Chapter 10
Privacy in Mobile Devices

Rinku Dewri and Ramakrishna Thurimella

10.1 Introduction

Personal computing has entered a new era with the wide adoption of mobile devices. Affordability of devices, reduction of implementation costs, and the possibility of novel business paradigms have driven the public acceptance of mobile technology at a scale much larger than desktop computing. The world's population is now close to having at least one mobile device per individual. Current generation mobile devices offer high-end computing platforms, connectivity to the Internet, and a range of applications, including but not limited to, local search, social networking, content sharing, entertainment, navigation, office suites, and games.

Consequently, this proliferation has also put the fundamental right to privacy of an individual at risk. Privacy is loosely defined as a "personally" assessed restriction on when and to whom an individual's data is deemed appropriate for disclosure. As user activities on a mobile device grow, so does the extent of personal information left behind in the device, and the inferences one can derive about the lifestyle, choices, and preferences of the user. Catering to privacy becomes difficult when the available data start to channel out to entities that are outside the trust boundaries of the user.

In the early days, privacy of mobile technology subscribers was regulated by federal legislations such as the Electronic Communications Privacy Act in the United States and the Data Protection Directive within the European Union. Network operators are bounded in terms of what data can be retained and how it can be used. The data itself involved records of calls and text messages. The same

R. Dewri (✉) · R. Thurimella
University of Denver, Denver, CO 80210, USA
e-mail: rdewri@cs.du.edu

R. Thurimella
e-mail: ramki@cs.du.edu

S. Zeadally and M. Badra (eds.), *Privacy in a Digital, Networked World*,
Computer Communications and Networks,
DOI 10.1007/978-3-319-08470-1_10

standards also applied when communication modes moved from the telephone network to the Internet. However, the rise of the Internet significantly increased the type and amount of data flowing through an internet service provider. Ideally, any user data flowing through the service providers are open to viewing, given that most is unencrypted. This is already concerning from a privacy standpoint; the only consolation is that the number of internet service providers are limited, and they can be monitored for illicit activities.

The use of mobile devices for personal communications, and subsequently for almost all types of data transactions, introduced the next level of privacy problems. Unfortunately, the adoption outpaced the law makers. Businesses and freelance developers flooded the market with applications never thought possible, and started engaging in massive data collection efforts about what users do on the mobile Internet, what they prefer, and what their tendencies are. The sheer number of service providers exploded, and innovative techniques came into light to use personal data collected at different sites to enhance the services. Data collection became more of a regular business practice. Explicit monitoring of what a provider is doing with the users data is no longer possible. Fair information practice principles had to be ruled out instead. These principles asked for a more transparent system where user consent should be obtained prior to data collection, and appropriate notices are given to the user about what is collected, how it will be put to use, who are the potential recipients, the method of collection, and how the data will be retained. The enforcement of these principles is mostly self-regulatory. This evolution of how personal data is handled has raised the question about who owns such data, and whether the individual is really in control any more.

The industry often does not view personal data collection as privacy breaching, as long as it is collected and used in a way such that the user cannot be identified from the data. This probably created the most common phrase one can read in a privacy policy—"data is collected and shared in an anonymized form." Perhaps this well-popularized practice prompted people to share more! The past decade saw a significant number of demonstrations to prove the loophole in this privacy blanket. Large fractions of the population can be identified by a combination of their gender, date of birth, and place of stay [15]. People may enter locations, interests, affiliations, in search queries, which make them unique in an anonymized web search database [3]. Knowing the ratings assigned to eight movies is sufficient to identify an individual, even when there is a two week error in obtaining the dates of the ratings [29]. Half of the individuals in the US population can be uniquely determined if their home and work locations are known at the level of a census block [16]. In GPS logs, people can be identified based on the last destination of the day and the most populated cluster of points [21, 25]. Individuals can also be identified by their social network structure [30], or their family tree [27].

New forms of inference possibilities are identified on a regular basis. This led to a significant push in educating the population about privacy in an electronic society, generating guidelines for the common application developer to be privacy sensitive, and a whole line of work in privacy-preserving design of application architectures. This chapter is a short review of the attempts that are under to retain the privacy of

users constantly interacting with mobile devices for most daily activities. Given its breadth, we are not including the legal framework for privacy protection in this review. We begin with a background in mobile devices and their associated technologies in Sect. 10.2. In Sect. 10.3, we highlight the privacy issues surrounding the use of a mobile device. Section 10.4 discusses the solutions put forth to mitigate the emerging privacy concerns. We end the chapter in Sects. 10.5 and 10.6 with a discussion on the challenges we are faced with in making a mobile device a more privacy sensitive platform, and the opportunities they provide us.

10.2 Background

Mobile devices have been defined in multiple ways since their inception. The most basic of these is that of a small, handheld device capable of receiving and transmitting voice calls over a long range wireless medium. These basic wireless communication systems became well-known as "walkie-talkies," once more and more capabilities became a standard in generic mobile devices. Features such as the ability to perform rudimentary computing, user interactions through a display and a keyboard, and execute common business and entertainment applications, are extensions of what the device can do, but soon became synonymous with what a mobile device means. However, the features that are typical in current generation devices will change and be extended, thereby leaving us little room to succinctly lay down the defining characteristics of a mobile device.

10.2.1 Mobile Devices

Mobile device characteristics are constantly changing. Features become obsolete even before a large fraction of the population gets a chance to use them. Therefore, for the purpose of this chapter, we simply adopt characteristic hardware and software features that dominate the current mobile device market. These features have been outlined by National Institute of Standards and Technology of the United States Department of Commerce [36].

A mobile device is defined as a collection of the following hardware and software characteristics.

- A small form factor device with a display and a real/virtual keyboard.
- A microprocessor that executes instructions on data stored in memory, and has lower power requirements than a typical desktop processor.
- Access to a Wi-Fi and/or cellular network that provides connection to larger network infrastructures such as the Internet and the telephone network; modern device capabilities often include personal network interfaces such as Bluetooth and NFC (near-field communication), as well as interfaces to access satellite navigation systems such as the GPS.

- A local non-removable data storage medium, often in the form of EEPROM (electrically erasable programmable read-only memory) flash memory.
- An operating system similar to that of a desktop/laptop environment, however with a comparatively lower resource requirement.
- A system to deliver, manage and execute applications.

A cellular network is composed of roughly hexagonal cells, each covering a small area, a low power transceiver and a base station in each cell, and an underlying telephone network between the base stations. Each cell can serve a fixed number of users depending on the available number of channels. A control center, called the Mobile Telephone Switching Office (MTSO), keeps track of the cell where a user is located, and aids the transfer of a call from one cell's base station to that of another. The same mechanism helps connect a mobile device to other network infrastructures such as the Internet. Depending on the size of the cell and the number of base stations serving it, radio signal strengths can be weak or strong. Digital communication technology have made it possible to serve more users in a cell, have more reliable communication, higher data transmission rates, broader bandwidth, and provide more secure communication channels.

10.2.2 Device Types

Mobile devices started off as two-way communication devices. Compared to the bulky DynaTAC from Motorola in 1983, modern mobile devices weigh just over 100 grams, and are constantly being enhanced for portability. Early generation devices consisted of a basic dial-pad and LED display. As mobile devices became popular amongst the masses, major players like Nokia, Sony Ericsson and Motorola flooded the market with newer, smaller, and lighter versions of their devices. Nonetheless, for almost a decade, mobile devices were nothing but an embedded system designed for a very specific task, voice and text communication. With reductions in the cost of microprocessor technology, the modern notion of a smartphone was introduced by IBM in 1994, with capabilities such as fax and email, and employing x86-compatible microprocessors. Since then, the hardware in mobile devices have seen an exponential growth, and vendors frequently provide upgrades to the state-of-the-art computing power and possibilities of a mobile device. Current mobile devices boast of quad-core giga-hertz level microprocessors comparable to desktop systems 5 years ago, embedded graphics processors, high-definition touch screen displays, multi-sensor interfaces, extended battery life, and all of these wrapped in the thinnest and lightest possible casings.

The software applications available in a mobile device have seen equally impressive advances. Popularly known as a "personal digital assistant" (PDA), mobile devices stopped being simple voice communication devices and started offering other capabilities such as text messaging, email composition, personal information management, and audio/video playback and recording. Palm's Palm OS, Nokia's Symbian OS, and Microsoft's Windows CE were some of the

forerunners in converting the mobile communication device to today's smartphone. These systems ported the notion of operating systems prevalent in desktop/laptop environments to the mobile environment, which were later replaced by more dominant players such as the Apple iOS and Google Android. Software applications that can run within these mobile operating systems are as diverse as office productivity suites, entertainment applications to watch streaming content, local search using GPS information, road navigation, networking with like-interest individuals, sharing content, playing games, and almost any other typical computing need we can satisfy using a desktop system. The combination of fast processor technology, wide adoption of mobile devices, and the surrounding business possibilities, have made it possible to develop an engaging computing device that has reached more individuals that any other technology.

10.2.3 Software Frameworks for Controlled Data Access

The complexity of the software system within a mobile device has seen tremendous growth in a short span of time. As our dependence on these systems grow, so does the necessity to carefully protect the personal data gathered during execution of the system. Almost all major mobile operating system vendors today realize this requirement, and employ one or more techniques to prevent software applications from arbitrarily accessing each others data and resources. Further, obtaining explicit user consent is crucial before any application is granted access to potentially private data.

10.2.3.1 Android

The Android platform is based on the Linux kernel, which is fundamentally based on separation of resources and data between different users. This user-based permissions model is also key to "sandboxing" an application's resources from that of others. Each application in Android is executed as if it is run by a different user. Since the user-based permission model, in the default form, does not allow interaction between processes originating from different users, an attempt by an application to read data belonging to another application will be prevented by the Android kernel. Applications under this model also have limited access to the operating system; in fact, operating system libraries, runtime environments, and application managers are also sandboxed in Android.

Every Android application has an associated manifest file (AndoridManifest.xml). An application developer uses the manifest file to inform Android about important application components, dependent libraries, and permissions required by the application to function, among others. As with any XML document, the manifest file is a collection of elements. For example, by using the uses − configuration element, an application can state its requirement for a certain

type of keyboard, navigation device, or touch screen. An application can request for permissions such as access to the Internet, read location data, clear caches, read contacts data, send SMS, update security settings, and others, through the uses – permission element. Each permission has an associated protection level, and the Android system seeks user confirmation for higher-risk ones, during installation. Permissions given to installed applications can also be viewed by accessing the application's information in the system settings. However, the recent versions of Android are closed to any user manipulation of permissions already granted during installation. Also, all permissions requested by an application must be granted by the user for Android to install and execute the application.

10.2.3.2 iOS

The Apple iOS system provides application sandboxing by placing each application in its own home directory, and then allowing the application to read and write to files and subdirectories only under the home directory. Several key directories such as Documents , Library and tmp are created by the installer, and these standard directories form the view of the file system for the application. The iOS system uses access control lists (ACLs) to provide such a contained view to the applications.

Unlike in Android, permissions in the iOS system are granted as they are requested by the application. However, the set of features that require explicit user consent is limited to location services, contacts data, calendars, reminders, photos, bluetooth sharing and microphone. For these features, iOS provides a relatively straightforward control to enable and disable an application's access to the feature.

10.2.3.3 Windows Phone

Windows Phone applications run in a strictly isolated environment, with a structured method to communicate with other features. Data storage is also isolated for the applications. The executing environment also provides a minimal set of capabilities to the application. Similar to an Android manifest file, Windows Phone applications have a package manifest file containing details about the capabilities that the application requires to function. Software capabilities can include access to contacts data, to the camera, to location services, and to Wallet payment instruments, among others. Users are notified about the requested software capabilities during installation of the application from the Windows Phone Store.

10.3 Privacy Issues

Privacy, by definition, is a subjective matter. A technology that can potentially encroach on the privacy of individuals is often deployed with the argument that individual consumers have the option of rejecting the technology if they decide. It

should be noted that the privacy issues surrounding a particular technology is not common knowledge, and without a detailed understanding, they can easily pass as minimal disclosures one must make for the technology to work. This specially holds true in software systems where interconnected components can transfer data between each other using highly technical methods, and vulnerabilities can help adversaries to evade securities protecting the data. Nonetheless, it is true that privacy is important only to the extent that an individual wishes to keep the right to control one's information. A 2013 consumer mobile data privacy study reveals that, while 76 % of users believe that they are ultimately responsible for their privacy, only 35 % read the privacy policy [38]. It is worth noting that less than 20 % of users are willing to share information such as date of birth, web surfing behavior, precise location, phone number, home address, photos/videos and list of contacts. Ironically, the most popular mobile applications constantly collect such forms of information.

In the following sections, we will elaborate upon the type of (personal) data that may be collected by a mobile application, and the methods by which this data may leak to third-parties. We will not attempt to highlight a certain form of data collection as a privacy issue, but will leave the inference of such a conclusion to the subjective discretion of the individual.

10.3.1 Private Surveillance

Public surveillance, specially using CCTV cameras, has always created privacy uproars in democratic societies. Such systems have been argued to be susceptible to abuse, and instruments for discrimination. Nonetheless, from a national security standpoint, they serve as a valuable tool to monitor high-traffic public places. Moreover, since only publicly visible activities are monitored in public surveillance, the threat to privacy is often viewed as minimal. On the other hand, constant surveillance of the private life of individuals is strongly opposed, and has till now been regulated by legislations. An enormous amount of resources will also be required to manually monitor the daily activities of even a small fraction of the population. Notably, the cost of privately surveilling large sections of the population has drastically come down due to the growing attachment between technology and the society.

Our daily activities in the physical world are getting more and more integrated with the digital world. We are constantly leaving a trail of digital breadcrumbs throughout the day, which if brought together, can produce a picture of our activities (public and private) with an alarmingly high accuracy. Further, the trails can be constructed on demand, for any period of time one seeks, and at a fraction of the cost of manual surveillance.

Let us explore what information may be left behind by an average technology-savvy individual on a typical day. We refer to this individual as Bob. All tracking activities stated here are very real, as has been evidenced in

publications in academic journals, trade publications, news articles, and business privacy policies. From the smartphone in his pocket and the computer on his desk, to his job, to the purchases he makes and the food he eats, just about everything Bob does leaves some sort of digital trace behind. Bob's smartphone is monitoring his location at all times in order to improve the quality of his services. This data can potentially be shared with a wide variety of applications, websites, and advertisers. Like many people, Bob uses an online email provider, such as Gmail or Yahoo! Mail. All emails he sends and receives are stored by the email providers, along with other information such as the search queries he makes on his mail. As Bob drives to work, his car's GPS constantly triangulates his location. This data may be transmitted to a central host to allow the GPS to give directions. Bob also uses an electronic pass to make going through toll booths easier on him. However, geolocation data on these passes is read all through cities in order to assess traffic congestions. Speed cameras along his way also read his license plate, irrespective of whether he was speeding or not.

Like many modern professionals, Bob uses LinkedIn to make connections in the business world. It is likely that a quick perusal of his profile will reveal his entire work history, the people he works with and has worked with, his educational background, and more. Almost everything that Bob does at work is tracked, whether that be for tax purposes, payroll processing, to protect the company from illegal activities performed with company equipment, or simply to monitor behavior and assess the performance of employees. All Internet traffic is scanned, key strokes and mouse clicks are logged, calls are tapped, and email conversations are monitored by management and IT departments. Bob's employer is immune to privacy lawsuits in this regard, as long as it can be established that such monitoring is for the interest of the underlying business.

When it is time for lunch, Bob uses Grubhub, an online platform that lets him order food from any restaurant in the area. Such portals can indefinitely store almost every piece of information that passes through their system, including name, delivery address, phone number, food habits, and payment information. The information may not be disclosed to marketers and third-parties, but at the same time, is not proactively deleted. A restaurant's digital security may also not be a high priority. Physical eateries also record all reservations, orders, billing, and customer mannerisms and preferences. With the objective to enhance customer experience, this data can be quite comprehensive and accessible to most employees of the restaurant.

On his way back home, Bob stops by a store to replenish his supplies. His retailer membership card allows him to get discounts and coupons. The same card also lets the retailer profile Bob, his buying habits, and tendencies. The retailer can also track Bob inside the store through his smartphone, and later send out special coupons for items that Bob spend time looking but did not buy. Once back at home, Bob spends some time on the Internet. Most sites he visits use some form of user tracking, often in the form of beacons, cookies and browser fingerprinting. Bob's expenditure habits, on the Internet or in physical stores, are all well summarized in his credit report. Other businesses can also look at this data with little forewarning.

After a tiring day, Bob decides to play a few rounds of Madden on his XBox. Data on practically everything Bob does on the XBox Live service is collected, including the games he plays, the music and videos he purchases and listens/watches, and even samples of his voice if he uses the voice recognition service. Usage data is also collected when Bob interacts with other services such as iTunes, Amazon MP3, Spotify and the likes. Right before ending the day, Bob reads a book on his Kindle, which can monitor his reading habits, type of books he likes, and even offer special deals. Throughout the day, Bob also did some voluntary data sharing through status changes, comments, and posts in online social networks.

Clearly, an average individual is unlikely to be aware of all small data bits that is being tracked. A single technology, such as the mobile device, is also not responsible for making this possible; although, much of this has become possible due to the advances in one or another form of mobile technology, and the increasing usage of mobile devices. A full assimilation of the collected information is perhaps not feasible for a single commercial entity, but the push in Big Data mining activities do leave the possibility open.

10.3.2 Data Collection

Mobile devices have changed how people access information on the Internet. Typical desktop usages are interactions that a user initiates to find the desired information, often well in advance. With the vast proliferation of mobile devices, information access has become more reactive, where a user performs an action at the time when it is needed. Therefore, locality and usability of the served information has become an important feature of mobile applications. This also implies that many applications require metadata about the user in order to serve information that is relevant in the current context of the user. A mobile device being a personal companion, obtaining data corresponding to the specific user in question is also not difficult. This driving force has resulted in massive data collection exercises about users, and seems to have blurred the line between what is necessary for the application and what is simply extraneous.

10.3.2.1 Implicit Collection

Most of the basic voice and messaging features available in a mobile device are exclusive to mobile devices. It is only when a device is used to exchange information with an application server on the Internet, a mobile device's functionality is more like a portable computer. For a mobile device to be able to provide call, text/picture messaging, and Internet services, certain features such as call details, cell tower in use, text message details, and IP session and destination information are visible to the mobile carrier. Such data has huge marketing value. Since the

phone number is a pseudo-identifier of the person/family, call records can reveal who we call, who calls us, as well as the locations from where the calls are made or received. They can also reveal the identities of our friends, the doctor we visit often, our vacation plans, the financial institutions we deal with, and even our favorite handyman. In the United States, voice, messaging and IP data can be retained from 3 days to 7 years depending on the carrier and type of information [39]. Long-term retention of such rich data poses one of the most severe threats to privacy.

10.3.2.2 Pertinent Collection

Mobile operating systems make it possible for an application to access software, hardware and user-specific features for proper functioning. For example, a typical social networking application in Android can specify the following elements in its manifest file.

```
<uses-permission android.name="android.permission.ACCESS_FINE_LOCATION">
<uses-permission android.name="android.permission.ACCESS_COARSE_LOCATION">
<uses-permission android.name="android.permission.INTERNET">
<uses-permission android.name="android.permission.CAMERA">
<uses-permission android.name="android.permission.READ_PROFILE">
<uses-permission android.name="android.permission.RECORD_AUDIO">
```

Once granted, the permissions enable the application to access the GPS module, location approximated based on the radio signal strength, the Internet, the camera, the user's profile stored in the device, and record audio. This data is also pertinent in the context of the application, without which the full potential of the application cannot be realized. However, once an application receives access to the requested data, the operating system loses control of how the data is used, and does not prevent the application from storing it or forwarding it to its servers.

When reviewing the pertinence of a data request, a question that we often fail to ask is whether the granularity of the requested information is critical for the application to offer its services. We exemplify this observation using the GPS access permission that a location-based application is likely to request. At first glance, it is unlikely that a location-based application can function without accessing the location feature. However, will the information generated by the application change substantially if accurate location data is not provided? For example, a weather application can retain its accuracy even by using an area code. For a non-trivial example, consider a local search application that helps the user find points of interest in the vicinity of the user's location. Although location is an important component in obtaining relevant results, local search services utilize a number of other factors (popularity, cost, references, etc.) to determine relevant results. Search results are also unlikely to change instantly due to movement of the

user. Even in dense cities such as Los Angeles and New York, search results for the ten nearest Starbucks locations can stay invariant for areas as large as 87 km^2. [6].

Similar to the implicit data collected by network carriers, application specific data collected by application providers also have huge marketing value. Unlike the case of call records, whose sharing is heavily governed by federal legislations, data collected by application providers is very likely to be used for business gain. Potential (mis)uses of the data is invisible to the user, thereby opening up privacy concerns.

10.3.2.3 Extraneous Collection

Extraneous data collection refers to the activity where an application accesses information in the device that cannot be rationally attributed to the functioning of the application. A flashlight application accessing the user location, the contact list, the calendar, or media files is an example of extraneous collection. Clearly, the motivation here is not service quality, but creating a rich database of user data. While in some mobile operating systems, requests for such access can be denied at the time of access and still keep the application running, in others, the application must be granted all requested permissions prior to its execution. The latter design forces the user to comply with extraneous data collection requests, effectively foregoing privacy expectations.

Another potential issue arises when permission requests are added during the upgrade of an application. Early versions of an application are restricted to pertinent data collection; however, extraneous permissions may be added as upgrades are released. A user may be tempted to permit the extraneous collection simply to avoid the hassle of learning a new application. Such practices can lead to a user weighing down on privacy over time. Extraneous data collection can be performed by the mobile operating system vendors as well.

10.3.2.4 Web Tracking

In addition to data collection through mobile applications, user behavior tracking on the mobile web is an increasingly growing trend. Behavior tracking in the conventional Web is a well-known phenomena [24]. As web services grow in diversity, and vendors start offering entire product suites, the ability to monitor user viewing habits across the range of services is unquestionably an advantage. Various forms of web technologies are used for this purpose. The most common of these are cookies, which are small pieces of text data stored in the user device by a browser at the request of the web server. The data stored in a cookie can be a simple identifier (such as an IP address) of the user, along with details on the actions performed by the user at the web site. When the user visits the web site, or a companion site, at a later time, all previously created cookies can be retrieved from the user device and prior actions of the user can be extracted from them. Cookies can be disabled and

deleted by the browser upon request by the user. An alternative is to use a web beacon, which can be a tiny (1 pixel by 1 pixel) image placed on web sites across different services. When a user visits the different web sites, the browser will request this particular image, and pass along metadata about the user as part of the request. Although not as flexible as cookies, web beacons can be used to track a user's browsing activity page by page, site by site.

All of the user-tracking technologies in the conventional Web can also be applied in the mobile Web. The mobile ecosystem is host to a number of applications that engage in data collection, often much personal than that is currently possible through a web page. When such data also become available to vendors of the web services, supposedly anonymous activities on the web can be linked with user profiles (and identities). A mobile browser can itself have access to multiple sensor data from the device (not possible in the conventional setting), including location, camera, microphone, and personal data such as calendars and contact lists. Efforts are ongoing (see Mozilla WebAPI[1]) to design Web APIs that allow web applications to access device hardware and other data stored in the device at a much granular level.

10.3.3 Data Leaks

Users of mobile devices are often made aware of the type of personal data collected by an application. The sandboxed model of data access in mobile operating systems makes it difficult for an application to bypass a user-consent dialog when accessing data that is deemed sensitive by the system designers. However, the data, once collected by an application, may be shared with (or leaked to) parties that are not directly authorized by the user.

10.3.3.1 Business Policy

The most common form of data leak is in fact a totally legal, user-authorized form of data sharing. A careful scrutiny of an application's privacy policy can reveal that the data collected from the device is not only used to improve the services of the application, but can also be shared with entities that support the execution of the different services pertaining to the application. For example, a section of the Facebook data use policy reads as follows.

> We give your information to the people and companies that help us provide, understand and improve the services we offer. For example, we may use outside vendors to help host our website, serve photos and videos, process payments, analyze data, conduct and publish research, measure the effectiveness of ads, or provide search results. In some cases we provide the service jointly with another company, such as the Facebook Marketplace. In all of these

[1] https://developer.mozilla.org/en-US/docs/WebAPI.

cases our partners must agree to only use your information consistent with the agreement we enter into with them, as well as this Data Use Policy.

Any user using an application agrees to its terms and conditions, in other words, grants permission to the application provider to share the collected data. In 2010, the *Wall Street Journal* published an article on the data being collected by some of the most popular applications and games in the iOS and Android platforms, and the subsequent automatic sharing of the data with other vendors [37]. The study used a *man-in-the-middle attack*[2] to observe which applications collected features such as username, password, contacts, age, gender, location, phone ID and phone number of the user, and to which other vendors this data is sent. A standard issue here is that details on the identity of the third-party service providers are often missing in the policy declaration. Hence, it is difficult to evaluate if the security infrastructure of a third-party provider is as good as that of the trusted application provider. It is argued that the anonymity of individuals is retained by sanitizing the shared data of personal identifiers such as names and addresses. However, quasi-identifiers such as age, gender and zip code are often sufficient to identify a person by cross-referencing with other databases [3, 15, 29, 30]. In addition, given that a mobile application can be developed and deployed by any developer, most of them do not have a privacy policy.

10.3.3.2 Permission Leaks

A second potential for data leak can originate from the functional design of the software application. An ill-designed application that has permission to access personal data content may erroneously leak the data to other applications. For example, an Android application can use the services of a map application to display the current location of the user. The application can request the service with two simple lines of code.

```
Intent intent = new Intent(Intent.ACTION_VIEW,
                Uri.parse(geo : 0, 0?q = 34.99, −106.61(Current Location)));
startActivity(intent);
```

While potentially harmless, this action can lead to a permissions leak, if the user only trusted this application with the precise location co-ordinates. The mobile operating system will not intervene in this case, since the map application is not attempting to obtain the data from the system. Mobile applications also implement handlers for many of the services they may provide to other applications. For example, an application with access to the contacts data in the device can serve it to

[2]An attack where an eavesdropper intercepts messages between two parties, and relays them either after simple observation or after modification, without detection by either party.

other applications. This can be a legitimate and useful action, but handlers that serve requests to such permission-protected data may have been left open for access by any application. Ideally, any interface that provides access to permission-protected data should be subjected to the same permissions as that of the data. In our example, if the user's permission is required to access the contacts list, then permissions should be set in the manifest file to protect access to any public method that accesses the contacts data.

10.3.3.3 Leaks via Embedded Content

Mobile applications can embed content of different types in their user interface. A free version of an application, for example, may embed advertisements from mobile marketers, or a mobile web page can be composed of elements collected from multiple sources. Consider a user interacting with an online shopping application. The user shares some personal information such as age, gender, and location, with the application in order to customize the search results. The shopping application returns useful results to the user, along with a pertinent advertisement from an ad-network. This observation is also typical in web browsing sessions. The HTML pages returned by such web servers often include embedded requests that encourage the browser to ask for advertisements and images from third parties.

Many of the current web-based applications expose web APIs that allow any other application to request content in the form of formatted objects by using standard HTTP protocols. Web APIs also make it easy for the calling application to pass parameter values to the web application. For example, once connected with the googleapis.com server, an application can request the translation of the word "Hello" to Spanish using the following request.

$$\text{GET /language/translate/v2?q} = \text{Hello\&target} = \text{spanish HTTP/1.1}$$

A mobile application can use the web APIs exposed by an ad-network to retrieve advertisements (or other content) and in that process, leak private information, if user-specific data is included as parameter values in the request. When an online shopping application generates a request such as

$$\text{GET /adj/...product; age} = 30; \text{gnd} = 1; \text{zip} = 12345; \ldots \text{HTTP/1.1},$$

it leaks demographic information about the user to the ad-network, and potentially in unencrypted form. Request parameters can also include device identifiers, account preferences, and other forms of user data generated as part of the interaction process. While a single occurrence of such leaks is probably not concerning, if a single entity provides such services to a large and diverse set of applications, then tracking a user's activities across different types of applications becomes possible.

10.3.4 Data Interception

Signal interception, or eavesdropping in popular terms, is a threat to any form of private data exchange, and has raised concerns in technologies such as phone calls, text messaging, and emails. All data originating from a mobile device finally has to travel to the base station, from the base station to the MTSO, from one MTSO to another, and then from an MTSO to the public telephone network or the Internet infrastructure. All points in the flow are vulnerable to a man-in-the-middle attack, where an entity can intercept the signal, analyze it, block it, and in some cases modify it.

The 3G security specification (3GPP TS 33.102 V12.0.0 2014-03) requires that the mobile network and the device compare their ciphering capabilities to establish an encryption algorithm for use during communications. The specification gives the option to agree on unencrypted communications if the device and network agrees so. Because of the requirement to be backward compatible with the GSM security architecture (still predominantly in use), the ability to use unencrypted communications is still retained. With capabilities available for the network to choose an encryption algorithm, an adversary can set up a false base station in a localized area and force communications to be unencrypted. The false station can also continue to provide connections to the actual network by forwarding incoming content through a valid channel that it has established with a real base station. As a result, the adversary is able to observe all data originating from any device connected to the false base station. Also, since a mobile device automatically drops to a 2G network when a 3G network is unavailable, possibilities exist to force devices to use the 2G network by jamming the 3G frequencies. It does take considerable expertise to set up such a system; however, not outside the reach of law enforcement organizations and individuals with the required skills.

The most vulnerable portion in the communication path is when the data reaches the core network of the provider after the radio-based network. The internal infrastructure of telecommunication networks is often not encrypted. Starting from the cell base station site, to the telephone switching network, data flows through physical mediums such as wires, switches, servers and other mediums. Any raw data (basic voice/messaging and unencrypted application content) traveling through these nodes is susceptible to passive monitoring by an employee, a hacker who has compromised an intermediate server, or an agency who is lawfully allowed to intercept user communications.

10.4 Privacy Solutions

Multiple solutions to the potential privacy breaches that can emanate from data collection and tracking methods have been proposed. We will explore them under three headings (a) developer guidelines: best practices that an application developer

should follow to respect and protect the user's personal data (b) user options: the tools available and proposed to enable self-regulation and (c) application architectures: novel technical solutions to execute common types of mobile applications, yet without requiring personal data disclosures. Solutions discussed under developer guidelines correspond to some of the issues mentioned in Sect. 10.3.2, and those under user options correspond to the issues in Sects. 10.3.3 and 10.3.4. The efficacy of all these techniques remains to be tested. Potential privacy issues in mobile devices are identified on a regular basis; although, solutions for their resolution are designed and enforced at a much slower pace. Given this widening gap, it is worth pondering whether personal privacy is indeed on its way to becoming a "thing of the past."

10.4.1 Developer Guidelines

Developers of mobile applications are the first party responsible for protection of user data. Every mobile operating system provides security features to restrict access to personal data, but their proper use can only be guaranteed by the developers. As such, vendors that provide the application development frameworks, such as Google[3] and Apple,[4] have guidelines for building trustworthy applications.

A privacy-sensitive application is transparent about the type of data collected from the device and how it is subsequently used. A clearly written privacy policy can go a long way in earning the trust of users, and getting the requisite access to make the application function. It has also become important to educate users why a certain piece of data is important for the application. Extraneous data collection should be avoided as they do not serve any useful purpose for the user. Collection of identifiers such as IP addresses and unique device identifiers also put the user at privacy risks. Similarly, if third-party sharing will be performed, a clear explanation should be provided about what will be shared, why is it necessary, and who it will be shared with. More details are better, but in a concise manner.

Whenever possible, applications should be developed assuming limited access to user data. Therefore, developers should include alternative control paths for cases when a user is not willing to share a certain piece of data. This enables the user to assess if the limited functionality of the application is still acceptable, and also enhances the user's trust in the application. Requesting access to sensitive data at the time when it is required also helps the user to understand the context in which it will be applied. On the other hand, asking for a set of permissions prior to usage only raises suspicion. It is common in open-source environments to reuse code written by other developers; it is critical to understand if the borrowed code follows the data access rules set forth by the developer.

---

[3] http://android-developers.blogspot.com/2010/08/best-practices-for-handling-android.html.

[4] https://developer.apple.com/library/ios/documentation/iphone/conceptual/iphoneosprogrammingguide/AppDesignBasics/AppDesignBasics.html.

As far as possible, data logging and data storage on offsite servers should be avoided. Storage of data opens up possibilities for misuse later, some of which may directly impact the privacy expectations of the user. If data storage is crucial for the application, developers should determine the largest possible granularity of the information that the application requires–GPS location or postal code, phone number or area code, user details or username–and employ encryption techniques to ensure that the information is not visible on its route to the server.

Privacy oversight bodies also suggest that mobile application platforms (operating systems) should set requirements for developers to follow before their applications can be approved for distribution [10]. For example, dialog boxes used to obtain access permissions can be designed to obtain a one-time or a persistent consent. Platforms should also provide an informative dashboard where users can review existing permissions, logs of when applications accessed their data, and modify/revoke permissions if necessary. There should also be ample notification given to the user when personal data is being accessed, preferably through the use of icons in familiar locations of the user interface. Custom-built firmwares such as the TaintDroid [9] do make such notifications possible. However, these tools are mostly research prototypes, difficult to install for a novice user, and are not maintained by the mobile operating system vendors. As such, their availability and adoption in the long run are questionable.

10.4.2 User Options

User controls are available in most mobile platforms to help regulate what an application can learn about the user from the device. A platform such as Android requires a user to provide explicit permission to access all requested features before an application can be installed, while platforms such as the iOS ask for the permission when the application tries to access the sensitive data. The two models are different in terms of when they engage the user to make a decision, one giving the option to the user to review all information that the application will attempt to access, and the other allowing the user to potentially understand how the information will be used. Despite the differences, it must be noted that the user only controls access to data features that the platform designers have marked as sensitive. An application may be collecting other forms of personal data that do not require user permission. Some applications may assume that the user is always willing to share such data (default opt-in), while others assume that permissions are required for all kinds of access (default opt-out).

Besides the ability to review permissions during installation or just-in-time, users also have the option of changing permissions in certain platforms. The iOS platform provides a privacy dashboard that allows the user to enable or disable permissions of an application for features such as location, contacts, calendars, and microphone. Other systems may only allow viewing of permissions, with little or no option for the user to revoke single permissions. It is important for users to explore what

privacy options are provided within the application itself. For example, a social networking application can provide controls to choose what category of other users (no one, friends, friends of friends, or everyone) can view the text, pictures, and videos posted by the user; a local search application can provide the option to use location approximations instead of precise GPS coordinates; in general, an application may require the user to tune the preferences so that it can perform within the privacy expectations of the user. By doing so, users may also be able to turn off extraneous data collection activities of the application, especially in platforms that do not allow permission modification. Research proposals also exist that analyze the application ecosystem in a mobile device to determine if permission leakage is possible during the execution of an application. Proof-of-concept tools such as the SCanDroid [13] can statically analyze if it is safe for an application to run with certain permissions, provided that data may flow in or out of other applications with different permission levels.

In order to prevent third-party tracking, a user may choose to disable cookies and scripts, or use the private browsing modes now provided in many browsers. However, disabling of cookies may also disable the functionality expected from certain web interactions, for example an online shopping experience. Technologies such as opt-out cookies and AdChoices notifications can help the user to proactively request a third-party to refrain from tracking, or reactively become aware of how a third-party may be tracking the user.

Opt-out cookies are similar to regular cookies, but they are placed by the user to be read by tracking websites. Trackers that honor such cookies always attempt to read opt-out cookies before storing and serving content corresponding to the users personal data. However, opt-out cookies have to be periodically renewed by the user, and may get removed while deleting other cookies. If an ad-network is instead enrolled in the AdChoices program, the user can learn more about the provider of an advertisement and also install opt-out cookies from provided links. Under such a program, the displayed ads embed a small text and graphical icon, clicking on which reveals details and opt-out links for the ad's source. Blocking tools are also available that consult public lists of third-party networks, and block content originating or addressed to such sites. Browser extensions are the most popular methods of executing such blocking tools.

Mobile operating systems have started deploying a unique user-specific identifier called an "advertising ID." An application can use it as a pseudonym for the user, although it is not as strongly tied to the user as a device identifier, and can be reset or disabled by the user. Both iOS and Android systems provide this feature so that developers can monetize their applications, and users have the control to protect their privacy by periodically breaking, or opting out of, tracking attempts carried by the application and associated third-parties. The World Wide Consortium (W3C) is also standardizing Do Not Track, which is a method that can be used by users to signal web sites about their tracking preferences. A simple implementation is by using an extra element in the HTTP header that browsers send while requesting web pages. For example, the presence of the header DNT : 1 can tell the web site that

the user does not want to be tracked. However, it cannot be guaranteed that the web site will honor the request.

Protection against eavesdropping is possible using encrypted communications at various points of the communication channel. Users can choose to install end-to-end encryption solutions to prevent eavesdropping on their communications. Applications such as Cellcrypt can encrypt voice communications and text messages between two parties running the application. Such applications do not use the typical channels used during voice calls, but instead use the data channel to send communications as encrypted data, and then decrypt them at the other end. Therefore, depending on the data coverage, calls may be noisy. Incorporation of encryption technologies within the mobile network itself can free this dependence on data coverage, and provide protection against outsiders in basic services such voice calls and text messaging. Similarly, although proposals based on mix-networks and pseudonyms are available to hide the cell of a mobile device from the carrier [11, 23], they lack any real world deployment.

10.4.3 Application Architectures

Privacy solutions discussed so far rely on the adoption of best practices by the developer, and on the user being responsible about evaluating and regulating access to private data. A third method is to employ technical solutions such that private data is not directly available to an application, albeit the application can still provide the services. As such, methods in these categories are crafted to take advantage of an application's specific data requirements, and do not necessarily generalize to other applications. We will consider five prominent application types in this discussion, namely local search, friend finder, online social networking, navigation and web browsing. The solutions presented require significant modifications to existing application architectures; this can be one of the reasons why their adoption in practice is slow or non-existent.

10.4.3.1 Local Search

In a mobile local search application, the user specifies a search term signifying the type of objects (e.g. cafe, pizza, gas station, etc.) of interest, and the application, with access to the location of the user, retrieves the top few objects that match the user's query. The selection of these top objects is based on multiple factors. For example, selection of result objects in the Google Places [32] application is driven by two factors, namely the distance of the objects from the user's location and their prominence. The prominence of a POI is derived from multiple subfactors such as reference counts, highest score of objects that refer to this object, number of user reviews, and the extent of services offered, among others. The private data to be protected here is the location of the user.

An extensively explored method is to use the services of a trusted third party. In such a method, it is assumed that the users trust a third party with their location data, which in turn communicates with the application provider to retrieve results on behalf of the user. Since the trusted party will have access to the locations of different users, it can issue local search requests by specifying a region, instead of precise GPS coordinates. The region can be formed such that, at any given point in time, it includes multiple users [14, 18]. Such a "cloaking" region can also be formed to satisfy other properties: the number of still-object counts must also be above a user-specified threshold [2], the query terms corresponding to users in a region should be diverse [5], or the profile of users in a region should not be similar [35]. It is usually difficult to determine who the trusted party should be, and also introduces a single point of failure in the system. A compromise of the trusted party can compromise the location data of multiple users. Most of these techniques apply in the context of local search. Technical solutions for private local search assume that distance is the only factor used to determine the relevance of objects; unfortunately, it is only one of many factors, because of which the solutions are not easily applicable in a real application. A few recent proposals have appeared where it is treated more practically, but demands more exploration [7].

10.4.3.2 Friend Finder

Friend finder applications notify users when their friends (users included so in the application) are in close proximity. Applications such as Foursquare, Facebook's Nearby Friend, Find My Friends!, and other social networking applications provide such a feature. Finding nearby friends privately is referred to as "private proximity testing" in the literature, where the objective is to determine if two users are geographically close to each other without revealing the location of either user to the other. Such a problem can be reduced to a "private equality testing" problem where the task is to determine if two values are equal, and no user will know the value supplied by the other, unless they are equal. To perform such a reduction, the geographic area is divided into three overlapping hexagonal grids, and the size of cells in a grid is set subjective to a distance threshold [31]. Using a few encrypted data exchanges, a private equality testing protocol will allow two users to determine if they are in the same cell. The three grids help the users determine proximity even if they are on different, but nearby, cells.

An alternative to using location in proximity detection is to use transient signals from the environment. Referred to as "location tags," these signals can be derived from electromagnetic technologies such as Bluetooth, WiFi, GPS, and GSM, and are collectively specific to a certain area [33]. A location tag can be viewed as a vector of these signals, e.g. WiFi SSIDs visible to the device. An application can compute the number of elements common in the location tags of two users, and signal proximity when the number exceeds a certain threshold. To prevent inference of the location of the user from the tag itself, "private set intersection" protocols can be used [20, 22].

10.4.3.3 Online Social Networking

Users of online social networking applications entrust the application with their personal preferences and posts, and not simply the data that can be gleaned from the device. The tools provided by the applications to control who can view the voluntarily shared content is often limited. In addition, their sharing with unknown third-party affiliates is concerning. The use of public key cryptography to protect the data is not scalable. Since users may want to share different content with different users, they will be required to keep multiple copies of the encrypted content, will be unable to easily share content with a group of users, or must have already exchanged cryptographic keys prior to sharing. A solution to this problem is to encrypt the content with respect to an "access structure" [1]. An access structure is simply a logical expression of attributes that define a certain group of users, e.g. (neighbor OR football fan). Any user with access to a decryption key with respect to this structure, and has the attributes that satisfy the logical expression, can then view the content. This form of encryption is known as attribute-based encryption [17]. The encrypted content resides with an untrusted third-party. Removal of a user from a group will require that other users of the group are given new secret keys.

The privacy issues surrounding online social networking applications are primarily due to the centralized collection of user data. Decentralized architectures can diffuse such concerns. In a trivial implementation, the user's device can be the data storage location, and other users can view shared data by querying the device. Clearly, this method can have availability issues. As an extension, the data can be replicated across other locations, either in encrypted or in clear form. These locations are chosen based on a trust index assigned by a user to other users [4]. The solution does involve a third-party who can help manage pseudo-identifiers for the users, and connect and find them.

10.4.3.4 Navigation

Phone-based navigation has gained wide popularity in mobile devices with the introduction of GPS capabilities and the Internet in the devices. Services such as Google Maps, and applications such as Waze are typical examples. The ability to use real time traffic information in route determination has significant advantages. Privacy issues emanate because the routing service must know the origin and destination to compute the best route, as well as know intermediate locations to dynamically update the route. In essence, the routing service can track the places visited by the user. Privacy preserving navigation is a relatively new area of research, and techniques are rare.

Assuming that the user's device has the capability to store the topology of the road network (in the form of a graph), the route can be computed directly on the device. However, dynamic information such as traffic congestions, road constructions, and accidents cannot be stored on the device. The problem then is to compute the shortest route between the origin and destination nodes in the graph, using cost

information from the service provider. However, the origin and destination nodes cannot be revealed to the service provider. The service provider can compute the shortest path for all pairs of possible origin and destination nodes, and store this data as a four column database–start node, end node, intermediate node, and a bit signifying if the intermediate node is in the shortest path from the start to the end node [40]. Private information retrieval methods will then allow the user to query the service provider if a certain neighboring node is in the shortest path. Such retrieval methods provide querying mechanisms such that the service provider will be oblivious to the row of data returned to the user [26]. Variants of the method include designing the database so that the user can privately retrieve the next hop with a lesser number of queries.

10.4.3.5 Web Browsing

Web-based services are becoming multi-party businesses, with advertisers, analytics services, social integration services, content providers, front-end services, and hosting platforms working together to create the ultimate customer experience [28]. Of course, data sharing can happen uncontrollably in such a model. Earlier, we discussed what options a user has to control this sharing. Several technical solutions have also been proposed to design third-party services that can benefit the user while preserving privacy. However, the coverage is mostly limited to online advertising.

Services such as online advertisement can be made private by employing anonymizing proxies [19]. The approach requires users to subscribe to advertisement networks via a proxy server. The proxy server collects non-sensitive broad demographics data from the users, assigns each user a unique identifier, and retrieves relevant ads from the advertisement network. The unique identifier can be used to track user activities. The proposal is similar to the concept of using advertising IDs in iOS and Android. The use of an anonymizing network such as TOR [8] is suitable in this context. User profiles are also created on the mobile device so that the ads retrieved from the proxy server can be further filtered in the device. Such profiles can also be released directly to the advertiser with permission from the user [12]. This model is similar to how browsers request permission to release the location (IP address) of the user when requested by a web site.

Ad auctions is another business paradigm that can be affected by Do Not Track compliant applications. In an ad auction, advertisers submit ads to an advertising network along with bids for their ad to be displayed to the user. The ad dealer (advertising network) attempts to serve ads (to users) that have higher bids on them, as well as a higher chance of being clicked. Calculation of the click chance requires information about the web page being browsed by the user, ad keywords, user search terms, and other characteristics describing the user's actions. If applications refrain from sharing such information with the advertisement network, ad auctions cannot be performed. More specifically, the ranking of ads is not possible for the ad network. However, assuming that the rank of an ad is a product of the bid and a

score derived from user profile information, the ranking can still be performed through the exchange of a few encrypted messages [34]. Opportunities exist for the ranking to be done in the device, at the ad network site, or at a third party site.

The architectures discussed here are representative solutions for the application types. Readers should use the citations provided to explore more related proposals.

10.5 Challenges and Opportunities

It has become clear through the above discussions that the usage of mobile devices can potentially put a user at privacy risk. There have also been extensive attempts to mitigate this risk. Nonetheless, privacy worries continue to grow in the mobile arena, and technology and law has not been able to control the growth in collection of private data. We highlight below some of the challenges that lie ahead of us, and the opportunities they provide for new research directions.

Opinions on privacy. Enforcing any solution to a privacy problem in a mass scale is a challenge. There is hardly any consensus on whether a technology is privacy breaching or not. Businesses see value in the collection of user data as it allows them to create services highly customized to the requirements of the user. The individual users are divided in opinion–some do not mind sharing their information, while others are skeptical about sharing even broad demographics. Given this broad set of perspectives, neither a purely restrictive nor a discretionary solution will be adopted in practice.

Disclosure with consent. It is understood that data leakage is a concern. However, no simple solution exists to curb the leakage. Privacy policies tend to throw a big net to cover almost any form of data, although much of it may not be necessary or is not collected from the user. However, it is equally important to understand that an application that seeks consent for every small piece of data it records will undoubtedly create usability issues, and will not be accepted by users. Current methods to engage the user in the permission process are limited to a select few types of data. The question remains open as to when and how often the user should be asked for permission to access personal data.

Granularity in disclosures. Current mobile systems have adopted a binary method of controlling disclosure of information. An option is available either to disclose information in the least granular form, or not disclose at all. Similarly, applications are developed to operate with the most precise version of a piece of data, or not use it at all. This binary design forces the user to choose between using an application as it is or disregarding it. There exists opportunities to design fine-grained controls for data disclosures, and applications that can gracefully degrade in services based on a chosen disclosure level.

Mobile hardware. Research on privacy-preserving architectures for modern mobile applications started a decade ago. Many of these architectures try to preserve privacy by significantly augmenting the server-side functionalities, or engaging an intermediate party to perform computationally expensive operations.

The motivation for doing so is justified, given that mobile devices have only recently become powerful enough to be considered full fledged computing platforms. The state-of-the-art in the processing capabilities of mobile devices was around 400 MHz in 2007 (e.g. the 32-bit RISC ARM in the first iPhone), which has grown exponentially to support multicore systems with clock speeds of up to 2.5 GHz in 2014 (e.g. the Snapdragon 801 on the HTC OnePlus One). These advances in the computing power of mobile devices are yet to be availed in novel privacy preserving architectures. They may also prove to improve the performance of otherwise slow methods. Engaging the mobile device also allows for a more scalable architecture, since the computations will be highly distributed.

Using existing infrastructures. A number of technical solutions do exist to execute some of the prominent type of mobile applications and services in a privacy preserving manner. However, they involve complex architectures and data exchanges that are not typical in existing implementations. Heavy investments have already been made to deploy these implementations, and large user databases have already been created through such implementations. It is therefore unlikely that existing infrastructures will be discarded in favor of user privacy. Privacy-preserving solutions will have better chances of adoption if they can be executed using already existing server-side components. It is also important that solutions stay close to real world design trends and assumptions. For example, while local search trends continue to rise, the wide array of location privacy solutions seem unusable since most of them solve a nearest-neighbor problem, instead of a top-neighbor problem. Similarly, many solutions are based on using encryption to privately match two individuals in an online social network; albeit the sophisticated key management components necessary to achieve it in practice are missing in existing applications.

Providing data transparency. Application developers often do not implement each and every component in their design. In open-source environments, it is common for a developer to borrow code written by other developers. Many vendors also supply software development kits (SDK) that provide core libraries, debuggers, emulators, and sample code in order to make app-development easier. APIs are also available to interface one application with another. However, any code that executes as part of an application also inherits the access permissions granted to the application. Therefore, even if the developer's code is not actively collecting any data, the developer cannot assure that the library/API calls in the application are also not performing it. This prevents the application from being transparent about its data collection attempts.

Eco-system diversity. Current applications let users perform more than just a single task. For example, a news application pulls articles of interest from different sources. It also allows the user to share an article via one of many social networking platforms. It displays advertisements for trade magazines and periodicals based on the articles often explored by the user. It can help a group of readers in a locality to get together in person to discuss a story they all have been following closely. In other words, a news reader application also has components like a social networking application and a local search application, and is also an advertising

platform. Given this diversity, will it be possible to enforce a privacy method that can cater to the requirements of all components? The technical solutions we discussed are in fact very specific to an application type! It is also unknown if different privacy solutions can work in conjunction to provide the necessary guarantees from all components of the application.

Privacy from carriers. Network operators routinely collect metadata on the traffic flowing through their networks. Ensuring the privacy of data collected at these operators is a difficult task, partly because the data must be visible to the carrier in order to route traffic, and partly because "lawful interception" rules require the operators to leave avenues open for interception. Interception technology has also become commonplace. Encrypted communications ought to be a standard in such environments, but lacks adoption in majority of applications.

Mobile privacy interface. Privacy disclosures often take the form of lengthy documents. Application vendors have shown effort in making them understandable by minimizing the amount of legal terminology. While this is certainly encouraging in desktop systems, it is not sufficient in a mobile interface. Mobile devices are not suitable for extensive reading, and the privacy policy should not become a long reading exercise for the user. Innovative designs are necessary to compress the most important parts of a mobile application's privacy policy, and present it using visualizations, instead of text. Standardizations for the format of a privacy policy are also required to facilitate fair comparisons. Besides the presentation of a privacy policy, interfaces are also required for users to review permissions, data access logs, and take actions accordingly.

Quasi-identifiers. Data collected from a user's mobile device do not pose a privacy threat to the user if it cannot be associated back to the user. The association is trivial if the collected data has personally identifiable information with unique identifiers (e.g. device ID). It is argued that without such unique identifiers the association is not possible, and hence privacy guarantees can be provided for any application that does not collect personally identifiable information. Such a perspective is flawed, since non-identifying information can sometimes uniquely identify an individual, especially when they are used in combination. This has been demonstrated many times, over different forms of data. Unfortunately, no technique exists to determine if a certain type of data can be used as a quasi-identifier, and if so, what contribution will it play in the identification. The challenge is in knowing how much sharing a user can do before becoming identifiable by the shared data.

Privacy by design. Mobile applications are developed by a wide spectrum of developers, ranging from novice programmers to dedicated teams of professional software developers. Most applications are designed with the objective of generating revenue, either directly from the user, or indirectly through advertisements or data sharing. Privacy is not an active part of the design of the application, and is added as a supplement at the end of the design cycle. This makes it difficult to restructure the data and control flow in the application once a privacy threat is identified. Privacy efforts in the application development cycle ought to be more proactive and user-centric. An application should have privacy settings enabled by default, and should attempt to accommodate all privacy expectations of the user. In

fact, privacy should be a core functionality in the application, and be embedded in the design from the very beginning. However, application development can be prolonged if such requirements are to be met. An application developer may also not have the training necessary to accomplish such extensive design objectives.

10.6 Conclusion

Mobile devices will dominate the personal computing arena in the years to come. For businesses, this platform provides a rich environment to learn more about the user's intentions, likes, and dislikes, and use it to create custom experiences for every user. Mobile applications engage in the collection of varied types of user data, such as demographics, location, and behavior, and have been found to share it with multiple third-party service providers. This opens up a number of privacy concerns since users are no longer able to control who sees their data, and how it is used. Solutions based on best practice guidelines, public awareness, and novel privacy-preserving algorithms have been proposed to cater to such concerns. Unfortunately, their adoption has been slow, leaving behind a number of challenges to address. An effective solution will require application providers to consider privacy during design, users to be aware of available privacy options, and researchers to involve the existing infrastructure as much as possible in novel proposals.

Acknowledgments We thank the graduate and undergraduate students of Computer Science at the University of Denver who explored the privacy issues and solutions highlighted here, and helped bring it together in the form of this chapter.

References

1. Baden R, Bender A, Spring N, Bhattacharjee B, Starin B (2009) Persona: an online social network with user-defined privacy. In: Proceedings of the ACM SIGCOMM 2009 conference on data communication, pp 135–146
2. Bamba B, Liu L, Pesti P, Wang T (2008) Supporting anonymous location queries in mobile environments with privacy grid. In: Proceedings of the 17th international world wide web conference, pp 237–246
3. Barbaro M, Zeller T (2006) A face is exposed for AOL Searcher No. 4417749. New York Times
4. Cutillo LA, Molva R, Strufe T (2009) Safebook: a privacy-preserving online social network leveraging on real-life trust. IEEE Commun Mag 94–101
5. Dewri R, Ray I, Ray I, Whitley D (2010) Query m-invariance: preventing query disclosures in continuous location-based services. In: Proceedings of the 11th international conference on mobile data management, pp 95–104
6. Dewri R, Thurimella R (2013) Can a Phone's GPS lie intelligently? IEEE Comput Mag 46 (2):91–93

7. Dewri R, Thurimella R (2014) Exploiting service similarity for privacy in location based search queries. IEEE Trans Parallel Distrib Syst 25(2):374–383
8. Dingledine R, Mathewson N, Syverson P (2004) TOR: the second-generation onion router. In: Proceedings of the 13th USENIX security symposium, p 21
9. Enck W, Gilbert P, Chun BG, Cox L, Jung J, McDaniel P, Sheth A (2010) TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones. In: Proceedings of the 9th USENIX symposium on operating systems design and implementation, pp 393–407
10. Federal Trade Commission: mobile privacy disclosures: building trust through transparency. Technical report. www.ftc.gov/os/2013/02/130201mobileprivacyreport.pdf
11. Federrath H, Jerichow A, Pfitzmann A: MIXes in mobile communication systems: location management with privacy. In: Proceedings of the 1st international workshop on information hiding, pp 121–135
12. Fredrikson M, Livshits B (2011) RePriv: re-imagining content personalization and in-browser privacy. In: Proceedings of the 2011 IEEE symposium on security and privacy, pp 131–146
13. Fuchs A, Chaudhuri A, Foster J (2009) SCanDroid: automated security certification of android applications. Technical report, University of Maryland (2009)
14. Gedik B, Liu L (2008) Protecting location privacy with personalized k-anonymity: architecture and algorithms. IEEE Trans Mob Comput 7(1):1–18
15. Golle P (2006) Revisiting the uniqueness of simple demographics in the US population. In: Proceedings of the 5th ACM workshop on privacy in electronic society, pp 77–80
16. Golle P, Partridge K (2009) On the anonymity of home/work location Pairs. In: Proceedings of the 7th international conference on pervasive computing, pp 390–397
17. Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th conference on computer and communications security, pp 89–98
18. Gruteser M, Grunwald D (2003) Anonymous usage of location-based services through spatial and temporal cloaking. In: Proceedings of the 1st international conference on mobile systems, applications, and services, pp 31–42
19. Guha S, Cheng B, Francis P (2011) Privad: practical privacy in online advertising. In: Proceedings of the 8th USENIX conference on networked systems design and implementation, p 13
20. Hazay C, Lindell Y (2008) efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries. Theory Cryptogr 155–175
21. Hoh B, Gruteser M, Xiong H, Alrabady A (2006) Enhancing security and privacy in traffic-monitoring systems. IEEE Pervasive Comput 5(4):38–46
22. Jarecki S, Liu X (2010) Fast secure computation of set intersection. In: Proceedings of the 7th conference on security and cryptography for networks, pp 418–435
23. Kesdogan D, Federrath H, Jerichow A, Pfitzmann A (1996) Location management strategies increasing privacy in mobile communication. In: Proceedings of the 12th IFIP international information security conference, pp 39–48
24. Krishnamurthy B, Wills CE (2009) Privacy diffusion on the web: a longitudinal perspective. In: Proceedings of the 18th international conference on world wide web, pp 541–550
25. Krumm J (2007) Inference attacks on location tracks. In: Proceedings of the 5th international conference on pervasive computing, pp 127–143
26. Kushilevitz E, Ostrovsky R (1997) Replication is not needed: single database, computationally-private information retrieval. In: Proceedings of the 38th annual symposium on foundations of computer science, p 364
27. Malin B (2006) Re-identification of familial database records. In: AMIA annual symposium proceedings, pp 524–528
28. Mayer JR, Mitchell JC (2012) Third-party web tracking: policy and technology. In: Proceedings of the 2012 IEEE symposium on security and privacy, pp 413–427
29. Narayanan A, Shmatikov V (2008) Robust de-anonymization of large sparse datasets. In: Proceedings of the 2008 IEEE Symposium on security and privacy, pp 111–125

30. Narayanan A, Shmatikov V (2009) De-Anonymizing social networks. In: Proceedings of the 2009 IEEE symposium on security and privacy, pp 173–187
31. Narayanan A, Thiagarajan N, Lakhani M, Hamburg M, Boneh D (2011) Location privacy via private proximity testing. In: Proceedings of the network and distributed system security symposium
32. O'Clair B, Egnor D, Greenfield LE (2011) Scoring local search results based on location prominence. US Patent 8046371 B2
33. Qiu D, Boneh D, Lo S, Enge P (2009) Robust location tag generation from noisy location data for security applications. In: The Institute of navigation international technical meeting
34. Reznichenko A, Guha S, Francis P (2011) Auctions in do-not-track compliant internet advertising. In: Proceedings of the 18th ACM conference on computer and communications security, pp 667–676
35. Shin H, Vaidya J, Atluri V (2011) A profile anonymization model for location based services. J Comput Secur 19(5):795–833
36. Souppaya M, Scarfone K (2013) Guidelines for managing the security of mobile devices in the enterprise. Technical report 800-124 Rev 1. National Institute of Standards and Technology
37. Thurm S, Kane YI (2010) Your apps are watching you. What They Know: The Wall Street J
38. Truste: US 2013 Consumer data privacy study—mobile edition. Technical report, Truste Inc
39. United States Department of Justice: Retention Periods of Major Cellular Service Providers. https://www.aclu.org/files/pdfs/freespeech/retention_periods_of_major_cellular_service_providers.pdf
40. Xi Y, Schwiebert L, Shi W (2013) Privacy preserving shortest path routing with an application to navigation. Pervasive Mobile Comput