to even more dangerous scenarios: researchers have demonstrated the ability for an attacker to strobe connected smart lights at a frequency which may trigger seizures in people suffering from photosensitive epilepsy [Ronen and Shamir, 2016]. Combined with the fact that these same researchers were also able to infect the same lightbulbs with a worm that allows malware to spread across an entire city [Ronen et al., 2016], these attacks can have disastrous effects.

Security issues in IoT devices do not only impact the devices themselves or their owners: in October 2016, hunderds of thousands of IoT devices infected by the Mirai malware were used to mount a Distributed Denial of Service (DDoS) attack on DNS service provider Dyn [Woolf, 2016]. This attack rendered a large part of popular services (such as Amazon, Netflix and Reddit) unavailable for a period of almost 12 hours. The same type of malware has also been used to permanently render the IoT devices themselves unusable, in an attack known as a Permanent Denial of Service (PDoS) attack [Radware Security, 2017]. The Mirai malware is not an isolated case: lots of different IoT malware families exist, many of them being used to execute DDoS attacks [Symantec Security Response, 2016].

We performed a short study with a few of our own personal connected devices, consisting of a Xiaomi Mi Band[2], a Garmin Vívosmart[3], a set of Cubesensors[4] and a Gigaset Elements starter kit[5]. We were able to find all kinds of different security and privacy issues, ranging from not using certificate pinning[6] in the mobile app when sending data to the server (as was

[1] A popular Twitter account dedicated to pointing out problems in IoT devices, which amassed over 200 000 followers at the writing of this text, is `@internetofshit`, available at `https://twitter.com/internetofshit`.

[2] More information about the Xiaomi Mi Band is available at `http://www.mi.com/en/miband/`.

[3] More information about the Garmin Vívosmart is available at `https://buy.garmin.com/en-US/US/p/154886`.

[4] More information about Cubesensors is available at `https://cubesensors.com/`.

[5] More information about Gigaset Elements is available at `http://www.gigaset.com/hq_en/smart-home/`.

[6] As we explained in Section 8.2.2, certificate pinning ensures that the certificate for the service is embedded in the app, instead of requiring it to be validated by the system's certificate chain.

Table 10.1: Results of a short study on IoT devices.

	Mi Band	Vivosmart	Cubesensors	Gigaset Elements
Certificate pinning	✗	✗	✗	✗
Encryption	✓	✓	✗	✓
Informs user about data collection	✗	✓	✗	✓

the case for all four tested devices), over sending a vast amount of personally identifiable information (PII) – such as the smartphone's IMEI and its Wi-Fi MAC address – to the service provider without informing the user (as was the case for the Mi Band), to not having any notable security at all (as was the case for the Cubesensors, sending all gathered sensor data over an unencrypted HTTP connection, and having the keys used for local ZigBee communication stored in plain text on the device's SD card). An overview of this short study is available in Table 10.1.

As an increasing number of devices are becoming internet-enabled, care must be taken that they are not susceptible to the same privacy and security issues as were present in earlier technologies. We envision that regulations will be essential in the IoT space. First, consumers should be informed about data collection practices of the manufacturers of internet-enabled devices through methods such as the privacy labels proposed by Kelley et al. [Kelley et al., 2009], or our own Privacy API from Chapter 9. Second, these manufacturers should be held accountable for their update and security policies, requiring them to provide support and updates for their devices similar to the minimum warranty periods imposed by the European Union. These measures are not only important to the consumers themselves: as we stated before, insecure IoT devices often have an impact on others as well. Indeed, requiring these devices to be secure might well be important to safeguard the security of the internet as we know it.

Part III: Conclusion

In the final part of this dissertation, we looked at ways in which mobile device privacy and security can be improved. Taking into account the results from our studies in the second part of this thesis, we keep in mind the usability factor when presenting such improvements.
We started by providing more technical countermeasures to different groups of involved actors, such as developers, manufacturers and network or service providers, referring back to the issues presented in the first part. Taking into account the fact that smartphone users tend to prefer a set-it-and-forget-it approach to smartphone privacy (again as shown in Part II), we also developed a tool that automatically mitigates Wi-Fi attacks on smartphones. This tool, called Wi-Fi PrivacyPolice was first developed as a proof-of-concept, but was afterwards released as a consumer app through official channels. The app allowed us to smartphone privacy issues without degrading performance or usability.
Going further than only technical solutions to privacy, we also present our thoughts on how we can improve the privacy and transparency for data streams of users' personal and sensitive information, collected by service providers as part of their legitimate operation. Our initial proposal takes the form of a "privacy API", allowing users to see exactly which personal data is shared with third parties by service providers. We showed that implementing such an API can have benefits for all parties involved, and that it allows also consumer organizations to be included in the process. We consider this concept to be a first step towards future work in privacy and security, allowing interested service providers to cooperate with researchers in the fields of computer science and privacy.
We closed the last part of this dissertation with a comparison between the state of Internet of Things (IoT) devices now with that of smartphones and mobile devices some years ago. Indeed, as we showed to be the case with smartphones some years ago in the first part of this dissertation, IoT devices will now provide a new context for existing vulnerabilities, re-surfacing them or making them more acute. We provided some examples of this, and performed a short study ourselves to show that IoT devices in general could benefit from better security.

Chapter 11

Conclusion and Future Work

11.1 Conclusion **167**
11.2 Future work **171**

11.1 Conclusion

In this work, we showed that the shift to mobile devices (and in particular, smartphones) has made a significant impact on privacy and security of internet users. Not only do these devices make it so that people are always connected (as we showed, mobile users connect to 8 different Wi-Fi networks on average in a 30-day period), with more data being able to be gathered about them, they also introduce some new security risks and provide a new context for older security risks. We gave an example of these issues (covering research question **RQ1**), showing that they enable any third party to surreptitiously track the movements of smartphone users by passively monitoring their Wi-Fi signals. We demonstrated that this can be done at a very low cost by using Raspberry Pi's (with software designed to operate within the limited resources of these devices), without requiring active cooperation from the users themselves, and without requiring them to be connected to an access point. We demonstrated this in the context of a three-day long music festival, by tracking 29% of the 100 000 festival visitors. These issues go even further: as we demonstrated later on in a study, these signals allow to map between 64% and 88% of mobile device users to at least one previously visited physical location not in range of our detectors. We showed that this tracking does not need to be privacy-invasive: data can be used anonymously to enable research of movement patterns. We also provide an example of a potential application: an opportunistic communication app that can be used at mass events such as music festivals.

Since publishing our work on Wi-Fi tracking, similar techniques have been employed by many different entities, ranging from public bodies, such as the city of London and the city of New York, to marketing companies trying to build profiles of passers-by. Furthermore, different Mobile Location Analytics (MLA) companies have popped up, specializing in tracking devices through a variety of signals. Notable examples of MLA companies are Renew and Accuware (formerly Navizon). Similarly, commercial alternatives to wardriving databases used to determine a location based on the Wi-Fi networks that are in range, have appeared in the past years, with notable examples in this space being Combain, Skyhook and, again, Navizon.

Some of these companies offer smartphone users a way to opt out of being tracked by their systems, requiring them to enter their device's Wi-Fi MAC address on an opt-out page. Similarly, commercial Wi-Fi location databases sometimes allow networks to be excluded from their lists by not saving their data if the `_nomap` tag is appended to the network's SSID. Unfortunately, not all companies offer a way to opt out. Moreover, due to the fact that these systems are opt out rather than opt in, users need to be aware of this tracking occurring, with the need to opt out of all different MLA services individually. The "Future of Privacy Forum" tries to alleviate this last problem at least for smartphone users with the "Smart Places" initiative[1], offering them with a way to opt out from different MLA services at once. We argue

[1] The Smart Places website, offering an explanation of Wi-Fi tracking, and ways to opt out, is available at `https://smart-places.org/`

that requiring these MLA companies to use an opt in mechanism, offering incentives to users that allow companies to track them, is fairer to users than the current opt out strategy, and should be required by law.
Apart from allowing users to be tracked (both on a small scale, through detectors, as well as on a large scale, based on their broadcasted networks), we showed that other security issues were prevalent for mobile device users (covering research question **RQ2**): over one third of the Wi-Fi networks that smartphone users connect to do not have any encryption enabled (allowing a third party in range to monitor traffic), with 45% of users being susceptible to Evil Twin attacks (where the third party itself can lure the smartphone into connecting automatically to its malicious network). Moreover, 13% of connections made by apps on users devices are insecure, happening both without any form of (SSL) encryption by the app and over unsecured networks, allowing for interception of data by eavesdroppers.
To aid other researchers in assessing the prevalence of other security and privacy issues in wireless networks, we also introduced the Wicability platform, which can be utilized as a tool to quantify the impact and remediation rate of protocol vulnerabilities. We demonstrated the utility of the system by providing a case study based on our own work that shows how the Wicability platform can be used by researchers to assess the impact of such newfound techniques and vulnerabilities.

After explaining the different privacy and security risks inherent to smartphone use in Wi-Fi networks, we went on to create and assess smartphone users' awareness about these risks (covering research question **RQ3**). To raise awareness, we created a setup (named SASQUATCH) consisting of a public display that confronted passers-by with their private information that we were able to infer surreptitiously from their devices. After creating awareness, we used the setup to inform users about how they could improve their smartphone security. This was highly effective, as 76% of users indicated they were not aware about at least a part of the issues we informed them about. Specifically, 52% was not aware that the aforementioned Evil Twin attack is possible. Furthermore, 38% of users we surveyed as part of the study on Wi-Fi Privacy decisions were unaware about unsecure connections being made by their devices.
Even though they are often unaware, users are worried about these dangers: our study on Wi-Fi privacy decisions showed that for 91% of data that was found to be transmitted in an insecure fashion (the 38% number from the previous paragraph), participants were concerned about an eavesdropper having access to this data.
Importantly, we showed that an increased awareness about security and privacy issues did not translate to better security practices (covering research questions **RQ4** and **RQ6**). Indeed, even though users with a higher expertise in computer networks tend to be more aware about the (insecure) connections that are made by apps on their smartphones, they were just as likely as others to connect to insecure Wi-Fi networks.
Analyzing users' decision making regarding permissions for apps on their devices, and tackling research question **RQ5**, we showed that smartphone users are relatively comfortable with

the information access they are granting to apps at the moment this access is allowed. We also showed that when asked about this at a later time, giving explicit examples of which data is available to the app developer, this level of comfort decreased substantially.
A recurring theme in our studies is that the kind of specific, personalized scenarios such as those used in SASQUATCH may help to better inform users about security and privacy issues: this was suggested in the study on Wi-Fi privacy on mobile devices, where participants were worried when presented with specific scenarios about unencrypted connections made by their apps on insecure networks, as well as in the study on app permissions, where users reported a lower comfort level when given examples of data access for different apps on their devices. This finding corresponds to results from earlier studies, such as a study on browser warnings by Felt et al. [Felt et al., 2015] (and many others referenced therein), which describe that people are more likely to comprehend and comply with a security warning if it provides specific, explicit, and comprehensive details about the consequences of ignoring it.

As our studies confirmed earlier works in that smartphone users might see security as a one-time (as opposed to a continuous) issue, and that they tend to prefer a set-it-and-forget-it approach over continuous vigilance, we developed a tool that automatically mitigates Wi-Fi attacks on smartphones (covering research question **RQ7**). This tool, called Wi-Fi Privacy-Police, can be used to prevent both the leakage of privacy-sensitive information contained in the network identifiers sent out by users' devices, and to prevent Evil Twin (or 'network spoofing') attacks, without degrading performance or usability. It does this by (ab-)using Android's `WifiManager` to dynamically enable or disable networks depending on whether they should be available. The tool was first developed as a proof-of-concept, but was afterwards released as a consumer app through official channels, with great results concerning ratings and number of installations.
Apart from our automated solution, we also provided some general recommendations on smartphone security to different stakeholders: the smartphone users themselves, the network providers and ISPs, the developers and the smartphone manufacturers.
Recently, we see that the industry has been implementing measures to provide additional security and privacy to smartphone users. For example, Android O (to be released in the second half of 2017) will update its Wi-Fi stack in order to randomize MAC addresses in probe requests for all supported chipsets, and Google is working with vendors to add support in the firmware as well [Hogben, 2017]. Incidentally, another action taken by Google in Android O is to remove unnecessary Information Elements from probe requests, also thwarting techniques that fingerprint devices based on this information (such as our own [Robyns et al., 2017]), further improving privacy to their users. These changes to device identifiers go even further, also making sure that apps will no longer be able to use permanent device identifiers that can not be reset by the user. This is a step in the right direction, provided that users are clearly informed about the fact that they have the ability to opt-out of tracking by resetting device identifiers that will still be available, such as the Google Play advertising ID, providing that apps will target the newer Android version (apps targeting an older version of Android

will still be able to use the persistent SSAID), and providing device manufacturers are willing to add in support for MAC address randomization.

As a last way to improve privacy and security for mobile device users, we formulated a proposal for more transparency in the way user data that is handled by service providers, and how this data is used for monetization. This proposal takes the form of an API that allows users of a service to see exactly which information is shared by the service provider to third parties. We argued that implementing such an API can have benefits for all parties involved, and that it also allows consumer organizations to be included in the process. While not a complete solution, we hope that this proposal can provide a starting point to researchers in different fields (not only in computer science, but also in e.g. law studies) to reason about how privacy-sensitive information is used in a world where an increasing amount of services is financed by monetizing user data. With grassroots organizations on privacy, such as the Future of Privacy Forum[2] or the CAPrice community[3] emerging, existing consumer rights organizations, such as the Electronic Frontier Foundation[4] and the American Civil Liberties Union[5], putting an increased focus on people's privacy, and lawmakers catching up to existing technologies [Council of European Union, 2016], we are optimistic about smartphone users' privacy in the future. However, these organizations are serving a cause that will never end: with new technologies emerging come new challenges, which will require cooperation between different fields, and between the research community and lawmakers, in order to safeguard the privacy and security of the users. Already, we are seeing laws being passed in the United States to eliminate rules being put in place by the Federal Communications Commission to require users' opt-in consent before selling or sharing web browsing history [Brodkin, 2017b].

Some of the topics in this work required thorough ethical consideration. As we discussed throughout this dissertation, multiple technologies and methods that we developed can be used to gather privacy-sensitive information about random people, and could even aid in stalking. Indeed, when performing our early experiments, we noticed how simple it was to identify our own colleagues, and we were astonished by the amount of personal information we were able to gather about these people (including past whereabouts, names of friends, and the times at which they were in the lab). For this reason, we consulted with both the university's legal team, and with the ethical commissions and/or legal themes of the organizations we worked with, on multiple occasions. As we noted in Section 2.8, anonymization of this type of data is notoriously difficult, and the approach of one-way hashing data that can be used to identify a person is not sufficient. Yet, this is exactly what most Mobile Location Analytics companies are doing. Indeed, in order to implement an opt-out service based on

[2]The website of the Future of Privacy Forum is available at: `https://fpf.org/`.

[3]The website of the Collective Awareness Platform for Privacy Concerns and Expectations (CAPrice) community is available at: `https://www.caprice-community.net/`.

[4]The website of the Electronic Frontier Foundation (EFF) is available at: `https://www.eff.org/`.

[5]The website of the American Civil Liberties Union (ACLU) is available at `https://www.aclu.org/`.

providing the MAC addresses of a device, a MLA company needs to look up any data points corresponding to this exact MAC address, thus being able to uniquely identify the device and its user in their records. The same remark can be made for other entities mentioned in this work, such as consumer organizations acting as the Trusted Third Party in the Privacy API proposal, who also require high ethical standards for managing sensitive user data.
The research leading up to this dissertation started out on very strong security-related topics, related to wireless protocols. As the research progressed, it became clear that these implementation issues were not the only problem, and that increased awareness among the general public was needed. Not only does awareness mitigate the "people problem" in security [Schultz, 2005], the increased consumer demand for security also puts an increased pressure on manufacturers and other companies involved in smartphone development to improve security on their devices. Attempting to raise awareness, we feared that doing so might feel patronizing to the general public. Fortunately, the opposite proved to be true: after demonstrating the SASQUATCH system only once, a large number of requests started coming in for talks, showing that even people who are not knowledgeable about technology are greatly interested in knowing how they can improve security on their devices. This resulted in a number of talks[6] and other ways of public outreach, which showed the importance of informing the general public about security issues (and, more broadly, research issues) that are relevant to them. On the other hand, scaring people just in the name of security is not sufficient: it is of crucial importance that these issues are explained thoroughly, to avoid the risk of the audience assuming all technology is insecure, causing them to be alienated from technology altogether.

11.2 Future work

The data collected using the WiFiPi system, together with the results of the simulation opportunistic routing protocols using data collected in a similar fashion, were originally meant to create and finetune an opportunistic messaging app that could be used in the context of mass events (and, specifically, music festivals). In the early years of smartphone availability, protocols such as Wi-Fi Direct or Apple's Multipeer Connectivity framework were still in their infancy, and implementations differed widely between devices, which hindered creating an actual opportunistic messaging app and testing it on a large scale. Nowadays, Android has a standardized implementation of Wi-Fi Direct that is available across the majority of Android devices, which would allow to develop such an app based on our results.
Furthermore, we expect that similarities between these message passing algorithms and other fields like epidemiology could foster interesting research, providing the ability to compare the spread of messages to the spread of diseases. This could in turn allow for better disease spreading models, influenced by message passing algorithms, and vice versa. Since these

[6] An example of such a talk, given at TEDxGhent 2014, is available on YouTube at `https://www.youtube.com/watch?v=2GpNhYy2l08`.

technologies would also open a new attack vector to malicious actors, allowing the spreading of malware [Ronen et al., 2016], this might lead to a direct mapping between the spread of computer viruses and biological viruses.

A very recent new standard from the Wi-Fi alliance, called Wi-Fi Aware (or Neighbor Awareness Networking), allows supported devices to discover and connect directly to each other without the need for an access point. This technology, which will be supported starting from Android O (to be released in the second half of 2017), would also allow easier development of opportunistic messaging apps. Furthermore, support for Wi-Fi Aware would also entail new privacy issues related to device discovery, which might again have an impact on the discussed tracking technologies. It would be interesting to study what the effect of these new technologies is on the issues described throughout this thesis.

The Privacy API as laid out in Chapter 9 is currently just an initial proposal, presented to other researchers and to the industry as a potential next step in solving ambiguity on the way personal data is handled and monetized by service providers.

Finally, as we explained in Chapter 10, we expect the same privacy and security issues to arise for devices in the context of the Internet of Things: just as smartphones provided a new context for existing vulnerabilities, these devices will again change the threat landscape, re-surfacing vulnerabilities that were deemed low-risk in the past as threats that now have a major impact on privacy-sensitive data. With these devices running insecure software that is rarely updated or patched, there is an interesting future for security and privacy research ahead. We are optimistic about platforms such as IBM Watson or Android Things, which aim – among other things – to provide security as part of the infrastructure and protocols, rather than as part of the application itself.

Appendices

Bijlage A

Nederlandse Samenvatting (Dutch Summary)

Het gebruik van smartphones en andere mobiele toestellen is sterk toegenomen in de voorbije jaren. Deze toegenomen populariteit heeft ook geleid tot een veranderd security- en privacylandschap, doordat persoonlijke apparaten vaker worden uitgerust met een grote hoeveelheid aan sensoren. Deze laten toe om elke stap van de gebruiker te volgen, en beschikken over een veel groter aanvalsoppervlak dan oudere, meer statische apparaten. Dit verleent verschillende soorten actoren, waaronder hackers, overheden en legitieme aanbieders van diensten, toegang tot een grote schat aan gebruikersdata.

In dit proefschrift onderzoeken we wat de huidige status is van privacy en security op smartphones, bepalen en verhogen we het bewustzijn van smartphonegebruikers rond deze problematiek, en voorzien we oplossingen om security en privacy te verhogen op mobiele toestellen. Om aan te tonen hoe eenvoudig het is om data van smartphonegebruikers te verzamelen beschrijven we een mechanisme dat kan gebruikt worden om bezoekers van massa-evenementen te volgen zonder hun medeweten. We laten zien dat dit geïmplementeerd kan worden aan een zeer lage kostprijs, en dat dit ons toelaat om de bewegingen van 29% van de bezoekers van een populair muziekfestival te volgen. We tonen aan hoe deze technieken gebruikt en misbruikt kunnen worden in verschillende scenario's; specifiek door de verzamelde data te gebruiken om verschillende opportunistische routeringsalgoritmes te vergelijken, die op hun beurt gebruikt kunnen worden voor ad-hoc communicatie op massa-evenementen. We voorzien verder ook een open platform voor onderzoekers dat gebruikt kan worden om de impact en remediëringssnelheid van vergelijkbare kwetsbaarheden in draadloze protocollen te kwantificeren.

Om smartphonegebruikers van deze problematiek bewust te maken, en om ze uit te leggen hoe ze zichzelf kunnen beschermen, voorzien we een methode om deze gebruikers te informeren wanneer ze draadloze netwerken gebruiken. Hierbij wordt privacygevoelige (maar

geanonymiseerde) informatie over voorbijgangers weergegeven op een publiek scherm; een opstelling die later ook gebruikt wordt in talks rond securitybewustmaking. Resultaten van onze gebruikersstudies tonen aan dat specifieke, gepersonaliseerde voorbeelden kunnen helpen om gebruikers beter te informeren over security- en privacyproblemen (waarbij het bewustzijn van 76% van de deelnemers in onze studie verhoogd werd), en dat een verhoogd securtitybewustzijn leidt tot een verhoogde bereidheid van 81% van de gebruikers om hun apparaten beter te beveiligen. Onverwachts tonen we in een latere studie aan dat een verhoogd bewustzijn zich niet vertaalt in een betere securityhygiëne.

Hiernaast onderzoeken we ook het privacy- en securitygedrag van smartphonegebruikers in twee verschillende studies. In de eerste studie onderzoeken we hoe bewust gebruikers zich zijn van de verbindingen die gemaakt worden door apps op hun apparaat, waarbij de veiligheid van zowel de Wi-Fi netwerken als van de gemaakte verbindingen in acht worden genomen. In de tweede studie breiden we de Paco ESM tool uit om te onderzoeken wat de redenen zijn voor Androidgebruikers om applicaties te installeren of te verwijderen, en om hun motivatie te kennen voor het toestaan of weigeren van toegang tot één van de verschillende permissies op het moment dat ze deze beslissingen maken. We onderzoeken ook hoe comfortabel en bewust deze gebruikers zich achteraf (enkele weken later) zijn van de gemaakte beslissingen.

We voorzien aanbevelingen voor verschillende belanghebbenden (ontwikkelaars, fabrikanten, aanbieders van netwerken, onderzoekers en gebruikers van mobiele toestellen) omtrent hoe de privacy en security op mobiele toestellen kan verhoogd worden zonder impact te hebben op het gebruiksgemak. Enkele van deze werden reeds geïmplementeerd door de ontwikkelaars van mobiele besturingssystemen (zoals het geval is bij enkele van de gemaakte verbeteringen in Android O). Een aantal van deze aanbevelingen worden geïmplementeerd als een applicatie die automatisch Wi-Fi aanvallen voorkomt op Android smartphones, en die aan het grote publiek beschikbaar wordt gemaakt. Bijkomend formuleren we ook een voorstel voor het verhogen van transparantie bij het delen van gebruikersdata door dienstverleners aan derden.

Appendix B

Survey questions for the study on Wi-Fi privacy

B.1 Recruitment survey questions **178**
B.2 Exit survey questions **178**
B.2.1 Introductory questions **179**
B.2.2 Network questions **179**
B.2.3 General questions **180**
B.2.4 Connection awareness questions **181**
B.2.5 Feedback question **181**

B.1 Recruitment survey questions

Disclaimer: the questions listed below are direct translations of the actual questions in Dutch, and may vary slightly in wording because of this translation.

Q1: Do you own an Android device which you use outside of your own home?
Options: Yes, No

Q2: Did you ever change the settings of your home network? e.g. the Wi-Fi password or the name of the network
Options: Yes, No, Don't know

Q3: Would you be able to explain what WEP, WPA and WPA2 are?
Options: Yes, No

Q4: If yes on the previous question, please explain what WEP, WPA and WPA2 are
Open text response

Q5: How would you rate your own expertise in computer networks?
Options: Very high, High, Average, Low, Very low

Q6: What is your gender?
Options: Male, Female

Q7: What is your birth year?
Open text response

Q8: What is your highest earned degree?
Options: None, Elementary school, Lower part of high school, High school, Bachelor, Master

B.2 Exit survey questions

Disclaimer: the questions listed below are direct translations of the actual questions in Dutch, and may vary slightly in wording because of this translation.

For each survey, at most three Wi-Fi networks the user connected to are chosen. For each network, at most three apps are chosen for which the participant is surveyed.

B.2.1 Introductory questions

Q1-3: Within the context of app <app name>, how concerned are you about the privacy of the following data?

- Username
- Password
- Data (messages, information within the app, pages visited from the app, ...)

Options (for each of the different data types): Not at all concerned, Slightly concerned, Somewhat concerned, Moderately concerned, Extremely concerned
This question is asked for at most three apps that set up a network connection per Wi-Fi network, for a total at most three Wi-Fi networks. This creates at most 27 ($3\ apps \times 3\ networks \times 3\ data\ types$) questions, with the total often being much lower as questions for the same app are consolidated.

Q4: How would you estimate the security of your home Wi-Fi network?
Options: Very bad, Bad, Acceptable, Good, Very Good, Don't know / I don't have wireless Internet at home

B.2.2 Network questions

Help text: *On <connection time> your device was connected to the <network name> Wi-Fi network. You provided the following information about this network: <user response>.*

Q5: In which of the following categories would you put the owner of network <network name>?
Options: Family (own home network), friend of family (someone else's home network), employer (own company network), other company (company network), public institution (city network, museum network, ...), commercial institution (restaurant or cafe network, supermarket network ...), roaming network of home Internet provider (<examples of local ISP's>), other

Q6: To what extent do you agree with the following statement: "The owner of <network name> is permitted to see all information (see previous page) of app <app name>"
Options: Strongly disagree, Disagree, Neither agree or disagree, Agree, Strongly agree

Q7: To what extent do you agree with the following statement: "A random person in the neighborhood of <network name> (e.g. someone standing on the street close to the building) is permitted to see all information (see previous page) of app <app name>"

Options: Strongly disagree, Disagree, Neither agree or disagree, Agree, Strongly agree
This question is only surfaced if the network is not secured with WEP, WPA, WPA2 or WPA-enterprise, *and* if the app is using unencrypted connections.

Q8: How likely would you say that the app <app name> actually sent data over the <network name> network?
Options: Extremely unlikely, Unlikely, Neutral, Likely, Extremely likely

Q9: How would you rate the security of the <network name> network?
Options: Much less secure than my home network, Less secure than my home network, As secure as my home network, More secure than my home network, Much more secure than my home network

Q10: To what extent do you agree with the following statement: "I trust the owner of <network name>"
Options: Strongly disagree, Disagree, Neither agree or disagree, Agree, Strongly agree

B.2.3 General questions

Q11-23: While using the internet, have you ever done any of the following things?
Both this question wording and the relevant tools and options from [Pew research center, 2014] are used. These tools and options are:

- Used a temporary username or email address
- Added a privacy-enhancing browser plugin like DoNotTrackMe or Privacy Badger
- Given inaccurate or misleading information about yourself
- Set your browser to disable or turn off cookies
- Cleared cookies and browser history
- Used a service that allows you to browse the Web anonymously, such as a proxy server, Tor software, or a virtual personal network (VPN)
- Encrypted your phone calls, text messages or email
- Decided not to use a website because they asked for your real name
- Deleted or edited something you posted in the past
- Asked someone to remove something that was posted about you online

- Used a public computer to browse anonymously
- Used a search engine that doesn't keep track of your search history
- Refused to provide information about yourself that wasn't relevant to the transaction

Options: Yes, No, Not applicable to me, Don't know

Q24: Do you think that people should have the ability to use the internet completely anonymously for certain kinds of online activities?
This question is derived directly from [Pew research center, 2014].
Options: Yes, No, Don't know

B.2.4 Connection awareness questions

Q25: The app <app name> has effectively been sending data over the network <network name>. Were you aware of this happening?
Options: Yes, No

B.2.5 Feedback question

Q26: Comments? Questions? Did we miss something? Let us know!
Open text response

Appendix C

Survey questions for the study on Android permissions

C.1 In-situ questions **184**
- **C.1.1 App installation scenario** **184**
- **C.1.2 App removal scenario** **185**
- **C.1.3 Permission grant scenario** **185**
- **C.1.4 Permission deny scenario** **186**

C.2 Exit Survey **186**

Responses to all questions are required.

C.1 In-situ questions

C.1.1 App installation scenario

The order of possible responses to the questions for the in-situ survey is always randomized (with the exception of the 'Other' option, which is always placed last).

Q1: Which factors influenced your decision to install <app>? (select all that apply)

- App rating
- App popularity
- Individual user reviews
- Requested permissions
- The company creating the app
- The app is free / price is reasonable

Q2: Why did you install <app>? (select all that apply)

- The app has fewer permissions than other apps like it
- My friends/family use it
- I want to try it out
- I was required to install it
- The app is part of a product/service that I use
- The app is useful
- The app is cool or fun to use
- I trust the app or the company making the app
- It was the only app of its kind (no other apps provide the same functionality)
- I was offered something in return (e.g. credits, monetary rewards, discount)
- I don't know
- Other: _____

C.1.2 App removal scenario

Q1: Why did you remove <app>? (select all that apply)

- The app required permissions I wasn't comfortable with granting
- I no longer use the app
- To free up space or speed up my device
- Because of advertisements in the app
- Because of in-app purchases
- I didn't like the app
- The app is crashing / very slow
- The app is not working as expected
- I don't know
- Other: _____

C.1.3 Permission grant scenario

Q1: Why did you choose to allow <app> to access your <permission>? (select all that apply)

- I want to use a specific feature that requires this permission
- I think the app won't work otherwise
- I trust the app developer
- Because the app is popular
- I won't be able to grant this permission later
- I have nothing to hide
- I wanted the permission screen to go away
- Nothing bad will happen
- I didn't know I did that
- The app developer already has this information about me
- Other: _____

Q2: To what extent do you agree with the following statement: "I don't mind giving <app> access to my <permission>"?

- Strongly disagree
- Disagree
- Neither agree or disagree
- Agree
- Strongly agree

C.1.4 Permission deny scenario

Q1: Why did you deny <app> to have access to your <permission>? (select all that apply)

- I do not use the specific feature associated with the permission
- I think the app shouldn't need this permission
- I expect the app will still work without this permission
- I consider the permission to be very sensitive
- I don't trust the developer enough to provide this information
- I can always grant it afterwards if I change my mind
- I wanted the permission screen to go away
- I think something bad might happen if I provide this permission
- I don't know
- I didn't know I did that
- Other: _____

C.2 Exit Survey

Each of the questions Q1-Q15 have the same three possible answers:

- Yes
- No

- I don't know what this is / means

Q1: Have you ever blocked another person on a social network?
Q2: Have you ever deleted an online account?
Q3: Have you ever downloaded your historical data from an account (e.g. Google Takeout)?
Q4: Have you ever changed the privacy settings for any of your accounts?
Q5: Have you ever read part or all of an online privacy policy?
Q6: Have you ever decided not to install an app on your mobile device because of permissions it requested?
Q7: Have you ever uninstalled an app on your mobile device because of permissions it used?
Q8: Have you ever declined to give an app permission to do something on your mobile device?
Q9: Have you ever declined to use a website because it asked for information you did not want to provide?
Q10: Have you ever stopped using an Internet service or website because you were concerned about how it might use your personal information?
Q11: Have you ever cleared cookies and/or browser history?
Q12: Have you ever installed software to block ads?
Q13: Have you ever installed software to stop websites from tracking what you do online?
Q14: Have you ever used a password manager?
Q15: Have you ever used account settings to limit the data that could be collected or used?
Q16: Which of the following best describes the time at which you try new technology?

- As soon as the technology is available / among the first people to try it
- Sooner than most people, but not among the first
- Once many people are using it
- Once most people are using it
- I don't usually buy or try out new technology

Q17: When an Internet company collects data about you while you are online, overall how beneficial or harmful is that likely to be for you?

- Extremely beneficial
- Moderately beneficial
- Slightly beneficial

- Neither beneficial nor harmful
- Slightly harmful
- Moderately harmful
- Extremely harmful

In questions Q18-Q22, we used a 5-pt Likert scale to measure comfort.

Q18: How comfortable or uncomfortable are you with online companies collecting data about what you do online?

- Extremely Comfortable
- Moderately Comfortable
- Somewhat Comfortable
- Slightly Comfortable
- Not at all Comfortable

In addition to the 5-pt comfort scale, for questions Q19-Q22 users could also select an option "I don't know the app" if they do not recognize the app in the question. The apps we showed users were ones on their phones, so most of the time apps should be recognized.

Q19: How comfortable would you be with the <app name> app knowing your home and work address? (only surfaced if an app exists that was given the Location permission)

- Extremely Comfortable
- Moderately Comfortable
- Somewhat Comfortable
- Slightly Comfortable
- Not at all Comfortable
- I don't know the app

The question answer options for Q20-Q22, were the same as in Q19.

Q20: How comfortable would you be with the <app name> app knowing the phone numbers of your friends and family? (only surfaced if an app exists that was given the Contacts permission)

Q21: How comfortable would you be with the <app name> app knowing who is calling you? (only surfaced if an app exists that was given the Phone permission)

Q22: How comfortable would you be with the <app name> app seeing the pictures taken with your camera? (only surfaced if an app exists that was given the Storage permission)

Q23: Do you have any feedback for us? Is there anything else you would like to tell us?

Open ended response

Appendix D

Scientific Contributions and Publications

The following list of publications, presented at scientific international conferences, contains work that is part of this dissertation:

[Bonné et al., 2013b] Bonné, B., Barzan, A., Quax, P., and Lamotte, W. (2013b). WiFiPi: Involuntary tracking of visitors at mass events. In *2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), workshop on Autonomic and Opportunistic Communications*. IEEE

[Barzan et al., 2013] Barzan, A., Bonné, B., Quax, P., Lamotte, W., Versichele, M., and Van de Weghe, N. (2013). A comparative simulation of opportunistic routing protocols using realistic mobility data obtained from mass events. In *2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), workshop on Autonomic and Opportunistic Communications*. IEEE

[Bonné et al., 2014] Bonné, B., Quax, P., and Lamotte, W. (2014). Your mobile phone is a traitor! – Raising awareness on ubiquitous privacy issues with SASQUATCH. *International journal on information technologies & Security*, 6(3):393–422

[Bonné et al., 2014] Bonné, B., Lamotte, W., Quax, P., and Luyten, K. (2014). Raising awareness on smartphone privacy issues with sasquatch, and solving them with privacypolice. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 379–381. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering)

[Robyns et al., 2016] Robyns, P., Bonné, B., Quax, P., and Lamotte, W. (2016). Poster: Assessing the impact of 802.11 vulnerabilities using wicability. In *Proceedings of the*

9th ACM Conference on Security & Privacy in Wireless and Mobile Networks, pages 217–218. ACM. Available in Figure D.3.

[Bonné et al., 2017b] Bonné, B., Quax, P., and Lamotte, W. (2017b). The Privacy API: Facilitating Insights In How One's Own User Data Is Shared. In *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), workshop on Innovations in Mobile Privacy & Security (IMPS)*, pages 72–75

[Bonné et al., 2017a] Bonné, B., Peddinti, S. T., Bilogrevic, I., and Taft, N. (2017a). Exploring decision making with android's runtime permission dialogs using in-context surveys. In *Proceedings of the Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017)*, Santa Clara, CA. USENIX Association. **Awarded the IAPP SOUPS Privacy Award.**

[Bonné et al., 2017c] Bonné, B., Rovelo, G., Quax, P., and Lamotte, W. (2017c). Insecure network, unknown connection: Understanding wi-fi privacy assumptions of mobile device users. *Information*, 8(3)

The following work is not part of this dissertation:

[Bonné et al., 2013a] Bonné, B., Barzan, A., Quax, P., and Lamotte, W. (2013a). Poster: Simulating the behavior of opportunistic network protocols at mass events with ns-3. In *The Workshop on ns-3 (WNS3) - held in conjunction with the sixth International Conference on Simulation Tools and Techniques (SIMUTools 2013)*, Cannes. Available in Figure D.2.

[Robyns et al., 2014] Robyns, P., Bonné, B., Quax, P., and Lamotte, W. (2014). Short paper: exploiting wpa2-enterprise vendor implementation weaknesses through challenge response oracles. In *Proceedings of the 2014 ACM conference on Security and privacy in wireless & mobile networks (ACM WiSec '14)*, pages 189–194. ACM

[Robyns et al., 2017] Robyns, P., Bonné, B., Quax, P., and Lamotte, W. (2017). Non-cooperative 802.11 mac layer fingerprinting and tracking of mobile devices. *Security and Communication Networks*, 2017

Other notable talks and presentations:

- TED talk at TEDxGhent 2014: "Your smartphone is leaking your information". Available on YouTube at: `https://www.youtube.com/watch?v=2GpNhYy2l08`.
- Talk at the European Commission's 9th Security and Safety Symposium in 2014: "Your smartphone is a traitor!".
- Talk at UHasselt Science Festival in 2015: "Je smartphone verklikt je". Newspaper article available in Figure D.1.

- Providing a technical explanation of the Facebook privacy case on national news (VTM). Available on the news website at: http://nieuws.vtm.be/binnenland/159382-zo-houdt-facebook-iedereen-de-gaten.

Articles and other resources related to the research presented in this thesis:

- Forbes article mentioning SASQUATCH and Wi-Fi PrivacyPolice: http://www.forbes.com/sites/ianmorris/2015/02/06/android-phones-are-leaking-valuable-information-heres-how-to-stop-them/
- Article on XDA Developers mentioning SASQUATCH and Wi-Fi PrivacyPolice: http://www.xda-developers.com/wifi-data-leaks-and-prevention/
- Reddit Q&A sessions about Wi-Fi PrivacyPolice: http://www.reddit.com/r/Android/comments/2uyw50/wifi_privacypolice_prevents_your_smartphone_or/ and http://www.reddit.com/r/androidapps/comments/2u2ww0/dev_wifi_privacypolice_prevents_your_smartphone/.

HET BELANG VAN LIMBURG
MAANDAG 11 MEI 2015

"Smartphone is een spion"

'Smartphones en privacy' scoren op Science Festival

HASSELT - "De meeste aanwezigen wonen in Hasselt, iemand komt uit Gent en twee van jullie zijn pas aan zee geweest." De 'festivalgangers' van het eerste Science Festival van de UHasselt trokken zaterdag grote ogen tijdens de lezing van Bram Bonné, over hoe we bespioneerd worden door onze eigen gsm. "Je smartphone gaat continu op zoek naar netwerken waar je mee verbonden was", zegt hij. "Op die manier kan je heel gemakkelijk afleiden waar iemand woont of geweest is."

Hanne DE BELIE

Bram Bonné van UHasselt over smartphones. FOTO'S TOM PALMAERS

Meest gesmaakte act van het 'Pukkelpop van de wetenschap' in de Oude Gevangenis: informaticus Bram Bonné. Hij heeft het dan ook over iets dat iedereen aanbelangt: smartphones, security en privacy. Het onderwerp van zijn doctoraat.

Gratis apps

"Je smartphone verklikt je", begint Bonné. "Het is eigenlijk een spion. Een smartphone is iets heel persoonlijks. Je neemt hem overal mee naartoe, zelfs in bed of op de wc. Je gebruikt hem om berichten te sturen, je bankzaken te doen, te daten. Er staat heel veel informatie op over jou, die er zomaar afgehaald kan worden."

Fysieke diefstal is niet het grootste probleem. "De apps die je gratis kunt downloaden, zijn niet voor niets gratis", zegt Bonné. "Ze vergaren heel veel gegevens van de gebruikers, die ze kunnen doorverkopen. Zo weet Facebook heel veel over jou: of je een relatie hebt, je gsm-nummer en mailadres, bij welke bank je bent, wat je sms't... Zelf ben ik zo paranoide dat ik geen Facebook heb."

De zaal is muisstil tijdens zijn betoog. "Iemand aan zee geweest?", vraagt Bonné. Twee aanwezigen steken de vinger op. "Wie woont in Gent?" Presentator Lieven Scheire pleit schuldig. "Iemand pas in Japan geweest? Niemand? Misschien niemand van jullie, maar wel één van de smartphones."

"Je smartphone gaat continu op zoek naar netwerken waar je in het verleden mee verbonden bent geweest", legt hij uit. "Je thuisnetwerk, dat van je kantoor, van de McDonalds waar je een hamburger gegeten hebt. Iedereen die een beetje van computers kent, kan daar met een laptop heel veel informatie uit halen. Er zijn mensen die continu met een laptop in een auto rondrijden, om te kijken met welke netwerken je verbonden bent geweest. Via een database op het internet - Wigel.net - kan elk netwerk gelinkt worden aan locaties. Daaruit kan je heel makkelijk afleiden wie waar geweest is. Als ik van jou weet dat je pas aan zee bent geweest, is dat op zich niet zo erg. Maar een politicus die vaak in een verdacht café komt, is al veel gevoeliger. Je kan ook gemakkelijk afleiden waar iedereen woont. Iemand met slechte bedoelingen kan dat misbruiken."

> **Alles wat je via een open wifi-netwerk verstuurt, hangt in de lucht en kan zomaar worden onderschept**
>
> BRAM BONNÉ

Bonné heeft tijdens zijn doctoraat zelf een gratis app voor Android-gebruikers ontwikkeld: 'Wifi Privacy Police'. "Met die app zorg je ervoor dat de netwerken niet continu worden uitgestuurd", zegt hij. "Op die manier verhoog je de veiligheid van je smartphone."

Vertrouwelijke info

Iets anders waar we volgens Bonné mee moeten opletten, is open wifi-netwerken gebruiken. "Als je bijvoorbeeld op café een open netwerk gebruikt, zullen je data onversleuteld verstuurd worden. Iedereen in de buurt met een laptop, kan daardoor aan vertrouwelijke info geraken. Alles hangt in de lucht en iedereen kan zomaar onderscheppen wat je verstuurt. Websites van Belgische banken zijn bijvoorbeeld wel extra beveiligd. Maar bij andere sites die geen extra sleutel hebben, is de kans reëel dat je data onderschept worden."

"Fan van Lieven Scheire"

HASSELT - Het eerste Science Festival van de UHasselt, "met de wetenschap als headliner", trok gisteren jong en oud naar Hasselt. Onder meer op de affiche: een dj-set met wetenschapsgeluiden, het noorderlicht en Vlaanderens bekendste nerd Lieven Scheire als gastheer.

Tijl en Nette krijgen een handtekening van hun idool Lieven Scheire.

Terwijl professor Jean Manca het Noorderlicht nabootst in de aula van de Oude Gevangenis, draait kunstenaar Frederik De Wilde een wetenschappelijk verantwoorde set. "Deze track is gemaakt van het geluid van zwermen bijen, vloeibaar stikstof en elektromagnetische straling", zegt de dj.

Eén van de publiekstrekkers was dan weer wetenschapsliefhebber Lieven Scheire. "Wij zijn fan van Lieven Scheire en kijken altijd naar 'De Schuur van Scheire'", zeggen Tijl (10) en Nette (9) uit Alken, die een handtekening van hun idool konden bemachtigen. "Wij doen thuis ook allerlei proefjes." (hdb).

Figure D.1: Newspaper article about the talk "Je smartphone verklikt je" at UHasselt science festival in Het Belang van Limburg, May 11th 2015

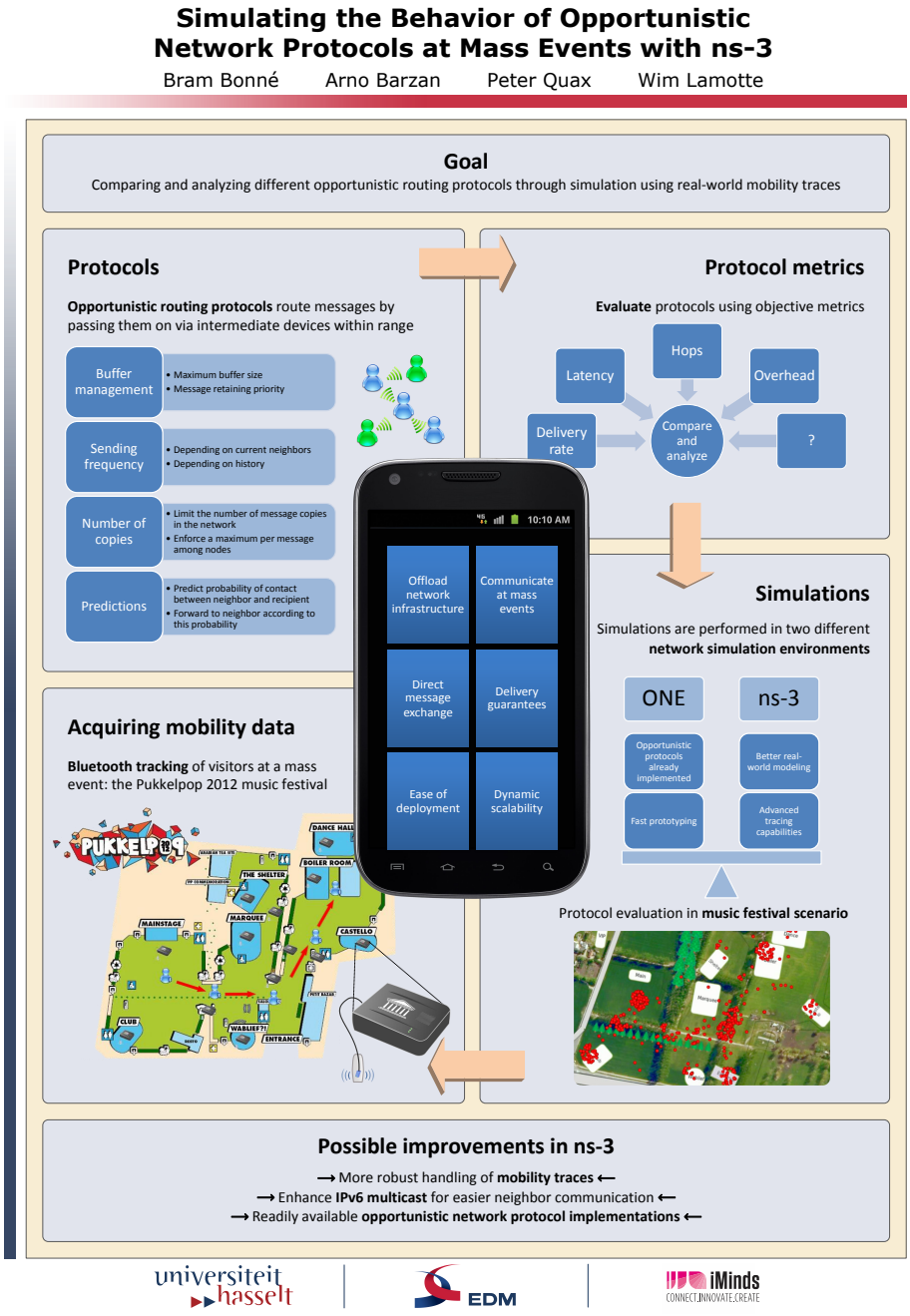


Figure D.2: Poster presented at The Workshop on ns-3 (WNS3) - held in conjunction with the sixth International Conference on Simulation Tools and Techniques (SIMUTools 2013), Cannes.

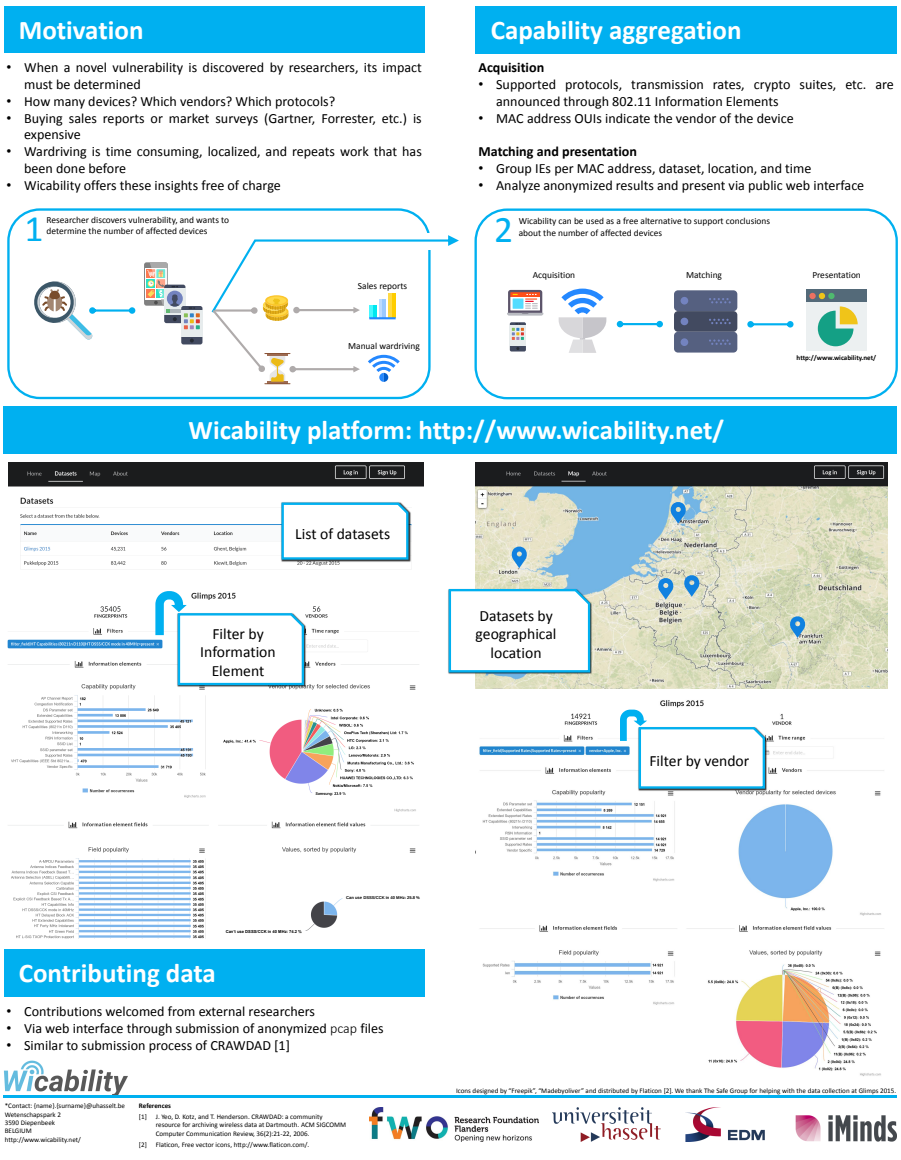


Figure D.3: Poster presented by Pieter Robyns at The 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks, Darmstadt.

Bibliography

Accuware (2017). Accuware Wi-Fi Location Monitor. Available from: http://public.accuware.com/files/Accuware_WiFi_Location_Monitor_Fact_Sheet.pdf.

Almohri, H. M., Yao, D. D., and Kafura, D. (2014). Droidbarrier: Know what is executing on your android. In *Proceedings of the 4th ACM Conference on Data and Application Security and Privacy*, CODASPY. ACM.

Almuhimedi, H., Schaub, F., Sadeh, N., Adjerid, I., Acquisti, A., Gluck, J., Cranor, L. F., and Agarwal, Y. (2015). Your location has been shared 5,398 times!: A field study on mobile app privacy nudging. In *Proceedings of the 33rd annual ACM conference on Human Factors in Computing Systems*, CHI. ACM.

Andel, T. R. and Yasinsac, A. (2006). On the credibility of manet simulations. *Computer*, 39(7):48–54.

Android Open Source project (2013). Bluetooth - Android Developers. Available from: http://developer.android.com/guide/topics/connectivity/bluetooth.html#EnablingDiscoverability.

Anthony, D., Henderson, T., and Kotz, D. (2007). Privacy in location-aware computing environments. *IEEE Pervasive Computing*, 6(4).

Apple Inc. (2012). iOS: Third-party Bluetooth headsets, headphones and keyboards. Available from: http://support.apple.com/kb/ht1664.

Aschenbruck, N., Munjal, A., and Camp, T. (2011). Trace-based mobility modeling for multi-hop wireless networks. *Computer Communications*, 34(6):704–714.

Au, K. W. Y., Zhou, Y. F., Huang, Z., and Lie, D. (2012). Pscout: Analyzing the android permission specification. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS. ACM.

Balebako, R., Jung, J., Lu, W., Cranor, L. F., and Nguyen, C. (2013). Little brothers watching you: Raising awareness of data leaks on smartphones. In *Proceedings of the Ninth Symposium on Usable Privacy and Security*, SOUPS. ACM.

Balebako, R., Leon, P. G., Almuhimedi, H., Kelley, P. G., Mugan, J., Acquisti, A., Cranor, L. F., and Sadeh, N. (2011). Nudging users towards privacy on mobile devices. In *Proceedings of the CHI 2011 Workshop on Persuasion, Nudge, Influence and Coercion*.

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

Barcena, M. B. and Wueest, C. (2015). Insecurity in the internet of things. *Security Response, Symantec*.

Barrera, D., Kayacik, H. G., van Oorschot, P. C., and Somayaji, A. (2010). A methodology for empirical analysis of permission-based security models and its application to android. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS. ACM.

Barzan, A., Bonné, B., Quax, P., Lamotte, W., Versichele, M., and Van de Weghe, N. (2013). A comparative simulation of opportunistic routing protocols using realistic mobility data obtained from mass events. In *2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), workshop on Autonomic and Opportunistic Communications*. IEEE.

Becker, R., Cáceres, R., Hanson, K., Isaacman, S., Loh, J. M., Martonosi, M., Rowland, J., Urbanek, S., Varshavsky, A., and Volinsky, C. (2013). Human mobility characterization from cellular network data. *Commun. ACM*, 56(1):74–82.

Belgian Commission for the Protection of Privacy (2015). The judgment in the facebook case. Available from: `https://www.privacycommission.be/en/news/judgment-facebook-case`.

Bilogrevic, I., Jadliwala, M., Lám, I., Aad, I., Ginzboorg, P., Niemi, V., Bindschaedler, L., and Hubaux, J.-P. (2012). Big brother knows your friends: on privacy of social communities in pervasive networks. *Pervasive Computing*, pages 370–387.

Biondi, P. (2005). Network packet forgery with Scapy. Talk at PacSec.

Bodden, E. (2013). Easily instrumenting android applications for security purposes. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS. ACM.

Bonné, B., Barzan, A., Quax, P., and Lamotte, W. (2013a). Poster: Simulating the behavior of opportunistic network protocols at mass events with ns-3. In *The Workshop on ns-3 (WNS3) - held in conjunction with the sixth International Conference on Simulation Tools and Techniques (SIMUTools 2013)*, Cannes.

Bonné, B., Barzan, A., Quax, P., and Lamotte, W. (2013b). WiFiPi: Involuntary tracking of visitors at mass events. In *2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), workshop on Autonomic and Opportunistic Communications*. IEEE.

Bonné, B., Lamotte, W., Quax, P., and Luyten, K. (2014). Raising awareness on smartphone privacy issues with sasquatch, and solving them with privacypolice. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 379–381. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

Bonné, B., Peddinti, S. T., Bilogrevic, I., and Taft, N. (2017a). Exploring decision making with android's runtime permission dialogs using in-context surveys. In *Proceedings of the Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017)*, Santa Clara, CA. USENIX Association.

Bonné, B., Quax, P., and Lamotte, W. (2014). Your mobile phone is a traitor! – Raising awareness on ubiquitous privacy issues with SASQUATCH. *International journal on information technologies & Security*, 6(3):393–422.

Bonné, B., Quax, P., and Lamotte, W. (2017b). The Privacy API: Facilitating Insights In How One's Own User Data Is Shared. In *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), workshop on Innovations in Mobile Privacy & Security (IMPS)*, pages 72–75.

Bonné, B., Rovelo, G., Quax, P., and Lamotte, W. (2017c). Insecure network, unknown connection: Understanding wi-fi privacy assumptions of mobile device users. *Information*, 8(3).

Bouton, M. E. (2007). *Learning and behavior: A contemporary synthesis.* Sinauer Associates.

Boyles, J. L., Smith, A., and Madden, M. (2012). Privacy and data management on mobile devices. *Pew Internet & American Life Project, September*.

Bratus, S., Cornelius, C., Kotz, D., and Peebles, D. (2008). Active behavioral fingerprinting of wireless devices. In *Proceedings of the first ACM conference on Wireless network security*, pages 56–61. ACM.

Bravo-Lillo, C., Cranor, L., Komanduri, S., Schechter, S., and Sleeper, M. (2014). Harder to ignore? revisiting pop-up fatigue and approaches to prevent it. In *10th Symposium On Usable Privacy and Security*, SOUPS. USENIX Association.

Brignull, H. and Rogers, Y. (2003). Enticing people to interact with large public displays in public spaces. In *INTERACT*.

Brodkin, J. (2017a). After vote to kill privacy rules, users try to "pollute" their Web history. Available from: https://arstechnica.com/information-technology/2017/04/after-vote-to-kill-privacy-rules-users-try-to-pollute-their-web-history/.

Brodkin, J. (2017b). For sale: Your private browsing history. Available from: https://arstechnica.com/tech-policy/2017/03/for-sale-your-private-browsing-history/.

Brunton, F. and Nissenbaum, H. (2013). Political and ethical perspectives on data obfuscation. *Privacy, due process and the computational turn: The philosophy of law meets the philosophy of technology*, pages 164–188.

Bugiel, S., Heuser, S., and Sadeghi, A.-R. (2013). Flexible and fine-grained mandatory access control on android for diverse security and privacy policies. In *Proceedings of the 22Nd USENIX Conference on Security*, SEC. USENIX Association.

Camp, T., Boleng, J., and Davies, V. (2002). A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing*, 2(5):483–502.

Chin, E., Felt, A. P., Sekar, V., and Wagner, D. (2012). Measuring user confidence in smartphone security and privacy. In *Proceedings of the Eighth Symposium on Usable Privacy and Security (SOUPS '12)*, page 1. ACM.

Cisco Systems (2013). Presence analytics. White Paper. Available from: https://meraki.cisco.com/lib/pdf/meraki_whitepaper_presence.pdf.

Clark, J. W., Snyder, P., McCoy, D., and Kanich, C. (2015). I saw images i didn't even know i had: Understanding user perceptions of cloud storage privacy. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1641–1644. ACM.

Consolvo, S., Jung, J., Greenstein, B., Powledge, P., Maganis, G., and Avrahami, D. (2010). The wi-fi privacy ticker: improving awareness & control of personal information exposure on wi-fi. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 321–330. ACM.

Council of European Union (1995). Directive on the protection of individuals with regard to the processing of personal data and on the free movement of such data. *L281*, pages 31–50.

Council of European Union (2016). Regulation on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation). *L119*, pages 1–88.

Cranor, L. F., Langheinrich, M., Marchiori, M., Presler-Marshall, M., and Reagle, J. (2002). The platform for privacy preferences 1.0 (p3p1.0) specification. *W3C recommendation*, 16.

Cranor, L. F., McDonald, A. M., Egelman, S., and Sheng, S. (2007). 2006 privacy policy trends report. *CyLab Privacy Internet Group*.

Cunche, M. and Boreli, R. (2012). I know who you will meet this evening! Linking wireless devices using Wi-Fi probe requests. *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–9.

Dainotti, A., Gargiulo, F., Kuncheva, L. I., Pescapè, A., and Sansone, C. (2010). Identification of traffic flows hiding behind tcp port 80. In *2010 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE.

Davies, J. (2005). Non-broadcast Wireless Networks with Microsoft Windows. *Microsoft TechNet*. Available from: `http://technet.microsoft.com/en-us/library/bb726942.aspx#EDAA`.

Del Duca Almeida, V., Oliveira, A. B., Macedo, D. F., and Nogueira, J. M. S. (2012). Performance evaluation of MANET and DTN routing protocols. In *Wireless Days (WD), 2012 IFIP*, pages 1–6. Ieee.

Dourish, P., Grinter, E., Delgado de la Flor, J., and Joseph, M. (2004). Security in the wild: User strategies for managing security as an everyday, practical problem. *Personal Ubiquitous Computing*, 8(6):391–401.

Enck, W., Gilbert, P., Chun, B.-G., Cox, L. P., Jung, J., McDaniel, P., and Sheth, A. N. (2010). Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones. In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, OSDI. USENIX Association.

Eurostat (2016). Ict usage in households and by individuals. Available online: `http://ec.europa.eu/eurostat/web/digital-economy-and-society/data/database` (acessed on 27 April 2017).

Evans, B. (2016). Paco – applying computational methods to scale qualitative methods. In *Ethnographic Praxis in Industry Conference Proceedings*. Wiley Online Library.

Felt, A. P., Ainslie, A., Reeder, R. W., Consolvo, S., Thyagaraja, S., Bettes, A., Harris, H., and Grimes, J. (2015). Improving ssl warnings: Comprehension and adherence. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 2893–2902, New York, NY, USA. ACM. Available from: http://doi.acm.org/10.1145/2702123.2702442.

Felt, A. P., Chin, E., Hanna, S., Song, D., and Wagner, D. (2011). Android permissions demystified. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS. ACM.

Felt, A. P., Egelman, S., Finifter, M., Akhawe, D., and Wagner, D. (2012a). How to ask for permission. In *Proceedings of 7th Usenix conference on Hot Topics in Security (HotSec)*.

Felt, A. P., Egelman, S., and Wagner, D. (2012b). I've got 99 problems, but vibration ain't one: A survey of smartphone users' concerns. In *Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, SPSM. ACM.

Felt, A. P., Ha, E., Egelman, S., Haney, A., Chin, E., and Wagner, D. (2012c). Android permissions: User attention, comprehension, and behavior. In *Proceedings of the Eighth Symposium on Usable Privacy and Security*, SOUPS. ACM.

Fernandes, E., Jung, J., and Prakash, A. (2016). Security Analysis of Emerging Smart Home Applications. In *Proceedings of the 37th IEEE Symposium on Security and Privacy*.

Fluhrer, S., Mantin, I., and Shamir, A. (2001). Weaknesses in the key scheduling algorithm of rc4. In *International Workshop on Selected Areas in Cryptography*, pages 1–24. Springer.

Friedman, B., Hurley, D., Howe, D. C., Felten, E., and Nissenbaum, H. (2002). Users' conceptions of web security: a comparative study. In *CHI'02 extended abstracts on Human factors in computing systems*, pages 746–747. ACM.

Georgiev, M., Iyengar, S., Jana, S., Anubhai, R., Boneh, D., and Shmatikov, V. (2012). The most dangerous code in the world: Validating ssl certificates in non-browser software. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS '12)*, pages 38–49, New York, NY, USA. ACM.

Gibler, C., Crussell, J., Erickson, J., and Chen, H. (2012). Androidleaks: automatically detecting potential privacy leaks in android applications on a large scale. In *International Conference on Trust and Trustworthy Computing*. Springer.

Gliman, P. and Glady, N. (2015). What's The Value Of Your Data? Available from: https://techcrunch.com/2015/10/13/whats-the-value-of-your-data/.

Gorla, A., Tavecchia, I., Gross, F., and Zeller, A. (2014). Checking app behavior against app descriptions. In *Proceedings of the 36th International Conference on Software Engineering*, ICSE. ACM.

Grasic, S., Davies, E., Lindgren, A., and Doria, A. (2011). The evolution of a DTN routing protocol – PRoPHETv2. In *Proceedings of the 6th ACM workshop on Challenged networks*, number 2 in CHANTS '11, pages 27–30, New York, NY, USA. ACM.

Grasic, S. and Lindgren, A. (2012). An analysis of evaluation practices for DTN routing protocols. *Proceedings of the seventh ACM international workshop on Challenged networks - CHANTS '12*, page 57.

Günther, O. and Spiekermann, S. (2005). RFID and the Perception of Control: The Consumer's View. *Commun. ACM*, 48(9):73–76.

Harbach, M., Hettig, M., Weber, S., and Smith, M. (2014). Using personal examples to improve risk communication for security & privacy decisions. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems*, CHI. ACM.

Harris, M. A., Brookshire, R., and Chin, A. G. (2016). Identifying factors influencing consumers' intent to install mobile applications. *International Journal of Information Management*.

Harris, M. A., Brookshire, R., Patten, K., and Regan, B. (2015). Mobile application installation influences: have mobile device users become desensitized to excessive permission requests? In *Proceedings of the Twentieth Americas Conference on Information Systems*, AMCIS.

He, C. and Mitchell, J. C. (2005). Security Analysis and Improvements for IEEE 802.11i. In *The 12th Annual Network and Distributed System Security Symposium (NDSS'05)*, pages 90–110, Stanford.

Henderson, T. R., Roy, S., Floyd, S., and Riley, G. F. (2006). ns-3 project goals. In *Proc. ACM Workshop on ns-2: the IP network simulator (WNS2 '06)*, New York, NY, USA.

Hogben, G. (2017). Changes to Device Identifiers in Android O. Available from: `https://android-developers.googleblog.com/2017/04/changes-to-device-identifiers-in.html`.

Hormuth, S. E. (1986). The sampling of experiences in situ. *Journal of Personality*.

Hornyack, P., Han, S., Jung, J., Schechter, S., and Wetherall, D. (2011). These aren't the droids you're looking for: Retrofitting android to protect data from imperious applications. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS. ACM.

Hunt, T. (2017). Data from connected CloudPets teddy bears leaked and ransomed, exposing kids' voice messages. Available from: `https://www.troyhunt.com/data-from-connected-cloudpets-teddy-bears-leaked-and-ransomed-exposing-kids-voice-messages/`.

IEEE Standards Association (2012). Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. In *IEEE Standard for Information technology*, number March in 2012, chapter 11. IEEE Computer Society, New York.

Ikram, M., Vallina-Rodriguez, N., Seneviratne, S., Kaafar, M. A., and Paxson, V. (2016). An analysis of the privacy and security risks of android vpn permission-enabled apps. In *Proceedings of the 2016 ACM Internet Measurement Conference*, pages 349–364. ACM.

Jung, J., Han, S., and Wetherall, D. (2012). Short paper: Enhancing mobile application permissions with runtime feedback and constraints. In *Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, SPSM. ACM.

Kang, R., Dabbish, L., Fruchter, N., and Kiesler, S. (2015). "my data just goes everywhere:" user mental models of the internet and implications for privacy and security. In *Eleventh Symposium On Usable Privacy and Security (SOUPS '15)*, pages 39–52.

Kelley, P. G., Bresee, J., Cranor, L. F., and Reeder, R. W. (2009). A nutrition label for privacy. In *Proceedings of the 5th Symposium on Usable Privacy and Security*, page 4. ACM.

Kelley, P. G., Consolvo, S., Cranor, L. F., Jung, J., Sadeh, N., and Wetherall, D. (2012). A conundrum of permissions: Installing applications on an android smartphone. In *Proceedings of the 16th International Conference on Financial Cryptography and Data Security*, FC. Springer-Verlag.

Kelley, P. G., Cranor, L. F., and Sadeh, N. (2013). Privacy as part of the app decision-making process. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI. ACM.

Keränen, A. and Ott, J. (2007). Increasing Reality for DTN Protocol Simulations. Technical report, Helsinki University of Technology.

Keränen, A., Ott, J., Kärkkäinen, T., and Ker, A. (2009). The ONE Simulator for DTN Protocol Evaluation. In *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, New York, NY, USA. ICST.

Khabbaz, M., Assi, C., and Fawaz, W. (2012). Disruption-tolerant networking: A comprehensive survey on recent developments and persisting challenges. *Communications Surveys Tutorials, IEEE*, 14(2):607–640.

Kim, Y. S., Tian, Y., Nguyen, L. T., and Tague, P. (2013). Poster: LAPWiN: Location-Aided Probing in Wi-Fi Networks. In *IEEE Symposium on Security & Privacy, San Francisco*.

Kindberg, T., O'Neill, E., Bevan, C., Kostakos, V., Stanton Fraser, D., and Jay, T. (2008). Measuring trust in wi-fi hotspots. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 173–182, New York, NY, USA. ACM.

Klasnja, P., Consolvo, S., Jung, J., Greenstein, B. M., LeGrand, L., Powledge, P., and Wetherall, D. (2009). "When I am on Wi-Fi, I am fearless": privacy concerns & practices in everyday Wi-Fi use. In *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*, page 1993, New York, NY, USA. ACM Press.

Klieber, W., Flynn, L., Bhosale, A., Jia, L., and Bauer, L. (2014). Android taint flow analysis for app sets. In *Proceedings of the 3rd ACM SIGPLAN International Workshop on the State of the Art in Java Program Analysis*, SOAP. ACM.

Könings, B., Schaub, F., and Weber, M. (2013). Who, how, and why? Enhancing privacy awareness in Ubiquitous Computing. In *PerCom Workshops*, pages 364–367.

Kowitz, B. and Cranor, L. (2005). Peripheral privacy notifications for wireless networks. In *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*, WPES '05, pages 90–96, New York, NY, USA. ACM.

Kung, H. T. and Vlah, D. (2003). Efficient location tracking using sensor networks. In *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, volume 3, pages 1954 –1961 vol.3.

Larson, R. and Csikszentmihalyi, M. (1983). The experience sampling method. *New Directions for Methodology of Social & Behavioral Science*.

Lee, K., Hong, S., Kim, S. J., Rhee, I., and Chong, S. (2009). SLAW: A New Mobility Model for Human Walks. *IEEE INFOCOM 2009 - The 28th Conference on Computer Communications*, pages 855–863.

Lee, M.-J. (2016). Samsung Adds More Ads to Its TVs. Available from: `http://www.wsj.com/articles/samsung-adds-more-ads-to-its-tvs-1464600977`.

Lella, A. and Lipsman, A. (2016). 2016 u.s. cross-platform future in focus. Available online: `https://www.comscore.com/Insights/Presentations-and-Whitepapers/2016/2016-US-Cross-Platform-Future-in-Focush` (acessed on 27 April 2017).

Lin, J., Amini, S., Hong, J. I., Sadeh, N., Lindqvist, J., and Zhang, J. (2012). Expectation and purpose: Understanding users' mental models of mobile app privacy through crowdsourcing. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp. ACM.

Lin, J., Liu, B., Sadeh, N., and Hong, J. I. (2014). Modeling users' mobile app privacy preferences: Restoring usability in a sea of permission settings. In *Symposium on Usable Privacy and Security*, SOUPS.

Lindgren, A., Doria, A., and Schelén, O. (2003). Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 7(3):19.

Lindqvist, J., Aura, T., Danezis, G., Koponen, T., Myllyniemi, A., Mäki, J., and Roe, M. (2009). Privacy-preserving 802.11 access-point discovery. In *Proceedings of the Second ACM Conference on Wireless Network Security*, WiSec '09, pages 123–130, New York, NY, USA. ACM.

Liu, B., Andersen, M. S., Schaub, F., Almuhimedi, H., Zhang, S. A., Sadeh, N., Agarwal, Y., and Acquisti, A. (2016). Follow my recommendations: A personalized privacy assistant for mobile app permissions. In *Twelfth Symposium on Usable Privacy and Security*, SOUPS. USENIX Association. Available from: https://www.usenix.org/conference/soups2016/technical-sessions/presentation/liu.

Liu, B., Lin, J., and Sadeh, N. (2014). Reconciling mobile app privacy and usability on smartphones: Could user privacy profiles help? In *Proceedings of the 23rd International Conference on World wide web*, WWW. ACM.

Madden, M. and Rainie, L. (2015). Americans' Attitudes About Privacy, Security and Surveillance. http://www.pewinternet.org/2015/05/20/americans-attitudes-about-privacy-security-and-surveillance/.

Marlinspike, M. (2009). New tricks for defeating SSL in practice. *BlackHat DC, February*.

Martin, J., Mayberry, T., Donahue, C., Foppe, L., Brown, L., Riggins, C., Rye, E. C., and Brown, D. (2017). A study of mac address randomization in mobile devices and when it fails. *arXiv preprint arXiv:1703.02874*.

Micinski, K., Votipka, D., Stevens, R., Kofinas, N., Mazurek, M. L., and Foster, J. S. (2017). User interactions and permission use on android. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI. ACM.

Mortier, R., Haddadi, H., Henderson, T., McAuley, D., and Crowcroft, J. (2014). Human-data interaction: the human face of the data-driven society. *Available at SSRN 2508051*.

Narayanan, A. and Shmatikov, V. (2009). De-anonymizing Social Networks. In *30th IEEE Symposium on Security and Privacy*, Berkeley, CA.

Nielsen (2012). America's New Mobile Majority: a Look at Smartphone Owners in the U.S. Available from: http://blog.nielsen.com/nielsenwire/online_mobile/who-owns-smartphones-in-the-us/.

Nissenbaum, H. (2004). Privacy as contextual integrity. *Washington Law Review*.

Norberg, P. A., Horne, D. R., and Horne, D. A. (2007). The privacy paradox: Personal information disclosure intentions versus behaviors. *Journal of Consumer Affairs*, 41(1):100–126.

Pandita, R., Xiao, X., Yang, W., Enck, W., and Xie, T. (2013). Whyper: Towards automating risk assessment of mobile applications. In *Proceedings of the 22Nd USENIX Conference on Security*, SEC. USENIX Association.

Perera, C., Liu, C., Ranjan, R., Wang, L., and Zomaya, A. Y. (2016). Privacy-knowledge modeling for the internet of things: A look back. *Computer*, 49(12):60–68.

Pew research center (2014). Pew research center's internet project/gfk privacy panel survey #2 topline. Available online: `http://www.pewinternet.org/files/2015/05/Privacy-and-Security-Attitudes-5.19.15_Topline_FINAL.pdf` (acessed on 27 April 2017).

Poushter, J. (2016). Smartphone ownership and internet usage continues to climb in emerging economies. *Pew Research Center*.

Radenkovic, M. and Grundy, A. (2011). Framework for utility driven congestion control in delay tolerant opportunistic networks. *2011 7th International Wireless Communications and Mobile Computing Conference*, pages 448–454.

Radware Security (2017). "BrickerBot" Results In PDoS Attack. Available from: `https://security.radware.com/ddos-threats-attacks/brickerbot-pdos-permanent-denial-of-service/`.

Reiman, J. H. (1995). Driving to the panopticon: A philosophical exploration of the risks to privacy posed by the highway technology of the future. *Santa Clara Computer & High Tech. LJ*, 11:27.

Robyns, P., Bonné, B., Quax, P., and Lamotte, W. (2014). Short paper: exploiting wpa2-enterprise vendor implementation weaknesses through challenge response oracles. In *Proceedings of the 2014 ACM conference on Security and privacy in wireless & mobile networks (ACM WiSec '14)*, pages 189–194. ACM.

Robyns, P., Bonné, B., Quax, P., and Lamotte, W. (2016). Poster: Assessing the impact of 802.11 vulnerabilities using wicability. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pages 217–218. ACM.

Robyns, P., Bonné, B., Quax, P., and Lamotte, W. (2017). Non-cooperative 802.11 mac layer fingerprinting and tracking of mobile devices. *Security and Communication Networks*, 2017.

Ronen, E., O'Flynn, C., Shamir, A., and Weingarten, A.-O. (2016). Iot goes nuclear: Creating a zigbee chain reaction. Cryptology ePrint Archive, Report 2016/1047. `http://eprint.iacr.org/2016/1047`.

Ronen, E. and Shamir, A. (2016). Extended functionality attacks on iot devices: The case of smart lights. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 3–12. IEEE.

Rose, I. and Welsh, M. (2010). Mapping the urban wireless landscape with Argos. *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems - SenSys '10*, page 323.

Roth, V., Polak, W., Rieffel, E., and Turner, T. (2008). Simple and effective defense against evil twin access points. In *Proceedings of the First ACM Conference on Wireless Network Security*, WiSec '08, pages 220–235, New York, NY, USA. ACM.

Sarma, B. P., Li, N., Gates, C., Potharaju, R., Nita-Rotaru, C., and Molloy, I. (2012). Android permissions: A perspective combining risks and benefits. In *Proceedings of the 17th ACM Symposium on Access Control Models and Technologies*, SACMAT. ACM.

Schneier, B. (2015). *Data and Goliath: The hidden battles to collect your data and control your world*. WW Norton & Company. ISBN: 978-0393244816.

Schultz, E. (2005). The human factor in security. *Computers & Security*, 24(6):425–426.

Shaik, A., Seifert, J., Borgaonkar, R., Asokan, N., and Niemi, V. (2016). Practical attacks against privacy and availability in 4g/lte mobile communication systems. In *23nd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, February 21-24, 2016*.

Shebaro, B., Oluwatimi, O., Midi, D., and Bertino, E. (2014). Identidroid: Android can finally wear its anonymous suit. *Transactions on Data Privacy*.

Shklovski, I., Mainwaring, S. D., Skúladóttir, H. H., and Borgthorsson, H. (2014). Leakiness and creepiness in app space: Perceptions of privacy and mobile app use. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems*, CHI. ACM.

Smith, M. (2016). Usable Security – The Source Awakens. *Usenix Enigma*.

Solove, D. J. (2008). Data mining and the security-liberty debate. *The University of Chicago Law Review*, 75(1):343–362.

Spradlin, L. (2012). Eric Schmidt Reveals New Activation Numbers. Available from: `http://www.androidpolice.com/2012/09/05/eric-schmidt-reveals-some-new-activation-numbers-1-3-million-android-devices-activated-each-day/`.

Spreitzenbarth, M., Freiling, F., Echtler, F., Schreck, T., and Hoffmann, J. (2013). Mobile-sandbox: Having a deeper look into android applications. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC. ACM.

Spyropoulos, T., Psounis, K., and Raghavendra, C. S. (2005). Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, WDTN '05, pages 252–259, New York, NY, USA. ACM.

Steel, Emily and Locke, Callum and Freese, Ben (2016). How much is your personal data worth? Available from: `https://www.ft.com/cms/s/2/927ca86e-d29b-11e2-88ed-00144feab7de.html`.

Stevens, R., Ganz, J., Filkov, V., Devanbu, P., and Chen, H. (2013). Asking for (and about) permissions used by android apps. In *2013 10th Working Conference on Mining Software Repositories (MSR)*.

Su, X., Hyysalo, J., Rautiainen, M., Riekki, J., Sauvola, J., Maarala, A. I., Hirvonsalo, H., Li, P., and Honko, H. (2016). Privacy as a service: Protecting the individual in healthcare data processing. *Computer*, 49(11):49–59.

Swanson, C., Urner, R., and Lank, E. (2010). Naïve security in a wi-fi world. In *IFIP International Conference on Trust Management*, pages 32–47. Springer.

Symantec Security Response (2016). IoT devices being increasingly used for DDoS attacks. Available from: `https://www.symantec.com/connect/blogs/iot-devices-being-increasingly-used-ddos-attacks`.

Tews, E. and Beck, M. (2009). Practical attacks against wep and wpa. In *Proceedings of the Second ACM Conference on Wireless Network Security (WiSec '09)*, pages 79–86, New York, NY, USA. ACM. Available from: `http://doi.acm.org/10.1145/1514274.1514286`.

Tews, E., Weinmann, R.-P., and Pyshkin, A. (2007). Breaking 104 bit wep in less than 60 seconds. In *International Workshop on Information Security Applications*, pages 188–202. Springer.

Thompson, C., Johnson, M., Egelman, S., Wagner, D., and King, J. (2013). When it's better to ask forgiveness than get permission: Attribution mechanisms for smartphone resources. In *Proceedings of the Ninth Symposium on Usable Privacy and Security*, SOUPS. ACM.

Troianovski, A. (2013). Phone Firms Sell Data on Customers. Available from: `http://www.wsj.com/articles/SB10001424127887323463704578497153556847658`.

Urban, J. M., Hoofnagle, C. J., and Li, S. (2012). Mobile phones and privacy. *BCLT Research Paper Series*.

Vahdat, A. and Becker, D. (2000). Epidemic Routing for Partially Connected Ad Hoc Networks. Technical report, Duke University.

Varga, A. and Hornig, R. (2008). An overview of the OMNeT++ simulation environment. In *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, Simutools '08, pages 60:1—-60:10, ICST, Brussels, Belgium, Belgium. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

Versichele, M., Neutens, T., Delafontaine, M., and Van de Weghe, N. (2012). The use of Bluetooth for analysing spatiotemporal dynamics of human movement at mass events: A case study of the Ghent Festivities. *Applied Geography*, 32(2):208–220.

Vukadinovic, V. and Mangold, S. (2011). Opportunistic wireless communication in theme parks: a study of visitors mobility. In *Proc. ACM Int. Workshop on Challenged Networks (CHANTS'11)*, pages 3–8, New York, NY, USA.

Warshaw, J., Matthews, T., Whittaker, S., Kau, C., Bengualid, M., and Smith, B. A. (2015). Can an algorithm know the "real you"?: Understanding people's reactions to hyper-personal analytics systems. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI. ACM.

Wei, X., Gomez, L., Neamtiu, I., and Faloutsos, M. (2012). Permission evolution in the android ecosystem. In *Proceedings of the 28th Annual Computer Security Applications Conference*, ACSAC. ACM.

Whitten, A. and Tygar, J. D. (1999). Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In *Proceedings of the 8th USENIX Security Symposium*, volume 99, page 16. McGraw-Hill.

WiGLE.net (2016 (accessed May 10, 2016)). *Wireless Network Mapping*. `https://wigle.net/`.

Wijesekera, P., Baokar, A., Hosseini, A., Egelman, S., Wagner, D., and Beznosov, K. (2015). Android permissions remystified: A field study on contextual integrity. In *Proceedings of the 24th USENIX Conference on Security Symposium*, SEC. USENIX Association.

Wijesekera, P., Baokar, A., Tsai, L., Reardon, J., Egelman, S., Wagner, D., and Beznosov, K. (2017). The feasibility of dynamically granted permissions: Aligning mobile privacy with user preferences. In *Proceedings of the 38th IEEE Symposium on Security and Privacy*. IEEE.

Woolf, N. (2016). DDoS attack that disrupted internet was largest of its kind in history, experts say. Available from: `https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet`.

Xu, R., Saïdi, H., and Anderson, R. (2012). Aurasium: Practical policy enforcement for android applications. In *Proceedings of the 21st USENIX Conference on Security Symposium*, Security. USENIX Association.

Yee, K.-P. (2002). User interaction design for secure systems. In *International Conference on Information and Communications Security*, pages 278–290. Springer.

Yeo, J., Kotz, D., and Henderson, T. (2006a). CRAWDAD: A Community Resource for Archiving Wireless Data at Dartmouth. *ACM SIGCOMM Computer Communication Review*, 36(2):21–22.

Yeo, J., Kotz, D., and Henderson, T. (2006b). CRAWDAD: a community resource for archiving wireless data at Dartmouth. *ACM SIGCOMM Computer Communication Review*, 36(2):21–22.

Zhang, Y., Yang, M., Xu, B., Yang, Z., Gu, G., Ning, P., Wang, X. S., and Zang, B. (2013). Vetting undesirable behaviors in android apps with permission use analysis. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS. ACM.

Zhang, Z. (2006). Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. *Communications Surveys Tutorials, IEEE*, 8(1):24–37.

Zhu, H., Xiong, H., Ge, Y., and Chen, E. (2014). Mobile app recommendations with security and privacy awareness. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD. ACM.