Can Smartphone Users Turn Off Tracking Service Settings?

Veelasha Moonsamy
School of IT, Deakin University
221 Burwood Highway,
Burwood, Australia
v.moonsamy@
research.deakin.edu.au

Lynn Batten
School of IT, Deakin University
221 Burwood Highway,
Burwood, Australia
lynn.batten@deakin.edu.au

Malcolm Shore
NBN Co.
360 Elizabeth St,
Melbourne, Australia
MalcolmShore@nbnco.com.au

ABSTRACT

Tracking services play a fundamental role in the smartphone ecosystem. While their primary purpose is to provide a smartphone user with the ability to regulate the extent of sharing private information with external parties, these services can also be misused by advertisers in order to boost revenues. In this paper, we investigate tracking services on the `Android` and `iOS` smartphone platforms. We present a simple and effective way to monitor traffic generated by tracking services to and from the smartphone and external servers. To evaluate our work, we dynamically execute a set of `Android` and `iOS` applications, collected from their respective official markets. Our empirical results indicate that even if the user disables or limits tracking services on the smartphone, applications can by-pass those settings and, consequently, leak private information to external parties. On the other hand, when testing the location 'on' setting, we notice that generally location is not tracked.

Categories and Subject Descriptors

D.4.5 [**Reliability**]: Verification; D.4.6 [**Security and Protection**]: Access Controls

General Terms

Location Services, Advertising, Man-in-the-Middle, SSL

Keywords

Android, iOS, Smartphones, Location Services, Advertising

1. INTRODUCTION

Between 2011 and 2013, smartphone consumerization has led to an upsurge in the number of devices sold [8]. Leading the competition are `Android`, developed by Google [3] and `iOS`, from Apple [22]. According to the author of [42], as of June 2013, the number of activated devices running on `Android` has reached 900 million, compared to 600 million activations for `iOS`. However, the gap for the number of downloaded applications on these platforms is smaller with 48 billion for `Android` and 50 billion for `iOS`, with the expectation that `Android` will surpass `iOS` by the end of 2013 [29]. These observations suggest that competition between these two platforms will continue well into the future.

A survey conducted by Forrester [10] in late 2012 showed that employees across 17 countries would choose `Android` or `iOS` as the smartphone platform of choice on their primary work device. The emergence of the 'Bring Your Own Device' (`BYOD`) trend is putting additional pressure on companies to cater for the influx of personal smartphones connected to corporate networks since ensuring that business intelligence is not leaked to competitors is of utmost priority to employers.

While employers generally support the concept of `BYOD` [38], they must ensure that appropriate privacy and security settings are maintained on the devices in order to prevent leakage of sensitive company information. However, many mobile phone companies offer convenience and 'rewards' in return for allowing the Operating System (OS) to track smartphones. For instance, if the smartphone can identify a user's location, the user can be sent information about a favorite restaurant which is nearby [27]. Moreover, offers of a free meal may be sent to the user via advertisements on the device. On the other hand, several existing publications (e.g. [6, 41, 26, 12]) have been able to show that sensitive information such as device ID and user location are often leaked via advertising libraries.

Mobile phones provide tracking services for several reasons, including those just mentioned: location identification offers assistance with driving (or walking) to a target destination and also indicates facilities along the way.

In this paper, we test two types of tracking features for smartphones equipped with Google or Apple OS. On both of these platforms, users are given the options to control the following two tracking services: (i) *Location Services* and (ii) *Advertising*. In (i), smartphone owners can either turn on or turn off location tracking to prevent installed applications from discovering their physical locations. As for advertising, users are allowed to either turn on or 'limit' tracking by advertising libraries embedded in applications. To our knowledge, these default options that are provided by the devices have not been tested in any prior work. We test them in this paper by addressing the following questions:

- Do the `On/Off` settings for *Location Services* operate

MoMM 2013, Vienna, Austria

as described?

- Do the *Advertising* setting options operate as described?
- What kind of information is disclosed by the smartphone when location or advertising are tracking?

For the purpose of our study, we conducted experiments on a dataset of 102 `Android` and 102 `iOS` applications downloaded from their respective official application market [16, 35]. In addition, in each case, half of the applications were free and the other half cost between $0.99 and $3.15. We used two smartphones: (i) Motorola Razr, running on `Android` 4.0 and (ii) iPhone 4, running on `iOS` 6.1. We set up *Mallory* [40], a monitoring platform, within a Virtual Machine (VM) to capture live traffic to and from each smartphone and external servers. All traffic was then recorded in an `SQL` database and exported outside the VM for further analysis.

Our contributions are as follows:

1. We develop an experimental setup that can be used to monitor real-time traffic for `Android` and `iOS` applications.
2. We determine how well the location and advertising setting options work on a sample of applications for the `Android` and `iOS` smartphone platforms.
3. We identify reasons for some of the unexpected tracking discovered in our experiment
4. We provide recommendations on how users can avoid man-in-the-middle attacks due to lack of proper `SSL` security measures present in smartphone applications.

The rest of the paper is organized as follows: In Section 2, we briefly describe some of the existing work related to tracking services on `Android` and `iOS`. In Section 3, we provide a general background on tracking services within the context of our work. Section 4 includes a detailed explanation of our experimental work, followed by an analysis of our empirical results in Section 5. In Section 6, we discuss our findings and we provide our recommendations and future research directions in Section 7.

2. RELATED WORK

Both `Android` and `iOS` have their own official application market which is hosted and maintained by Google and Apple, respectively. The application market ecosystem relies on application developers to register and upload their applications to make them accessible for smartphone users. Importantly, the developers are required to follow the rules imposed by the smartphone platforms to ensure that they do not misuse a user's private information.

In this section, we present some of the existing work on tracking services that have been proposed for each of our chosen platforms.

2.1 Android

The `Android` platform consists of four layers, which are the *Applications*, *Application Framework*, *Libraries* and the *OS*. An `Android` application resides on the *Applications* layer and is made up of (i) *Class*, (ii) *Resources* and (iii) *AndroidManifest.xml* files. The *Class* file contains the application source code written in Java and the *Resources* file stores the multimedia files. The *AndroidManifest.xml* file lists the permissions that are declared by the application developer; these permissions are presented to the user during the application installation process.

Google applies a permission-based model [18] as a measure to restrict access to privileged system resources and a user's private information for `Android` applications. As such, the user has to grant access to all permissions requested by the application in order for it to be successfully installed. Consequently, any advertising libraries embedded in the application receive the same privileges as the application that requested the permissions.

Pearce et al. [28] proposed a framework that can separate an advertising library from its main application. In order to do so, a new advertising Application Programming Interface (`API`) and two additional permissions were introduced. The authors applied a method known as *privilege separation*, which involved extracting the advertising component from the main functionality component of the application. This ensured that the advertising library did not inherit the same permissions assigned to an application. To evaluate their proposal, Pearce et al. chose a dataset of 964 `Android` applications, out of which 473 applications included advertising libraries. Their empirical results showed that the framework was successfully applied on 456 applications using advertising libraries. As for the remaining 17 applications, the proposed methodology failed to work because these applications included two permissions in their advertising libraries which were not compatible with the proposed methodology.

In [31], Shekhar et al. presented their method for separating applications and advertisements in the `Android` platform. The authors proposed a framework that can take as input an application with embedded libraries and rewrite it so that the main functionality of the application and the advertising libraries run as different processes. Shekhar et al. also verified that, in the rewritten version of the application, all the permissions requested by the application were indeed required for the application to function properly.

The authors of [20], [23], [33] and [45] investigated the privacy implications of having third-party advertising libraries embedded in applications. Their work is described below.

Stevens et al. [33] performed a thorough analysis of third-party advertising libraries to have a better understanding of whether these libraries are unnecessarily accessing private information stored on a user's smartphone. Additionally, the authors presented several vulnerabilities that attackers can exploit whilst being connected on the same network as the victim.

Grace et al. [20] presented a thorough study on the different types of private information that can be accessed by advertising libraries. They observed that some third-party advertising libraries used unsafe mechanisms to retrieve and execute code from the Internet. Consequently, this behavior rendered the user's private information vulnerable to external attacks that can be carried out via the Internet.

The authors of [23] and [45] proposed some techniques that can be used to predict if an application will leak private information when installed on the user's device. In their work, they presented their understanding of the different avenues through which sensitive information can be leaked to external entities. While the focus of Mann and

Starostin [23] was primarily on extending their knowledge of the Dalvik bytecode [34], Zhao and Osorio [45] explored the implications of personal smartphones within a corporate environment and how the use of vulnerable applications can impede business operations.

In their work, Han et al. [21] and Micinski et al. [24] addressed the concerns surrounding illegal use of a user's location information. Han et al. demonstrated that a user's location can be inferred by simply monitoring the accelerometers found in smartphones. Furthermore, they claimed that no permissions are required in order to access the information recorded by accelerometers; hence, making it difficult to detect the theft of this information. Micinski et al. [24] took a proactive approach towards sharing a user's location with third-parties. The authors investigated the possibility of truncating any location-relevant information that is sent to external servers, without compromising the user's experience while using the application.

2.2 iOS

The `iOS` platform consists of four principle layers, which are the (i) *Application*, (ii) *Media*, (iii) *Core Services* and (iv) *OS and Device Drivers* layers. Upon installation, an `iOS` application will reside on the first layer of the smartphone platform. Apple does not make use of a permission system to request access to a user's private information as each application is vetted before it is uploaded on the application market. Nevertheless, Apple does not always identify each instance of compromise of user data by tracking services. We give some examples of this in the next paragraphs.

In his work [32], Smith described how the *Unique Device Identifier* (`UDID`) and location information of a device running on the `iOS` platform can be captured by tracking services, unbeknown to the user. The experimental work involved monitoring traffic to and from the test device, using Wireshark [44], to determine whether any personally identifiable information was sent out to third-parties.

Egele et al. [9] took a proactive approach towards detection of personal information leaks in iPhone applications and proposed a tool, *PiOS*, which can statically detect any data leaks in an application. The authors explained that *PiOS* can automatically generate control flow graphs which are then used to identify any information leaks to external sources. To evaluate their tool, a dataset of 1400 (including both free and paid) iPhone applications was used. Egele et al. observed that the two most common information leaks were the `UDID` (leaked by 195 applications) and location information (leaked by 36 applications).

Agarwal and Hall [1] designed a framework that allowed `iOS` users to mask their private information thus, permitting them to send anonymized data to third-parties. The authors used a technique referred to as *crowdsourcing* to allow real users to contribute towards building a knowledge base of privacy recommendations. The privacy recommendations were then tested on a dataset of $10,000$ most popular applications. The proposed framework was able to successfully provide recommendations resulting in improved privacy for 97.1% of the applications in the experimental dataset.

3. BACKGROUND

As our main focus is on *Location Services* and *Advertising* for `Android` and `iOS`, in this section, we provide a general background on the key terms used in the the context of our work.

3.1 Tracking Services on Android

Location Services and *Advertising* on `Android` 4.0 can be accessed as described in [19] and [15], respectively.

For *Location Services*, users have the option to grant or completely block any access to location-related information. As a result, users expect that the installed applications will only use location information when *Location Services* is on. In order to verify whether this is indeed the case, we monitor access by installed applications to the following information: Media Access Control (MAC) address, Internet Protocol (IP) address and Global Positioning System (GPS). MAC address refers to a unique 12-character identifier assigned to a Wi-Fi enabled device. As explained by Goodin in [13], the MAC address can be used to reveal a user's "precise location". As for the IP address and GPS coordinates, they can both be used to track a user's physical location, as described in [5] and [17].

On the other hand, tracking via *Advertising* on `Android` devices cannot be fully disabled. By default, advertising libraries constantly track a user's behaviour and carry out targeted advertising. However, with the introduction of the 'Google Settings' application [15], after downloading an application from the official market, a user can either opt in or out of having targeted advertisements delivered through AdMob [14] - which is owned by Google. Ideally, when a user chooses to opt out, the expected outcome is not to receive advertisements delivered by AdMob libraries. In order to monitor advertising tracking, we observe access to the following information: International Mobile Station Equipment Identity (IMEI), serial number of the `Android` device (DeviceID) and the 64-bit number generated on the device's first boot (AndroidID). These are three distinct identifiers that can be used to track a device and subsequently, profile a user's behaviour. In Table 1, we list the keyword information that we will use to monitor tracking by `Android` applications.

Table 1: Keywords for Android

Location Services	Advertising
(i) MAC address	(i) IMEI
(ii) IP address	(ii) DeviceID
(iii) GPS	(iii) AndroidID

3.2 Tracking Services on iOS

Location Services on `iOS` 6.1 can be turned on/off as described in [36]. We apply the same rationale presented in 3.1 and use a list of keywords to identify any illegitimate access to location-related information.

With versions of `iOS` prior to 6.1, advertising companies were able to capture static ID including the unique device identifier (UDID) in order to identify and target the device with advertising. However, with the introduction of `iOS` 6.1, Apple began to discourage application developers from capturing and using the UDID for advertising purposes. As an alternative, Apple offered a randomly generated 'Identifier For Advertising' (IFA) [7] allowing targeted advertising without the use of the UDID.

Since the IFA is a randomly generated value, we cannot monitor it during real-time communication between smartphone and external servers. On the other hand, the situation

offered us the opportunity to check if application developers had indeed switched to use of the IFA, forgoing UDID capture. Thus, we monitor the static ID on the `iOS` version 6.1 device, serial number and UDID, both to test the tracking 'on/off' features and to see whether UDID is still being used.

Table 2 presents the keywords used in the context of our work in order to monitor tracking on the `iOS` platform.

Table 2: Keywords for iOS

Location Services	Advertising
(i) MAC address (ii) IP address (iii) GPS	(i) Serial number (ii) UDID

4. EXPERIMENTAL WORK

The aim of our work is to verify the completeness of tracking services on the `Android` and `iOS` platforms. In this section, we give further explanation about our dataset and describe our methodology.

4.1 Dataset Collection

In this paper, we focus only on the two most popular platforms, `Android` (version 4.0) and `iOS` (version 6.1), that are currently dominating the smartphone industry. In order to ensure that the experiment is consistent, we collected for our dataset only those applications that are from the official markets [16, 35] and which were developed by the same application developer for both platforms. It should be noted that since application developer profiles cannot be publicly accessed on the application markets, we manually checked the developer's information for each application (both from `Android` and `iOS`) before including it in our dataset.

We also restricted our collection to applications from the following four categories as they have a large user base and the ramifications of information leaks are high: `Games`, `Social Networking`, `Finance` and `Business`. For each category, we then selected a set of the top free and the top paid applications. Our final dataset included 102 `Android` and 102 `iOS` applications from four categories.

4.2 Experimental Setup

After completing the dataset collection, we proceeded to set up the experimental environment. We used a Motorola Razr and an iPhone 4 to test our dataset and a traffic sniffing tool, *Mallory*, to capture communication between device and server and vice versa. *Mallory* [2] is an open-source mobile application assessment tool developed by the security firm, Intrepidus Group [39]. We chose this tool as it is capable of intercepting Secure Socket Layer (SSL) traffic and acts as a Man-in-The-Middle (MiTM) proxy, hence facilitating the capture of real-time communication. It is also worth mentioning that although there exist other privacy monitoring and sniffing tools [4, 44, 37], from a research standpoint, we found that *Mallory* provided good granular control over the usage of the tool and manageability of the data captured.

We set up *Mallory* within a VM, running on Linux OS and allowed the VM to connect to the Internet. In order to relay traffic between the smartphone and *Mallory*, we established a communication channel using the Point-to-Point Tunneling Protocol [25]. As the smartphone and *Mallory* both shared the same Internet connection, any Internet-based communication on the smartphone was captured by *Mallory*, as illustrated in Figure 1. All traffic, including Client to Server (C2S) and Server to Client (S2C), was recorded in a database which was later exported for further analysis.

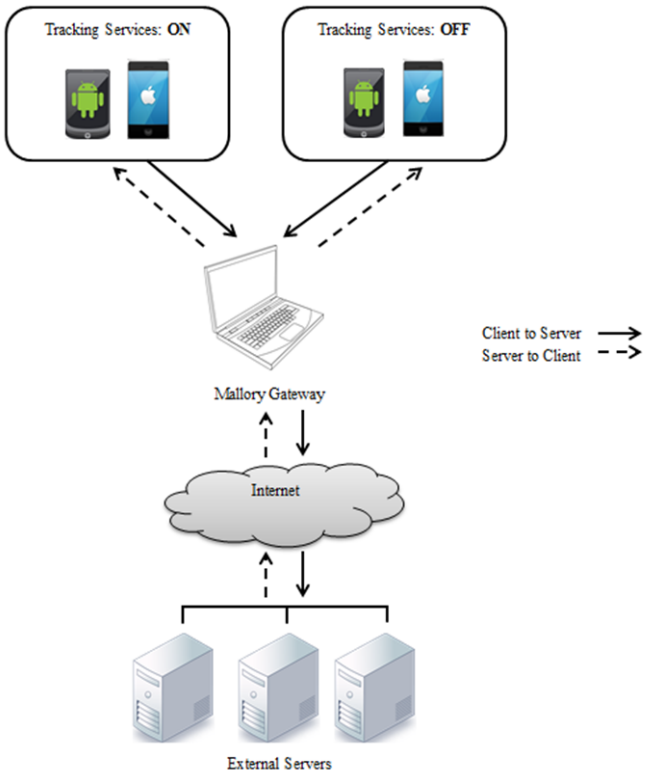

Figure 1: Experimental Setup

In order to monitor private information sent out via tracking services, we ignored any S2C traffic and focused only on C2S communications in our experiment. We started by allowing the smartphones to use tracking services - that is, *Location Services* and *Advertising* were both switched on. We then proceeded to do a clean install of each application and executed it for a period of two minutes. Once the execution time was over, we stopped *Mallory* from recording further traffic and killed the application process on the smartphone. We repeated the same steps for each of the 102 `Android` and 102 `iOS` applications in our dataset on their respective smartphones.

In the second part of our experiment, we disabled *Location Services* and limited *Advertising* tracking services on both the `Android` and `iOS` smartphones, as explained in Section 3. We then repeated similar operations as in the first part of the experiment. We installed the 102 `Android` and 102 `iOS` applications one by one on their respective devices, executed the applications for two minutes and recorded only C2S traffic.

Once the experiment was concluded, we exported all traffic logs outside the VM and searched for the keywords listed in Tables 1 and 2.

4.3 Experimental results

For each category of application and each keyword, we generated a table to record whether or not keyword information had been accessed by the applications in our dataset during execution time. This overall information is presented in Figures 2 and 4, and in Table 3

In Table 3, we have presented the resulting data separated by OS, and have divided each OS column into two parts, *Location Services* and *Advertising*. Under *Location Services*, as explained in Section 3, users have the option to either turn on or turn off location tracking. When *Location Services* is 'ON', we have the following outcomes: If the application accesses information about the keyword(s) mentioned in the first column of Tables 1 and 2, then we place a tick (✓); otherwise a cross (✕) is recorded. On the other hand, when *Location Services* is 'OFF', we expect that applications are not able to access any location-related information. Hence, a ✓refers to an application that did not use any location information while ✕ indicates that location information was accessed.

Similarly, for *Advertising*, we have the 'ON' and 'Limited' options - as presented in Table 3. When applications are allowed to track via advertisements, i.e. *Advertising* is 'ON', we place a ✓if the application accessed information about the keyword(s) listed in the second column of Tables 1 and 2; otherwise a ✕ is recorded. However, when *Advertising* is 'Limited', a ✓means that advertising libraries did not track smartphone users based on our pre-defined keyword(s); whilst a ✕ refers to the contrary.

Of the four categories, 6 out of the 7 `Android` applications that leaked the DeviceID and 7 out of 9 iOS applications that leaked the MAC address belonged to the `Games` category. Thus, we felt that it would be worthwhile to analyse this category carefully and have done so in Figures 3 and 5.

5. ANALYSIS OF RESULTS

We examined a dataset of 102 `Android` and 102 `iOS` applications with the aim to test if tracking services embedded on the two smartphone platforms operate as claimed. In the first part of this section, we elaborate on the information that was leaked when tracking was turned off and in the second part, we give an overview of our experimental dataset comparing results when tracking is turned on and off.

5.1 Information Leaks for Tracking Off

Figures 2 - 5 present the number of applications that leaked information related to the list of keywords in Section 3. The `y-axis` denotes the total number of applications and the `x-axis` refers to the types of information that are leaked. In order to facilitate the interpretation of our experimental results, we combine the items on the `x-axis` into two categories: (i) Dynamic and (ii) Static information. Dynamic information refers to any meaningful data that are liable to changes due to the user's physical surrounding. For both the `Android` and `iOS` platforms, we consider IP address and GPS as dynamic information. Conversely, Static information refers to those keywords that remain unchanged throughout the lifetime of the smartphone. In the case of `Android`, we have MAC address, IMEI, DeviceID and AndroidID; for `iOS` - MAC address and UDID.

From our empirical results, we found that for the set of `Android` applications, the most leaked static information is IMEI (75% of 102 applications) and AndroidID (27% of 102 applications) - as shown in Figure 2. Figure 3 indicates that 6 out of the 7 `Android` applications leaking the DeviceID, 22 of the 76 applications leaking the IMEI and 14 of the 28 applications leaking the AndroidID were from `Games`.

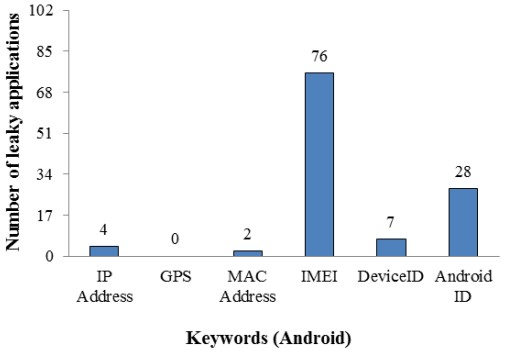

Figure 2: Leaky Android Applications with Tracking OFF (all 4 categories)

As for the dynamic information, only 5% of the 102 `Android` applications leaked the IP address and the GPS information was never read by the applications in our dataset. We also noted that only the applications belonging to the `Social Networking` category did not access any dynamic information.

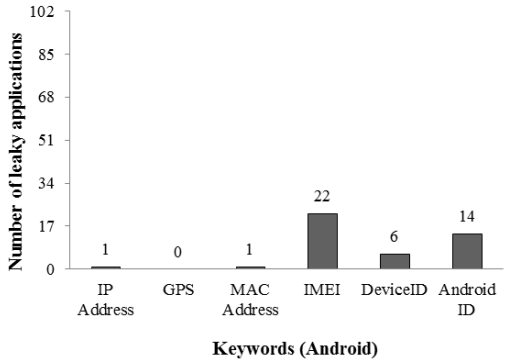

Figure 3: Leaky Android `Games` Applications with Tracking OFF

In Figure 4, it can be observed that static information was the most leaked data amongst the `iOS` applications. More precisely, 9% and 12% of the 102 applications sent out the MAC address and UDID, respectively. It is also worth noting that 80% of the applications that leaked the MAC address belonged to the `Games` category - as shown in Figure 5. Although this particular application category did not leak any dynamic information, we found that 11% of the applications from the remaining three categories, accessed the IP address and GPS information. While only the applications from the `Business` and `Finance` category leaked the IP address, the GPS information was sent out by applications from the `Business` and `Social Networking` category. In fact, the `Business` category is the only set of applications

Table 3: Tracking Services in Android and iOS Applications (Category: `Games`)

Android				Application Name	iOS			
Location Services		*Advertising*			*Location Services*		*Advertising*	
ON	OFF	ON	Limited		ON	OFF	ON	Limited
×	✓	✓	×	1. Sonic Dash	✓	×	✓	×
×	✓	✓	×	2. Pic Combo	×	✓	×	✓
×	✓	✓	×	3. Temple Run	×	✓	×	✓
×	✓	×	✓	4. Candy Crush	×	✓	×	✓
×	✓	✓	×	5. 4 Pics 1 Word	✓	×	×	✓
✓	×	✓	×	6. Megapolis	✓	×	✓	×
×	✓	✓	×	7. Doodle Jump	×	✓	✓	×
×	✓	✓	×	8. Subway Surfers	✓	✓	×	✓
×	✓	✓	×	9. Nimble Quest	✓	×	×	✓
×	✓	✓	×	10. UNO & Friends	×	✓	×	✓
×	✓	✓	×	11. Royal Revolt	✓	×	×	✓
×	✓	✓	×	12. Lucky Wheel for Friends	×	✓	×	✓
✓	✓	✓	×	13. Tiny Troopers 2	×	✓	×	✓
×	✓	✓	×	14. Smash it	×	✓	×	✓
×	✓	✓	×	15. Where's my Water	×	✓	×	✓
×	✓	✓	×	16. Temple Run: OZ	×	✓	×	✓
×	✓	✓	×	17. Angry Birds Star Wars	×	✓	×	✓
×	✓	✓	×	18. Plants vs Zombies	×	✓	✓	×
×	✓	✓	×	19. Wreck it Ralph	×	✓	×	✓
×	✓	✓	×	20. Vector	✓	×	×	✓
×	✓	✓	×	21. Where's my Perry	×	✓	×	✓
×	✓	✓	×	22. Bad Piggies	×	✓	×	✓
×	✓	✓	×	23. Backflip Madness	×	✓	×	✓
×	✓	✓	×	24. Slingshot Racing	×	✓	×	✓
×	✓	✓	×	25. Plague	×	✓	×	✓
×	✓	✓	×	26. Tetris	×	✓	×	✓

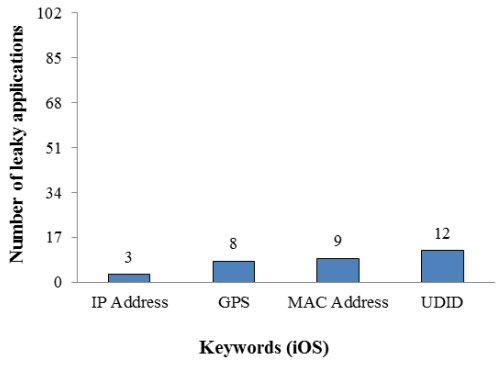

Figure 4: Leaky iOS Applications with Tracking OFF (all 4 categories)

that leaked all 4 types of information that we monitored on the `iOS` platform.

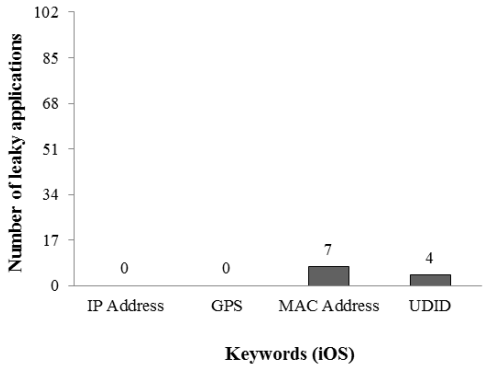

Figure 5: Leaky iOS `Games` Applications with Tracking OFF

5.2 Tracking On v/s Tracking Off

We present in Tables 4 and 5 the keyword information captured for `Android` and `iOS` applications. We compare the number of applications that sent out private information when tracking was turned on and off. It should be emphasized that the comparisons are based on a small dataset.

In Table 4, we observed that the IMEI is the most leaked

Table 4: Tracking On and Off in Android Applications

Keyword	Business		Finance		Games		Social Networking	
	ON	*OFF*	*ON*	*OFF*	*ON*	*OFF*	*ON*	*OFF*
IP Address	2	2	1	1	1	1	0	0
GPS	1	0	0	0	0	0	0	0
MAC Address	2	1	0	0	1	1	0	0
IMEI	19	14	10	18	18	22	18	22
DeviceID	0	0	0	0	11	6	0	1
AndroidID	1	3	1	5	14	14	8	6

Table 5: Tracking On and Off in iOS Applications

Keyword	Business		Finance		Games		Social Networking	
	ON	*OFF*	*ON*	*OFF*	*ON*	*OFF*	*ON*	*OFF*
IP Address	2	2	1	1	0	0	0	0
GPS	6	6	0	0	0	0	2	2
MAC Address	3	2	0	0	7	7	0	0
UDID	3	3	2	3	4	4	1	2

information by `Android` applications, irrespective of whether tracking is turned on or off. In fact, with the exception of the applications from the `Business` category, the rest of the dataset leaked the IMEI more often when tracking was turned off. Similarly, for the remaining list of keywords, there was not much difference in the number of applications that leaked a particular keyword when tracking was enabled and disabled.

In Table 5, UDID was the only keyword that consistently leaked across all four categories. We observed that for both tracking on and off, the applications from the `Business` and `Games` categories leaked the same number of times and there was only a slight discrepancy for the applications in the remaining two categories.

6. DISCUSSION

Our experimental results provide insights into the operability of the tracking services on the `Android` and `iOS` platforms. In this section, we present these insights along with some ideas for future research directions.

Dynamic Information. As described in Section 5, dynamic information used by tracking services, is affected by the user's physical location. Traditionally, advertisements are displayed through in-application banners. However, we observed that in one particular `Games` application, '4 Pics 1 Word', although *Location Services* was 'OFF', the device's IP address was sent out to the advertiser in order to display in-application high-definition videos - which is simply another form of advertising. One could argue that such leakage of dynamic information, which changes regularly, need not be of concern; however, as explained by Warren in [43], whilst a single piece of dynamic information is harmless to the user's privacy, a collection of such information can reveal the location history of the user and could potentially be misused.

Static Information. `Android`'s DeviceID and `iOS`'s UDID were the most leaked static information in smartphone applications. Generally, smartphone users tend to be cautious when downloading applications as they do not want to install applications that would compromise their device and private information. When a user agrees to grant access to the requested permissions by an `Android` application, the underlying assumption is that only the application in question will be given access to restricted resources, and these resources will not be sent to other parties. However, in our experiment, we observed that over 80% of the `Android` applications sent the IMEI information to the smartphone vendor's server without the user's knowledge.

IFA. We mentioned in Section 3.2 that in `iOS` version 6.1, Apple has tried to entice application developers away from the capture of UDID by offering an alternative randomly generated value (IFA). Table 5 indicates that MAC address and UDID are captured at about the same rate, averaged over four categories. Since application updates are pushed out to `iOS` devices, without further information, we can only conclude that developers do not feel it is worthwhile to make IFA-associated changes to their applications.

Tracking Services & SSL. Although the main focus of our work is to verify the tracking services settings embedded within `Android` and `iOS`, we should point out that information leaks do not necessarily only take place via such services. As observed by Fahl et al. [11], users who install compromised SSL certificates are in fact providing a backdoor to attackers and subsequently allowing them to sniff any sensitive information that is sent to and from the smartphone and external servers. In our experiment, we noticed that nearly 95% of our 204 applications sent email addresses and passwords in clear text, which is an indication that appropriate validation measures for SSL certificates are lacking. In some cases, applications did display a warning message to the user who can easily ignore it and proceed to communicate via the insecure channel.

7. RECOMMENDATIONS

There is no doubt that advertising is a fundamental pillar of the smartphone business model as it allows application developers to offer their application free of charge to the public whilst still earning revenue from in-application advertisements for their work. Moreover, although location services are primarily used for navigation purposes, advertising companies do certainly exploit this functionality to increase their revenue. Thus, advertising is unlikely to dis-

appear from smartphones in the near future. Based on our empirical results, we provide recommendations to the following three main stakeholders.

1. *Novice Smartphone Users.*
   We strongly recommend this category of users to download applications only from the official markets as they are less likely to be malicious. Moreover, although one cannot guarantee that all the applications found on official markets are definitely clean, there is always a chance for any malicious applications to be deleted from the market when reported to the designated authorities.

2. *Smartphone Manufacturers.*
   Recently, Google [30] announced that they will no longer allow application developers to upload applications that are designed to block advertisements as these applications do not conform to rules and regulations imposed by Google. Furthermore, from our experiment, we also found that the limited advertising tracking option introduced by Apple is not effective in preventing unauthorized access to the UDID information.

   In order to address the shortcomings in tracking services on the `Android` and `iOS` platforms, we believe that the device manufacturers can help to alleviate the issue of tracking by providing users with pre-installed applications so that they have more control of their private information instead of relying on the smartphone OS.

3. *Academia/Industry.*
   Lastly, we propose that researchers from academia and industry within the field come together to form an open-source research community. The community could develop open-source applications that will help to compensate for the security vulnerabilities found in existing applications offered by the official application markets.

   In order to encourage all major industry players to participate, similar testing of other major smartphone OS such as `Windows Phone 8` and `Blackberry` would be a good first step. Our team is already involved in such work.

8. ACKNOWLEDGMENTS

The authors would like to thank Professor Yuliang Zheng for his valuable feedback. We also thank our industry partner, NBN Co., for the financial support.

9. REFERENCES

[1] Y. Agarwal and M. Hall. ProtectMyPrivacy: Detecting and Mitigating Privacy Leaks on iOS Devices Using Crowdsourcing. In *Proceedings of the 11th International Conference on Mobile Systems, Applications and Services (MobiSys 2013)*, pages 1–13, Taipei, Taiwan, June 2013.

[2] J. Allen and R. Umadas. Network Stream Debugging with Mallory. Technical report, Intrepidus Group, 2010.

[3] Android. `http://www.android.com/`.

[4] Bitdefender. Introducing the new Clueful, 2013. `http://www.bitdefender.com.au/solutions/clueful.html`.

[5] J. Brock. Geo-IP location targeting: When is consent require? PrivacyChoice, January 2013. `www.blog.privacychoice.org/2012/01/23/geo-ip-location-targetingwhen-is-consent-required/`.

[6] S. Dhar and U. Varshney. Challenges and business models for mobile location-based services and advertising. *Communications of the ACM*, 54(5):121–128, 2011.

[7] D. Dilger. Apple adds new "Limit Ad Tracking" feature to iOS 6. Apple Insider, September 2012. `www.appleinsider.com/articles/12/09/13/apple_adds_new_limit_ad_tracking_feature_to_ios_6`.

[8] F. Donovan. Smartphone display shipments increase 120 percent in 2 years. FierceMobileIT, June 2013. `http://www.fiercemobileit.com/story/smartphone-display-shipments-increase-120-percent-2-years-says-npd/2013-06-21`.

[9] M. Egele, C. Kruegel, E. Kirda, and G. Vigna. PiOS: Detecting Privacy Leaks in iOS Applications. In *Proceedings of the 18th Annual Network & Distributed System Security Symposium (NDSS 2011)*, pages 1–15, San Diego, CA, February 2011.

[10] P. Elmer-DeWitt. BYOD: What do mobile information workers want? CNN Money, February 2013. `http://tech.fortune.cnn.com/2013/02/04/apple-vs-microsoft-enterprise-forrester/`.

[11] S. Fahl, M. Harbach, T. Muders, L. Baumgärtner, B. Freisleben, and M. Smith. Why Eve and Mallory love Android: an analysis of Android SSL (in)security. In *Proceedings of the 2012 ACM conference on Computer and communications security (CCS 2012)*, pages 50–61, North Carolina, USA, October 2012.

[12] E. Fife and J. Orjuela. The privacy calculus: Mobile apps and user perceptions of privacy and security. *International Journal of Engineering Business Management*, 5(6):7, 2012.

[13] D. Goodin. Google location tracking can invade privacy, hackers say. The Register, April 2011. `http://www.theregister.co.uk/2011/04/22/google_android_privacy_concerns/`.

[14] Google. Admob. `http://www.google.com/ads/admob/`.

[15] Google. Advertising. `http://www.google.com.au/policies/technologies/ads/`.

[16] Google. Google play. `https://play.google.com`.

[17] Google. Location Strategies. `http://developer.android.com/guide/topics/location/strategies.html`.

[18] Google. Android Security - Using Permissions, 2012. `http://developer.android.com/guide/topics/security/permissions.html`.

[19] Google. Enable location services, 2013. `https://support.google.com/gmm/answer/1646140?hl=en`.

[20] M. C. Grace, W. Zhou, X. Jiang, and A. Sadeghi. Unsafe exposure analysis of mobile in-app advertisements. In *Proceedings of the 5th ACM conference on Security and Privacy in Wireless and*

Mobile Networks (WISEC 2012), pages 101–112, Arizona, USA, April 2012.

[21] J. Han, E. Owusu, L. Nguyen, A. Perrig, and J. Zhang. ACComplice: Location inference using accelerometers on smartphones. In *Proceedings of the 4th International Conference on Communication Systems and Networks (COMSNETS 2012)*, pages 1–9, Bangalore, India, January 2012.

[22] iOS. http://www.apple.com/au/ios/what-is/.

[23] C. Mann and A. Starostin. A framework for static detection of privacy leaks in android applications. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing (SAC 2012)*, pages 1457–1462, Trento, Italy, March 2012.

[24] K. Micinski, P. Phelps, and J. S. Foster. An Empirical Study of Location Truncation on Android. In *Proceedings of the 2013 Mobile Security Technologies Conference (MoST 2013)*, pages 1–10, San Francisco, CA, May 2013.

[25] Microsoft. Point-To-Point Tunneling Protocol (PPTP). http://technet.microsoft.com/en-us/library/cc751470.aspx.

[26] V. Moonsamy, M. Alazab, and L. Batten. Towards an understanding of the impact of advertising on data leaks. *International Journal of Security and Networks*, 7(3):181–193, 2012.

[27] S. Murphy. More Smartphone Owners Use Location-Based Products. Mashable, May 2012. http://mashable.com/2012/05/11/location-based-services-study/.

[28] P. Pearce, A. Felt, G. Nunez, and D. Wagner. AdDroid: Privilege Separation for Applications and Advertisers in Android. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security (ASIA CCS 2012)*, pages 1–11, Seoul, Korea, May 2012.

[29] S. Perez. ABI: With 58% Market Share, Android Will Top iOS In Smartphone App Downloads This Year, But Apple Will Win On Tablets. TechCrunch, March 2013. www.techcrunch.com/2013/03/04/abi-with-58-market-share-android-will-top-ios-in-smartphone-app-downloads-this-year-but-apple-will-win-on-tablets/.

[30] K. Russel. Check out what data Motorola has been collecting about some of its users. Business Insider, July 2013. http://au.businessinsider.com/motorola-has-been-collecting-user-data-2013-7.

[31] S. Shekhar, M. Dietz, and D. Wallach. Adsplit: Separating smartphone advertising from applications. In *Proceedings of the 20th USENIX Security Symposium (USENIX Security 2012)*, pages 1–15, Bellevue, USA, August 2012.

[32] E. Smith. iPhone Applications & Privacy Issues: An Analysis of Application Transmission of iPhone Unique Device Identifiers (UDIDs). Technical report, Bucknell University, 2010.

[33] R. Stevens, C. Gibler, J. Crussell, J. Erickson, and H. Chen. Investigating User Privacy in Android Ad Libraries. In *Proceedings of the 2012 IEEE Mobile Security Technologies (MoST 2012)*, pages 1–10, California, USA, May 2012.

[34] Android Open Source Project. Bytecode for the dalvik virtual machine, 2012. http://source.android.com/tech/dalvik/dalvik-bytecode.html.

[35] Apple Inc. http://store.apple.com/au.

[36] Apple Inc. iOS 6: Understanding Location Services, April 2013. http://support.apple.com/kb/HT5467.

[37] Burp Suite. http://www.portswigger.net/burp/.

[38] Gartner, Inc. Gartner Predicts by 2017, Half of Employers will Require Employees to Supply Their Own Device for Work Purposes. Gartner, May 2013. http://www.gartner.com/newsroom/id/2466615.

[39] Intrepidus Group. http://intrepidusgroup.com/index.php.

[40] Intrepidus Group. Mallory: Transparent TCP and UDP Proxy. http://intrepidusgroup.com/insight/mallory/.

[41] J. Valentino-DeVries. What Your iPhone Knows About You. The Wall Street Journal, April 2011. http://blogs.wsj.com/digits/2011/04/20/what-your-iphone-knows-about-you/.

[42] M. Warman. Android Apps to Ovetake Apple. The Telegraph, June 2013. http://www.telegraph.co.uk/technology/news/10095401/Android-apps-to-overtake-Apple.html.

[43] C. Warren. Your iPhone Is Tracking Your Location History. Mashable, April 2011. http://mashable.com/2011/04/20/iphone-location-history/.

[44] Wireshark. http://www.wireshark.org/.

[45] Z. Zhao and F. Osono. TrustDroid: Preventing the use of Smartphones for information leaking in corporate networks through the use of static analysis taint tracking. In *Proceedings of the 7th International Conference on Malicious and Unwanted Software (MALWARE 2012)*, pages 135–143, Puerto Rico, USA, October 2012.