UPPSALA
UNIVERSITET

IT 14 070
Examensarbete 15 hp
December 2014

Privacy Concerns in Android Devices

Aleksander Okonski

Institutionen för informationsteknologi
Department of Information Technology

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Privacy Concerns in Android Devices

Aleksander Okonski

On a phone that is running the Android operating system, it is possible to create a set of services that can be inserted into a functioning application that would collect a broad set of information. To demonstrate this, an application was written that would utilize a set of services that gathered accelerometer, barometer, GPS, Task, Accounts, Contacts, and Call Log data. The data would then be used to create a detailed overview of a user's everyday activities. This data was collected using hidden services off of the phone and then transferred to a computer for analysis. In this paper I demonstrate the lack of user control over what the application can perform, how apps can run services unhindered, and how data from multiple sources can be combined to create a detailed profile of a user. This research is interesting to individuals who want to understand privacy on the Android devices. The audience that is targeted by this paper are individuals in the security and privacy field. Privacy advocates can use this research to promote user traceability controls in mobile devices.

Handledare: Volkan Cambazoglu
Ämnesgranskare: Per Gunningberg
Examinator: Olle Gällmo
IT 14 070
Tryckt av: Reprocentralen ITC

Table of Contents

1. Introduction

Privacy is becoming an important topic in today's world with more and more people becoming concerned about it. This concern is increasing as the technology grows ever more rapidly and moves towards mobile platforms. In today's world there are 900 million devices [12] running the Android operating system. These are phones or wearable devices that are on or around users most of the day. These devices allow for 3rd party applications to be downloaded and run on them. This opens a large privacy gap as most users are not aware of an app's permissions or what operations the app is performing on the phone. Throughout this paper a description of the problem will be presented alongside an example of how an Android phone could be used to infringe on a user's privacy.

1.1. Thesis

It is easy to write an application for Android 4.4.2 that would abuse granted access rights to OS recourses, collect user data (location, movements, communication history, and tasks) and build a detailed user profile. This in turn invokes serious concerns about user privacy with 3rd party applications.

1.2. Objective

The goal of this project is to build an adversarial application on an Android device that gathers information about user's behavior such as patterns of movement, activities on the device, and information on individuals that the person associates with. The goal for gathering the data is to prove how an app operator could use the data to create an integrated user profile.

- The accelerometer, barometer, and GPS can be used to gain information about a user's movements. In combination, they can provided a graphical portrayal of a user's movement and exact location in three dimensional space.
- The task data is used in combination with other data to infer what the user was doing at a particular time.
- Accounts data is collected to see what services a user utilizes.
- Using the address book and phone log gives access to the user's social circles.

The analysis of business, legal and social consequences of leaking private profile from the phone is not in the scope of this project.

1.3. Outcomes

This study shows that a smart device, which is on a user most of the time, has its privacy drawbacks. The sample application created during this research demonstrates that a hidden service can easily gather data in the background and send it to an offsite location. After data collection, an adversary is able to interpret the collected data to build user profile. In today's world focused on mobile and wearable technology, it may be beneficial to take a step back and examine some of the privacy concerns.

1.4. Related Work

There have been other papers published that relate the use of individual phone features to user's behavior. Tracking user's location has been performed in a study in which the authors used a similar method of tracking the users' position by using GPS [8]. However, instead of recording the data they would perform an action based on the users' location. There have also

been papers that depict activity recognition via the accelerometer [5][7]. These papers describe how, via a person's movements, it is possible to recognize jogging, sitting, standing, etc. In both of the research papers the groups only focused on the accelerometer. They did not look at any of the other sensors present on the phone. Using a similar approach this research integrates individual data points to construct a comprehensive view of an individual. In addition, there have been multiple articles published on the internet that talk about the pitfalls of the Android permission system. These articles are a crucial stepping stone for the app that was created for the project. We are now starting to see companies and governments utilizing user's phones as personal spying and data collection devices [6].

2. Background

Android is an operating system produced by Google for phones. When Google first acquired Android, their mission was to create an open source platform that empowers developers to create and publish their own applications (apps). Unlike other phone platforms, where apps have to be registered before they are put on the online store, Google allows anyone to upload an app for others to download. This has created both a wide market for many legitimate apps and allows apps to abuse the system and perform adverse actions [13]. Before a user can use a new downloaded app, he/she must allow features of the phone, such as GPS, to be used by that app. According to previous research [1], only a small percentage of people ever read or understand the permissions that are asked for by the app. Additionally, Android does not allow the user to manually change app permissions. Therefore, if a user wants to use the app, they must fully accept all permissions or not install the app. Once installed, an app is able to run unconstrained within its allotted permissions. This is worrying because it is hard to tell exactly what an app does in the background.

2.1. Android Ecosystem

Apps in Android OS are separated from each other using the Dalvik Virtual Machine (DVM). DVM protects against apps trying to gain access to Android OS or any other app. This protects the phone from the app that tries to perform malicious actions such as deleting other apps, accessing low level services like network I/O, or modifying the Android kernel itself. For legitimate apps to perform tasks Google has built an extensive API that allow applications to utilize system resources in a controlled manner. Due to the fact that apps can be created to perform a wide variety of tasks, the Android API is versatile and offers features such as: looking at SMSs on the phone, gaining access to phone logs, or finding the location of the phone. Such extensive APIs allow app developers to create some stunning apps, but it also allows personal users information to become easily available to these apps. Data on an Android phone can be placed in two different ways: 1) static and 2) active data. Static data represents the data that a user would have entered into the device him/herself or data that is stored on the device. Examples of such data are email addresses, names, contacts, calls, SMS's etc. Active data is data that is generated around the phone. This can include IP address, location of the phone, tasks, accelerometer, barometer, etc. This data usually does not persist on the phone for a long time.

2.2. Android Permissions

In Android, permissions are marked with 4 different levels: normal, dangerous, signature, and signatureOrSystem. The last two were not focused on because they require the app to come from the device distributor or Google itself. Androids apps access OS resources through the `Android:protectionLevel`[9] attribute in the permission app manifest located in Appendix – A.

When a permission has a normal setting, a user is not notified during the installation of the app. If an app has a dangerous setting then there is a notification during installation asking a user to accept these permissions explicitly. Google has classified the different permission types according to the risk they pose. *"... the potential risk implied in the permission and indicates the procedure the system should follow when determining whether or not to grant the permission to an application requesting it."* [2] This allows the system to check what permissions the app has been granted before giving it access to the system components.

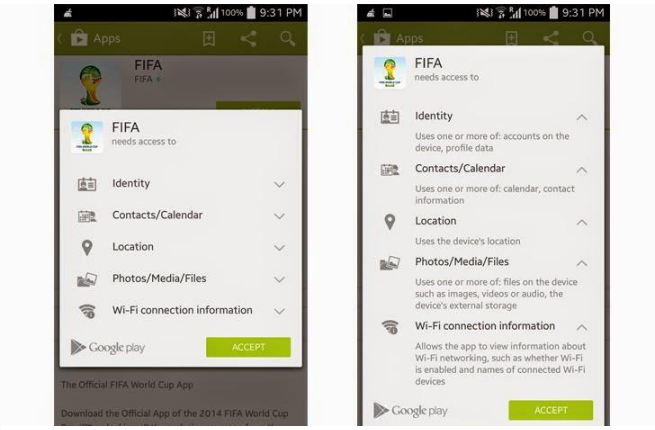

Figure 1

2.3. Android Permission Groupings

When Android was updated to 4.4.2, Google implemented auto update and group permissions (Figure 1). This groups similar permissions under a global group name. The 145 different permissions are grouped into 13 different categories. The consequence of this is that the permissions are masqueraded so that a user does not fully know what the app is going to use on the phone. An example of this grouping is shown on the left where the app is asking for entire set of data and resources to be accessed. This also poses another problem for users. An app can now auto update. Once an app is installed it will not ask a user to accept any more permissions again if the user has already allowed the permission group. An example of this would be SMS. There are three different permissions for writing, receiving, and sending SMSs. If an app first requests to only receive SMSs and the user accepts it, in subsequent self-installed update the app could give itself writing and sending privileges without ever notifying the user.

2.4. Intended Purpose

There are also legitimate purposes in having an application use certain features of the phone that require permissions. Such apps include Nike+ Running or Sleep Cycle, both of these apps try and help out with everyday activities. Nike+ Running attempts to challenge us to have a better workout by monitoring running distance, calories burned, and the users speed. Sleep Cycle helps gauge a user's sleep patterns and tries to wake the user between their rem cycles. Both apps are using phone features to provide feedback about user activities.

2.5. Privacy Vulnerability

An adversary app developer and operator can create an app with a useful utility such as game, productivity, or communication. This app will have legitimate function but may also include background services that will abuse the access and collect user's data beyond the official intent of the app. The collected data can then be sent to the app operator for further processing and user profile building.

3. Methodology

Phase – 1 Learning about the Technology

The project started with no prior knowledge of Android OS and its ecosystem. It consisted of researching on how apps communicate with the underling Android platform and learning how the DVM worked. The next step after understanding the DVM was to move on to learning the development of Android applications.

Phase 2 – Scoping the Project

When the project began there was a plan for more data collection from different phone features such as "SMS" and "Feeds." This, however, started to prove impractical due to project time constraints. A couple of key areas were selected that represent the majority of informative data that is on the phone. These selections encompassed a user's movement patterns, activities on the phone, and social contacts.

Phase 3 – Building the Application

After learning about the Android ecosystem and picking the phone features that were going to be explored, I started to code the application. This proved to be one of the hardest tasks because individual components of the application were not created to run as a services. An example of this is obtaining the task information from the phone. This feature was not intended to be placed in a service that would constantly run. A remediation for this was a service that calls another portion of the application which then retrieves task information.

Phase 4 – Data Capture and Parsing

Once the application collected the necessary data it was time to offload the data to a server for data analysis. This was problematic as some of the data files were 23Mb large. The server would terminate the connection and not allow the files to be transferred. It became clear here that most servers have a limit on the size of uploaded files. Default configuration was not able to be adjusted because the server being used was not under my control. The remediation of this problem was to use the file transfer protocol to send files from the phone. Once the server scripts parsed the files to eliminate data inconsistencies I put the data into standardized format files. This made it simple to import into Excel and graph the data. Once graphs were created I started to look for patterns or other interesting characteristics and began writing about the data interpretation and findings.

Applied Skills

This project utilized the skills that I have learned through my bachelor studies. The application was written in Java and used networking components to transfer data off of the phone. Courses in architecture and operating systems helped with quickly ramping up on the workings of the Android OS and communication in the VM.

4. Application

4.1. Introduction to Application

For conducting this analysis the Android platform (Kitkat 4.4.2), which runs on the Samsung Galaxy S5, was used. The user profiles were created by gathering the information off of the phone, including sensor information, user accounts and logs, and sent to a database for further profile creation to see patterns, user habits, and general information. To create a user profile, both static and active data have to be analyzed together. Once a user profile is created and continually updated, it can be used to target ads, track a person, and learn about user's behavior and private life. There are a multitude of different reasons why adversary would like to find out about the users private life.

Static Data:
1. Accounts – Any information that is used to identify an individual.
2. Contacts – Stored information about others on the users' phone.
3. Call logs – Data about who calls and sends messages to the phone.

Active Data:
1. Location – Phone physical location.
2. Motion Information – Moment of the phone in 3D space.
3. Barometer – Pressure of the atmosphere around the phone.
4. Tasks – View the other tasks that are occurring on the phone.

To gather the required active and static data, certain permissions must be used. The chart below shows relations between the permission level and the data that was gathered from the phone.

Static Data	**Name of Permissions in Android [3]**	**Permission Level [4]**
Accounts	Android.permission.GET_ACCOUNTS	normal
Contacts	Android.permission.READ_CONTACTS	dangerous
Call Logs	Android.permission.READ_CALL_LOG	dangerous
Active Data		
Location	Android.permission.ACCESS_FINE_LOCATION	dangerous
Motion Information	None	
Barometer	None	
Tasks	Android.permission.GET_TASKS	dangerous

Table-1

I have created an application on a Samsung Galaxy S5 (Figure 2) that runs a service in the background on the phone to collect both types of static and active data. This data is sent back to a server where it is parsed and analyzed for patterns. The goal of the final step is to

take the data and create a profile of the user. The code that was used to gather information could be hidden inside any legitimate app. As described in the background section, getting a user to accept Android permissions is relatively simple. Only around 17% of users even know about permissions and look at them [1]. This, coupled with the fact that many applications require a large number of permissions, should not alert a user about any wrongdoing that the app maybe performing.

For data to be collected without individual's knowledge requires that data collection has to occur even if the app is not running. An application typically has three states: active, paused, and stopped. However, an application can launch many services that will remain active outside of the main applications execution cycle. In addition, a service runs in the background; therefore, there is no visual information to the user that a service was started or is running. The Android operating system reserves the right to terminate any service if memory is low. However, it is possible to create a service that will restart itself when memory becomes available or the phone is rebooted. The application that was created through this research is a proof of concept that demonstrates the ability to develop and run hidden services that can collect user data.

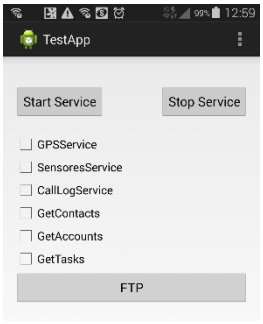

Figure 2

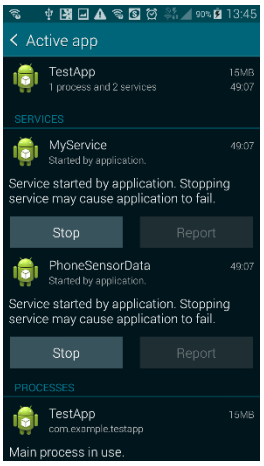

Figure 3

For the purpose of testing and analyzing the information the application that was developed has a user interface shown in Figure 2. The interface has start and stop buttons. These buttons control start and stop of data collection while the check boxes select different data collections. The "FTP" button uploads the collected data to a server. Due to the fact that this app is for experimental purposes it does not fully attempt to be run incognito on the phone. Later in this paper a description of some techniques that demonstrate how an application can attempt to conceal its services from a user are mentioned. For all scenarios in the scope, all the checkboxes are turned on. Once the "Start Service" button is clicked several services are started as shown in Figure 3. These include data collection from the GPS, accelerometer, barometer, and tasks. These services run unhindered in the background collecting information. The other data such as call log, contacts, or accounts is performed on application launch time.

4.2. Service Hiding Techniques

There are a number of ways to mask the applications activities. A user would be able to spot that a service was running if they went into the settings of the phone and viewed the apps subcategory (Figure -3). From there they would have to navigate to the running applications tab. On that page the phone will display what apps are running and what services are associated with the application. This may lead to users finding out that an application is running services that should not be present with the application. To address this, when a user enters the setting of a phone, the application can detect the navigation to the "settings" task and temporarily kill all the services associated with itself. The application then needs to set up a broadcast call that will rerun the services after an allocated amount of time has passed.

A significant challenge for this type of data gathering application is that it continually runs. This imposes a large strain on the battery. It may be noticeable to some users that the battery is running out quickly. The application should enable/disable services to limit the amount of processing that the phone needs to perform.

4.3. Profile Data

To collect the data, an experiment was set up utilizing the created application and a Samsung Galaxy S5. This experiment gathers real life data from an app that tracked an individual. This project started with using the phone for a couple of days. To produce some user data a Google account was created and connected to the phone. The application was installed and run on the phone. With the application running I proceed with normal daily routines. An hour long sample of data was collected due to the size limitations for file transfer to the server where the data and graphing analytics would occur.

4.4. Active Data Collection

4.4.1. Accelerometer

The first service that was implemented was to track acceleration in the X, Y, and Z coordinate spaces (Figure 4). Capturing accelerometer data does not require any permission from the user. The data can be gathered without the user ever noticing or having to accept anything. This app's feature included in the service tracks all the user movements. When accelerometer data is presented on a graph certain patterns start to emerge, allowing interpretation of the user movement in 3D space.

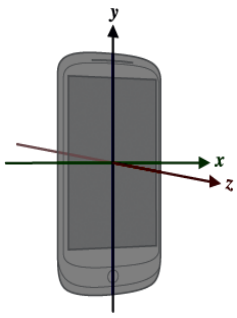

Figure 4

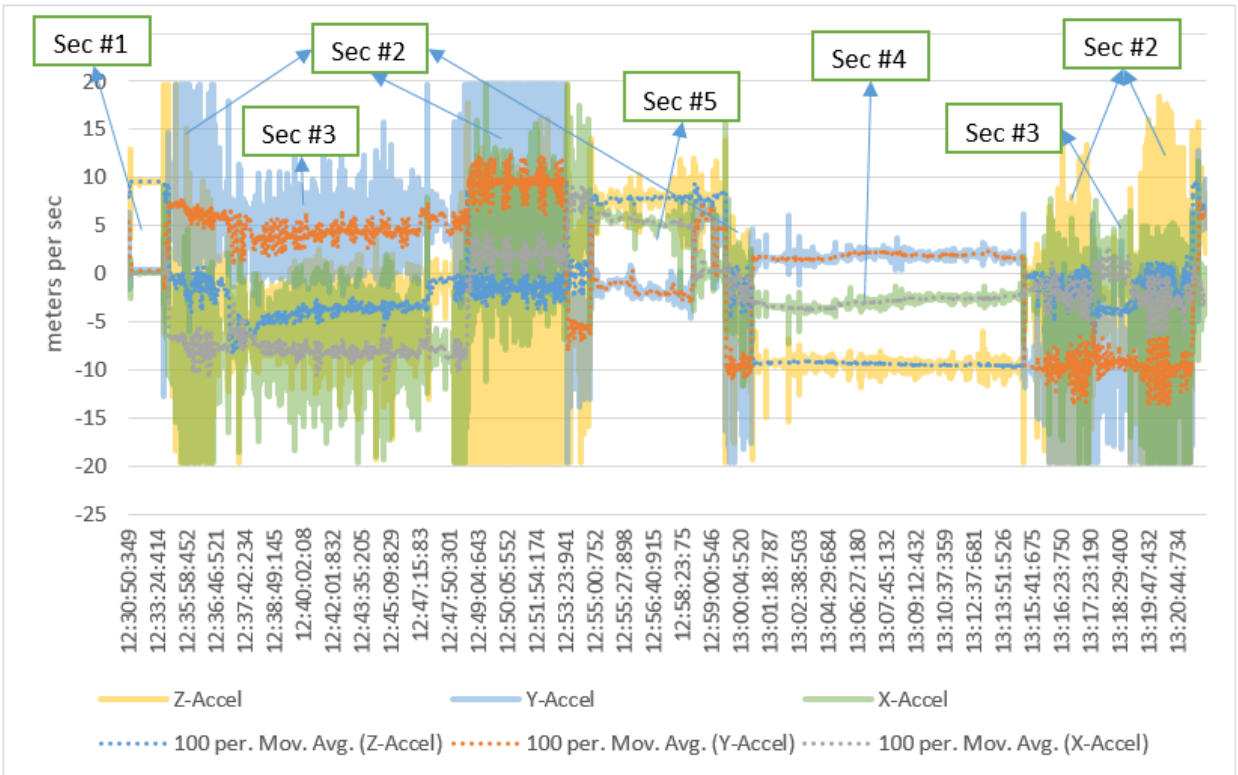

Diagram 1 – Accelerometer Movements

Diagram-1 shows acceleration data captured from the phone. To understand the correlation between accelerometer data and a user's actions relies on two methods: 1) being present when the data was recorded. 2) utilizing the research done in [7], in which they focused entirely on the accelerometer and mapped a multitude of different movements to the

data they captured. Using both methods, it was possible to come up with the following chart that depicts each section of data from my experiment.

Section #1	This section represents how accelerating data looks like when the phone is laying on top of a table face up. The Z axis is in the positive showing that the phone is lying face up on a surface.
Section #2	This section of the data shows large data lines reaching the max value of the sensor. This indicates that the phone is in a pocket and that the user is walking with it. When a user's foot hits the ground the sensor will register a very large acceleration/decoration causing the large spikes.
Section #3	The data in this section shows very gentle movements. This comes from a user biking. When a user bikes, there are no sudden jolts that wound cause the sensor to spike because pedaling is a fluid movement.
Section #4	In this section, there is a steady line for all 3 axis. This comes from a user keeping the phone in the pocket while sitting/standing in one position.
Section #5	This section represents a user who has the phone in their hands and is using it. There are slight jitters but overall the axis seem fixed to a linear pattern.

Table-2

4.4.2. *Barometer*

The next sensor that was looked at was the barometer. It is similar to the accelerometer sensor, such that is does not require any privileges to run. The barometer sensor can tell if a person is moving up or down by sensing air pressure changes. Although the graph is harder to interpret than the acceleration graphs, it is possible to obtain useful information. Using the trend line, it is possible to tell if the person is moving up or down, while the actual graph shows the velocity of movement.

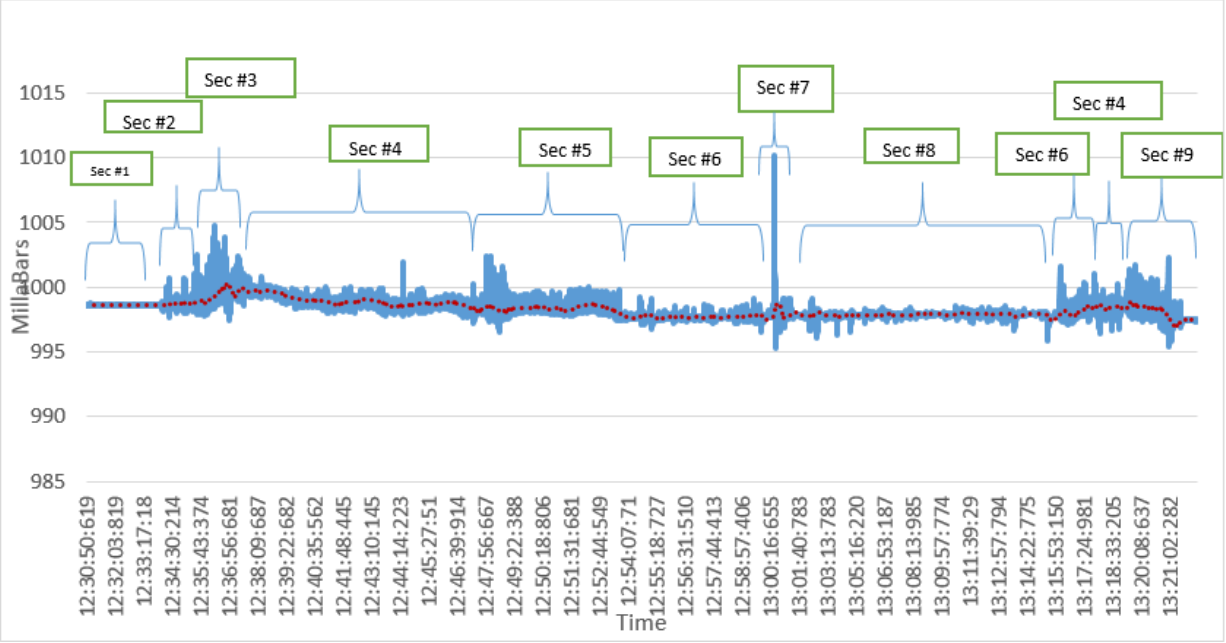

Diagram-2 – Barometric Data

The barometric data is graphed in Diagram-2 with the raw data represented in blue while the average values are shown on the red dotted line. The red line gives an overall representation of change in a user's elevation while the blue line allows to spot different patterns for different movements. The raw data jitters when the phone is moved, because there are pressure changes occurring due to the user's movements. The chart below depicts the user's action to the barometer data. In the following table a breakdown of each section from the graph is shown. It is important to note two cases with the pressure changes. First, the pressure increases closer to the ground lower altitude. Second, the pressure in the atmosphere moves up and down over time; therefore, only short time intervals should be used to compare change in the elevation. The correlation between the actions and the data came from being present while the data was collected.

Section 1	Represents a phone stationary on a flat surface.
Section 2	Represents a phone being handled.
Section 3	Represents a user walking down stairs.
Section 4	Represents a user biking.
Section 5	Represents a user walking.
Section 6	Represents a user holding a phone and using it.
Section 7	Represents a user opening a door.
Section 8	Represents a user sitting.
Section 9	Represents a user walking up stairs.

Table-3

4.4.3. Location

A location data feed was added to build a more extensive movement profile to gain more information about the user activities. The application obtains location data by calling Android's location data API. There are two types of location data that can gather from the phone, network or GPS data. Both will provide a user's general location with the exception that using GPS will provide a much more precise estimate. For this application the GPS location was utilized. Graphing the location data onto a map gives detailed representation of the path that the user has taken. It is important to note that location data only updates when a person moves from a spot. Therefore, if a user stands in one spot for a longer period of time the graph will not update. Using time data associated with each point shows the speed at which the user was moving and if they were stationary at any location.

When the location data is plotted onto a map Diagram-3 a clear path of movements can be seen. Combining that with time/date data tells where the user was at what time. This can be seen in Diagram-4. A good example of how time data can be used is marked by section #1. There we can tell that a user had stopped in that location for approximately 24 minutes.

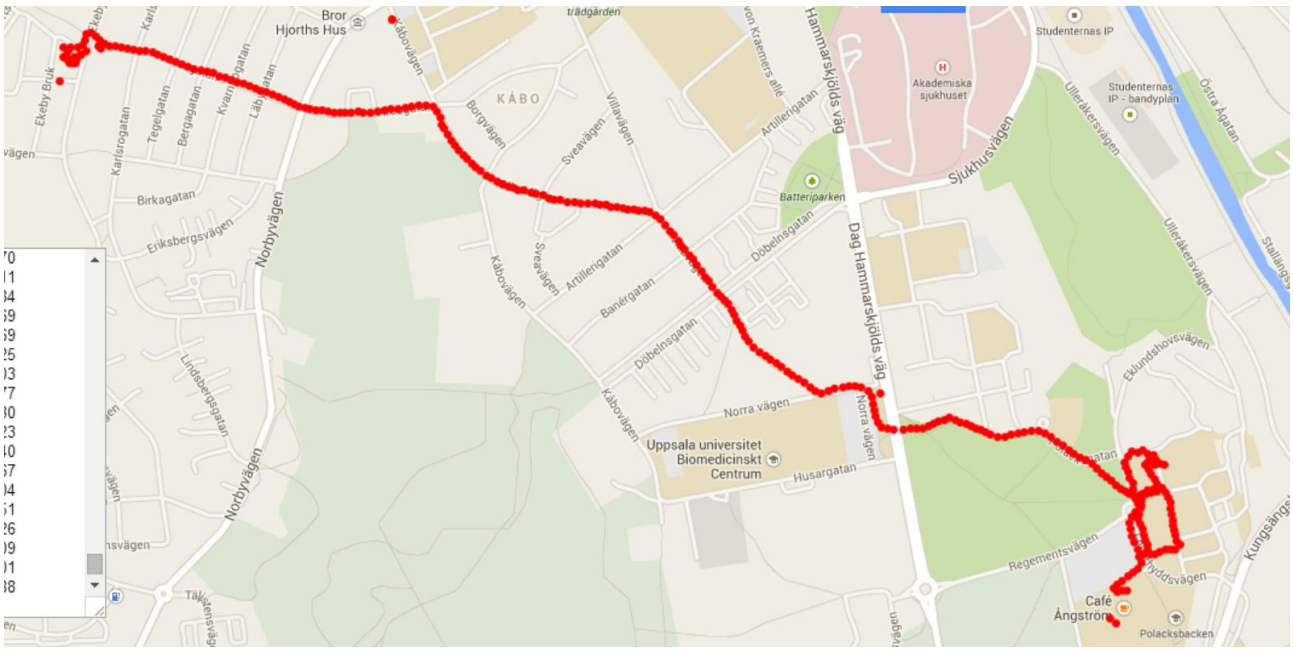

Diagram - 3 Location data represented on the map [14]

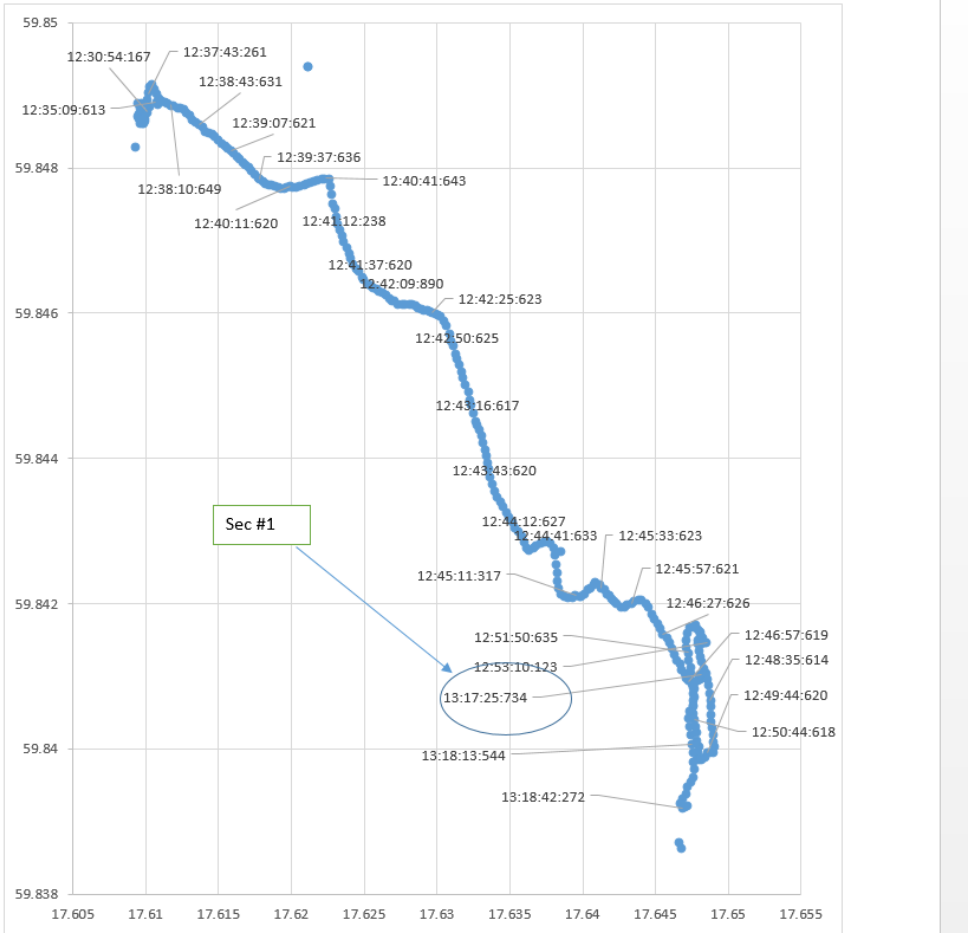

Diagram -4 Location mapped to the timeline

4.4.4. Tasks

To learn more about a user's activities tasks on the phone where used to tell what the user was doing on the phone. To access this information the app needs to have the "task" permission enabled. This level of permission is associated to the dangers group. With the phone task log we can get a table of what the application the user was using and for how long.

Now we are able to tell where the user was physically located but also what they were doing on their phone at that time. Table-4 contains a list of applications running and when they were being used. An example of this information is the music player usage showing that the user was listening to music while biking or that the user was playing Angry Birds while being stationary in a particular place.

Table-4

Application	**Time From**	**Time To**
com.sec.Android.app.launcher	12:30:39:490	12:30:43:472
com.example.testapp	12:30:48:466	12:30:55:336
com.sec.Android.app.launcher	12:31:00:326	12:34:15:330
com.sec.Android.app.music	12:34:20:330	12:53:25:323
com.sec.Android.app.launcher	12:53:30:780	12:54:40:319
com.rovio.angrybirds	12:54:45:318	12:58:30:531
com.sec.Android.app.launcher	12:58:35:343	12:58:40:324
com.lslk.sleepbot	12:58:45:314	12:59:35:315
com.sec.Android.app.launcher	12:59:40:351	13:22:05:352
com.Android.systemui	13:22:10:361	13:22:10:361
com.example.testapp	13:22:15:365	13:27:50:315

Static Data Collection

4.4.5. Accounts

The accounts permission allows to gain access to a user's email address. Gaining a user's email address can lead to a merit of attacker vectors on a person. Unlike a password, a person's email cannot be changed. This email can be used to authenticate do different websites. A great article about how an email or username can be used by attackers is [11]. The account permission on Android is normal. Knowing the accounts of a user allows to find out more about the user's virtual foot prints. Emails, Facebook name, and even which emails are associated to which accounts can help bind the real world to the virtual one. Viewing Table-5, it is possible to see which emails are associated to which accounts.

Table-5

Accounts	**Page / Application used**
aleksander.uppsala@gmail.com	com.google
aleksander.uppsala@gmail.com	com.osp.app.signin
aleksander.uppsala@gmail.com	com.dropbox.Android.account

4.4.6. Call Log

To understand the user's interaction with other people by analyzing their phone activity. By getting a user's call log it is possible to tell with whom they communicate with regularly via call or SMS. By looking at Table-6 the user's full call history can be seen, including who and for how long they talked and the time the call took place. Taking this data and then mapping with the location we can then tell when, at what location, and with whom a user communicated.

Table-6

Phone Number	**Caller Name in Phone**	**Data of call**	**What happens with call**	**New call**	**Time is Seconds**	**Time formatted**
4*********3	null	18-Jun-2014 19:02:51	2	1	17	00:17

*Phone numbers are masked for privacy

4.4.7. Contacts

Knowing personal contacts helps to establish who the person knows and communicates with. In Table-7 we can see an example of a contact from a phone. Contacts may appear on the phone in different manner. A user could type a contact in, connect to a service that will sync their online contacts (such as a social network), or import from a SIM card. The phone is able to keep track of what type of contact by the "type" parameter as seen in Table-7. Type number 1 represents that the contact is a home contact, number 2 represents a contact from work, and number 3 represents a contact from another source. As contacts come from different places, not all the metadata may have information. In some cases there may be no address, email, or other information.

Table-7

First Name	**Phone number**	**Phone number**	**Phone number**	**Email**	**poBox**	**street**	**city**	**state**	**postal Code**	**country**	**type**
Test User	4*******5	4*******0	4*******0	test@test.com	null	12345 999th Way	Redmond	Wa	98123	United States	1

*Phone numbers are masked for privacy

5. Discussion

5.1. Accelerometer, Barometer, and Location Data

The study shows that when different data sets from a phone are mapped to each other and analyzed together, a detailed picture of the phone owner can be created. Taking each individual data set separately may not lead to much useful information. For example, taking only one axis from accelerometer data will not lead to much information. It would only be possible to determine movement vs non-movement. When all three axis are added together, it gives the ability to tell different movement of the user. This tells us if the user is walking, running, biking, sitting, etc. Yet this data still does not provide a clear picture of where a person is located. Combining the two different sets of data together (accelerometer and

barometer) allows a broader view of what actions the user was performing. By adding barometer data, it is possible to tell if a user's movement is vertical as well as horizontal. Although individual barometer data points do not give a precise description, a trend line tells a more complete story. If a user is riding a bike, it is possible to tell if he/she is going uphill, downhill, vertically, etc. Lastly, location data will allow to learn whereabouts of the user. If the location is gathered using network data it will not be as precise as GPS data but offers an interesting advantage. Once a user's frequented location are known, other location-based attack vectors can be utilized. This includes drive-by downloads [15], phishing [16], or rogue access points [17]. Combining location data with the previous accelerometer and barometer findings can provide an accurate representation of where the person is, what movement he/she are performing, and what altitude he/she are at.

Combining location, barometer, and accelerometer data together can lead to important conclusions.

Accelerometer + Location data	With both these data sets graphed it is possible to tell where a person has gone and how they he/she traveled there.
Location + Barometer data	With the barometer data included it is now possible to tell if a person had gone to a different floor.
Location + Barometer data + Accelerometer	With more data it is possible to know when a user is sitting on the 2nd floor using his/her phone.

Table-8

If these data sets were collected over multiple days, it is possible to start reconstructing a person's life, habits and routines such as the time a user gets up, which could be observed when the first accelerometer movement is registered. It is also possible to figure out where a user lives, works, how they get to work, how they relax and where. It is possible to tell details like what floor a person works on by the change in pressure when they enter the office.

5.2. Tasks and Accounts

Understating what tasks are running on the phone can help to add individual's interests to the profile. An example would be seeing a health app such as Nike+ Running show up in a user's tasks regularly. This can hint that a user is conscious about health. We can then use the previous data to see other statistics like when, where, or for how long did the person run. Using task data and looking at the other data (location, movement) we can start to recognize what the user was doing and where at any time. Some tasks require a person to be logged in e.g. Google+. The accounts section of the phone stores the user's email with the account that it is connected to. Therefore, knowing that a user uses an app with a certain login can lead to a user's online activities to be connected back to the individual and his/her location context. This opens wide opportunity to understand characteristics of individual's life: health, finance, social interests, religious affiliations, hobbies, political views, etc.

5.3. Contacts and Call Log

Having a person's contacts can lead to more private information becoming available. It is possible to know with whom they associate and with whom they have been in contact. Using the type parameter, in Table-7, we can tell what form of contact it is, whether it is a home, work, or contact. Most social media sites allow a user to connect their address book to fill in any missing information about their friends. Obtaining this data can then be combined

with the call log data to create a web of social communication interaction. In addition, due to social media connections, it is possible to gain information that people would like to keep secret. For example, many people disallow viewing of their friends on social media or want to keep certain acquaintances secret [10]. With obtaining the contact list from the phone all of these secret connections are revealed. Cross referencing the phone number to the call log can show the frequency of contact the user has. This can then show if they are close friends or distant acquaintances. Using the location and time parameter can show if the user is a work associate or a friend. Using the name of the individuals in the contact group it is possible to tell relationships. Using "dad" / "mom" as a contact name can show who the relatives are. Looking at the street address of a contact and the location information of the phone, a user can give away the relative frequency of visits that occur. This can help to tell whether the contact is in the same location or a distant relations.

5.4. Sample Interpretation

From the gathered data in this experiment we can try and deduce a user profile. This started with the user accounts, noticing that the main phone account (com.google) is aleksander.uppsala@gmail.com. Using the email address it is possible to find the users profile on Google+. This allows for gleaning information about the user's name, social profile, etc. Using the contacts shows to what contacts the user is connected. This may then show that the user calls one phone number very often. It is possible to tell that the individual contacted is a family member because of the filled-out contact card. Using the location data gathered, it is possible to tell that the user was at three locations during the data recording. At the second location, the user used his/her phone to play some games as the user was sitting for a period of time. It is possible to tell that the user has a bike that he/she is using to get around. When the user is at each location, he/she either walks up or down a set of stairs. Although this sample interpretation came from being present while data was collected it should possible for a machine learning algorithm to reach a similar result.

Summary based on the interpretation of user activity from the project:

Time	**Description**
12:30-12:32	User is at home and the phone is sitting on a table.
12:33-12:36	Walking around with phone in pocket.
12:34	Turns music on.
12:36-12:36	Walking down a flight of stairs.
12:36-12:47	Riding a bike from home to a university building.
12:47-12:53	Walks around university building.
12:54	Turns music off.
12:55	Walks up stairs.
12:55-13:00	Plays angry birds while being stationary.
13:01	Walks for a minute then sits down.
13:02-13:15	Sits in one place at the university building.
13:15-1317	Walking down a set of stairs.
13:17-:13:18	Riding bike to another university building.
13:18-13:20	Walking up stairs.

Table-9

5.5. Future Work

There are still a broad area of research that can be performed around privacy and mobile applications. For a start, there are a couple of additions and extensions that I would like to see for my project.

- Adding even more phone sensors would allow for a more detailed user profile to be created. Adding gyroscopic, light, and temperature sensors would give more abilities to fine-tune recognition of a person's activities and surrounding.
- The collection process currently collects the data continually. It would be possible to research a way to collect meaningful data only. An example of this would be location data, if a user does not move from a location the data collection should pause.
- Implement ways that an app can mask the services it is running. I was not able to implement the service hiding technique stated earlier in the paper due to time constraints.
- Improve the data upload process so that it is faster and less noisy on the network. Currently the application uploads the data files in a single bundle. This causes a lot of network traffic, therefore if the application could send data periodically and compress the files, the resulting network load would be substantially limited.
- Optimize the app code to be less of an impact on the battery life.
- Build complex business intelligence (BI) functions to parse and analyze data automatically to build the profile.

These are some of the areas this work can leverage. Mass digital tracking is a new field of privacy that has not been intensely scrutinized.

6. Conclusions

The project shows that malicious app running on the Android OS that takes advantage of dangerous permissions can gather and send substantial set of data from the phone to the app operator allowing detailed user profile creations. From the gathered data, it is possible to tell a lot about an individual person. These profiles can later be used for further BI processing of the data gathered over time to create detailed information of the user's life. It would be beneficial for Google to rethink its strategy on permission in Android OS. This project was not created to demonstrate how this problem could be used within a real-world context. There are legitimate uses for the features described throughout this paper. Most applications will collect a user's data for a legitimate purpose such as fitbit, which track a user's fitness and diet. However, in this research, I point out how these functions can be used by an adversary. I hope to pursue a job in the field of computer security and privacy; therefore, this project relates directly to my plan.

7. Disclaimer

For this study the application has gathered data from one phone. The profile built by this application was done with the consent of the individual involved in the project. This application was not released onto the Google Play Store.

8. Appendix

`Android:protectionLevel[9]`

Characterizes the potential risk implied in the permission and indicates the procedure the system should follow when determining whether or not to grant the permission to an application requesting it. The value can be set to one of the following strings:

Value	Meaning
`"normal"`	The default value. A lower-risk permission that gives requesting applications access to isolated application-level features, with minimal risk to other applications, the system, or the user. The system automatically grants this type of permission to a requesting application at installation, without asking for the user's explicit approval (though the user always has the option to review these permissions before installing).
`"dangerous"`	A higher-risk permission that would give a requesting application access to private user data or control over the device that can negatively impact the user. Because this type of permission introduces potential risk, the system may not automatically grant it to the requesting application. For example, any dangerous permissions requested by an application may be displayed to the user and require confirmation before proceeding, or some other approach may be taken to avoid the user automatically allowing the use of such facilities.
`"signature"`	A permission that the system grants only if the requesting application is signed with the same certificate as the application that declared the permission. If the certificates match, the system automatically grants the permission without notifying the user or asking for the user's explicit approval.
`"signature-OrSystem"`	A permission that the system grants only to applications that are in the Android system image *or* that are signed with the same certificate as the application that declared the permission. Please avoid using this option, as the `signature` protection level should be sufficient for most needs and works regardless of exactly where applications are installed. The `"signatureOrSystem"` permission is used for certain special situations where multiple vendors have applications built into a system image and need to share specific features explicitly because they are being built together.

9. References:

[1] Felt, Adrienne, Elizabeth Ha, Serge Egelman, Ariel Haney, Erika Chin, and David Wagner. "Android Permissions: User Attention, Comprehension, and Behavior." University of California Berkley.

[2] Android Developers. Google. Web. 21 May 2014. <http://developer.Android.com/guide/topics/manifest/permission-element.html>.

[3] "Manifest.permission." Android Developers. Google, n.d. Web. 22 May 2014. <http://developer.Android.com/reference/Android/Manifest.permission.html>.

[4] "Android/platform_frameworks_base." GitHub. N.p., n.d. Web. 22 May 2014. <https://github.com/Android/platform_frameworks_base/blob/master/core/res/AndroidManifest.xml>

[5] Kwapisz, Jennifer R., Gary M. Weiss, and Samuel A. Moore. "Activity Recognition Using Cell Phone Accelerometers." *ACM SIGKDD Explorations Newsletter* 12.2 (2011): 1-9. Web.

[6] "Hacking Team's Tradecraft and Android Implant." *The Citizen Lab Police Story Hacking Teams Government Surveillance Malware Comments*. N.p., n.d. Web. 28 June 2014. <https://citizenlab.org/2014/06/backdoor-hacking-teams-tradecraft-Android-implant/>.

[7] Kim, Joo-Hee, Sang-Ha Nam, Se-Kyeong Heo, and In-Cheol Kim. "Design of an Activity Recognition System Using Smartphone Accelerometer." *KIPS Transactions on Software and Data Engineering* 2.1 (2013): 49-54. Web. Yonsei University

[8] Kushwaha, Amit, and Vineet Kushwaha. "Location Based Services Using Android Mobile Operating System."

[9] *Android Developers*. Google, Web. 30 June 2014. <http://developer.Android.com/guide/topics/manifest/permission-element.html>.

[10] Dey, Ratan, Zubin Jelveh, and Keith Ross. "Facebook Users Have Become Much More Private: A Large-Scale Study." *Facebook Users Have Become Much More Private: A Large-Scale Study* (n.d.): n. pag. Web. <http://cse.poly.edu/~ratan/facebookusertrends.pdf>.

[11] "SpiderLabs Anterior." *'SpiderLabs Anterior'* N.p., Web. 03 Aug. 2014. <http://blog.spiderlabs.com/2014/06/from-a-username-to-full-account-takeover.html>.

[12] Bort, Julie. "Google: There Are 900 Million Android Devices Activated." *Business Insider*. Business Insider, Inc, 15 May 2013. Web. 03 Aug. 2014. <http://www.businessinsider.com/900-million-android-devices-in-2013-2013-5>.

[13] Dunn, John E. "Google Cracks Whip to Stop Android Play Store Apps Abusing Users." *TechWorld*. N.p., n.d. Web. <http://news.techworld.com/security/3509243/google-cracks-whip-to-stop-android-play-store-apps-abusing-users/>.

[14] "No Bull, Just Hamster - Geocode, Convert Coordinates and Map Your Locations." *No Bull, Just Hamster - Geocode, Convert Coordinates and Map Your Locations*. N.p., n.d. Web. 08 Sept. 2014. <http://www.hamstermap.com/>.

[15] "Drive By Downloads: How To Avoid Getting A Cap Popped In Your App." - *OWASP*. N.p., n.d. Web. 08 Sept. 2014. <https://www.owasp.org/index.php/Drive_By_Downloads%3A_How_To_Avoid_Getting_A_Cap_Popped_In_Your_App>.

[16] "Phishing." - *OWASP*. N.p., n.d. Web. 08 Sept. 2014. <https://www.owasp.org/index.php/Phishing>.

[17] "Rogue Access Point." *Wikipedia*. Wikimedia Foundation, 24 Aug. 2014. Web. 08 Sept. 2014. <https://en.wikipedia.org/wiki/Rogue_access_point>.