Comparing Privacy Control Methods for Smartphone Platforms

Mohammed Alhamed Khalid Amir Mansoor Omari Wei Le

B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, USA
{ maa1515, kxa2185, mmo2912, wei.le }@rit.edu

Abstract—**Nowadays, many important applications are performed through mobile phones. It is essential to ensure that users' private information is not leaked through those applications. In this paper, we perform a comparison on privacy control methods implemented on the Android and iOS platforms based on the Bellotti and Sellen's framework. The comparison helps understand the discrepancies existent between the users' expectations for privacy and the privacy control methods currently implemented in Android and iOS. To better address users' privacy concerns, we propose a programming model for platform designers to improve privacy. Our initial study on 60 privacy bugs show that using the proposed programming models, 14 Android and 5 iOS privacy bugs can be eliminated.**

***Index Terms*—privacy, privacy control methods, smartphone, mobile, iOS, Android.**

I. INTRODUCTION

A large number of applications like social networking and travel advising applications enjoy their access to resources from smartphones that enable them to make context-aware intelligent decisions. However, these applications have always been a target of criticism for privacy issues. In 2011 a bug was discovered in iOS devices, which was storing users' one year location tracking data in an unencrypted file [18]. An example of privacy issue discovered with Android is that the phone was sending users' location data to Google [19].

Researchers have been spending a good amount of effort on this topic. The previous literature, which we studied, can be divided into three main categories. The first suggests technical solutions for privacy issues in mobile devices. For example, centralizing the location information releases to be done by one application [3]. The second tries to solve privacy problems by forcing policies such as regulating the usage of smartphones in the company campus [4]. The last category, which this paper fell into, assesses privacy situations in mobile environment from different perspectives [5].

In our research, we perform a preliminary study on identifying and assessing how smartphone platforms control user privacy and whether such methods are problematic and causes information leak. We limit our research on two smartphone platforms: iOS, and Android. Based on the problems found, we propose a programming model derived from *the Bellotti and Sellen's framework* of information flow for improving privacy for both platforms. Our initial results show that the proposed programming models can address about 30% of the privacy bugs found in Android and iOS platforms.

Our research consists of three steps. We first identify a theoretical framework regarding privacy requirements, based on which, we perform a comparison of privacy control methods for the Android and iOS platforms. In this preliminary study, we use location service as an example. Based on the inconsistencies found between the theoretical requirements and the existing implementation of the privacy control methods in Android and iOS, we present an improved design for privacy control methods for both Android and iOS. Finally, we study a set of privacy bugs and determine whether the bugs can be addressed using the suggested programming models.

In this paper, we made the following contributions: 1) we revealed the deficiency of the privacy control methods currently implemented on the Android and iOS platforms; 2) we suggested an improvement for these methods that may help remove many of the privacy bugs; and 3) we provide an initial evidence to show that the programming models proposed based on the Bellotti and Sellen's framework is helpful for improving privacy for both platforms.

Our initial results potentially provide advice for platform designers to better address users' privacy concerns. Although in this study, we focus on only location service and use only the Bellotti and Sellen's framework which provides minimum standards of information privacy for ubiquitous environments, our methodologies of studying privacy problems and solutions on smartphone platforms can be extended by incorporating a more complete set of privacy requirements and studying more privacy bugs found from a more variety of mobile apps. Therefore, our future work include 1) identifying a full set of privacy requirements, 2) comparing existing implementation of privacy control methods with the requirements on a more variety of mobile apps, and 3) performing a more in depth study on existing privacy bugs to understand whether the bugs can be fixed if the platform implements the privacy control methods as indicated in the privacy requirements. We are also interested in exploring the tradeoffs a platform designer makes

.

in balancing between privacy and other software qualities, such as performance and usability.

II. Related work

As we mentioned before, our research fell in the third category, assessing privacy situations in mobile environment, so we focused our literature review in this direction. Gambs et al. conducted their research on privacy issues in geolocation applications [12]. They examine social networking applications such as Facebook, Foursquare and Gowalla that exploit the user's location information. Based on the results, a smartphone operating system has a large role in managing users' privacy.

Kang et al. shows that Android stores the users' sensitive information in a log file, which can be exploited by a malicious user [10]. Security vulnerabilities also affect the user privacy. Xu et al. found security vulnerabilities in 3G smartphones, which can allow a malicious user to gain access to the user's videos [11].

Vidas et al. investigates the way permissions are managed in Android. It states that because the developers do not have an easy way to determine which of the 130 application permissions their application needs, they end up specifying more permissions than they need. This results in the violation of least privilege principle [17].

Smith [15] researched on tracking UDID in iPhones and iPad and how can it be used to track users. Egele [16] used their privacy bug detection tool PiOS and examined 1047 iOS applications for information leak. The tool reveals that an application with the name Gowalla accesses the address book and sends all the names and email addresses to a server. Another application with the name smartphone uploads the history of Safari and photo gallery to Mobile-Spy server (a server used by people to spy on others).

Xiao at el. mentioned that the current privacy models are still in a basic form where features such as how an app uses private data are not enabled [20]. This paper developed more sophisticated privacy model that grantees user awareness about how an app will use private information. However, Bellotti at el's framework is more general and covering other areas that are not covered in Xiao et al's proposal, e.g., where such information is stored and how actions are taken on the information after its release.

Our research identifies the privacy control methods for both Android and iOS. We use the Bellotti and Sellen's framework to analyze these methods and understand the weakness of the existing privacy control methods in smartphone platforms. The identified weakness in the models is correlated with the privacy bugs collected from the smartphone communities.

III. Methodology

A. Identifying a Framework for Comparison

In order to compare and evaluate privacy control methods in the targeted subjects, we need a methodology to help us, and to make it possible for other researcher to replicate it in the future. We use Bellotti and Sellen's framework of information flow [6] to help us identify the interesting points from the privacy point of view.

In the Bellotti and Sellen's framework, there are four main components that we should consider while examining the privacy control for a certain app. These components are :

a) Capture component: it discusses the information being acquired or services being inquired. This component asks "what" questions about the information. For example, what are the information being acquired?

b) Construction component: it cares about the actions that follow the information acquisition and impact it, such as sending, storing, or publishing the information.

c) Accessibility component: it talks about who is going to access the information, and what are the required permission for that. We can ask this question: who is able to access the information after they are acquired?

d) Purposes component: it justifies the capture component by giving a reason why the information is being acquired. By justifying the information acquisition actions, users can decide whether to give permission or not.

B. Comparing and Improving Privacy Control Methods

Privacy control methods are APIs a system provides to the apps to control the access of user's private information. Each OS has many different services that are connected to the user privacy. Our main purpose is to examine the privacy control methods implemented in these services. In our initial study, we focused on user location service. Examples of other possible services could be user contacts information, photos, schedule, and emails.

In order to understand the privacy control methods in iOS and Android location service, we examined both OS APIs and the usage of these APIs through a third party app. We selected Facebook Messenger as the third party application to realize what is described in the API level. During the study, we examined the apps by responding to the four components in the Bellotti and Sellen's framework illustrated above. The identification process is done by one researcher and reviewed by another. Based on the inconsistencies identified between the theoretical framework and the current privacy control methods, we suggest an alternative design of privacy control methods in the systems that potentially can give the user a better control over her/his information.

C. Assessing Privacy Bugs

To further understand the problems of current privacy control methods on the Android and iOS platforms, we conduct bug studies for Android and iOS. In our initial work, we gathered 60 bug posts from platform's bug repositories and community forums [13][14]. We use the key words *information leak*, *privacy problems/issues* to find likely privacy bugs. Among the 60 posts, 34 are iOS and 26 are Android related. We studied 48 posts that are understandable, 24 for iOS and 24 for Android respectively.

We analyzed these posts by reading them and trying to identify the location and cause of the bug. In order to make sure that our data is valid, we obtained the bug posts from the official bug repositories and development communities of iOS and Android. Moreover, we analyzed the bugs by two different members of our team and the conclusions are drawn when the agreement is made.

In the bug study, we focus to understand why the Android and iOS platforms or apps fail to address the users' privacy concerns, how many problems are related to the current privacy control methods implemented on the platforms, and whether we can fix the problems by improving the privacy control methods supported by the platforms.

IV. Results

Here, we provide our results on identifying and analyzing privacy control methods and also on assessing privacy bugs.

A. Privacy control methods

Our results are collected by analyzing Facebook Messenger location service.

1) Privacy Control Methods of Android

In Android, location service is provided by the "`LocationManager`" class, and it works via callbacks. In order to receive location updates, you need to pass a listener, "`LocationListener`", as a parameter to location manager through the "`requestLocationUpdates`" function [8].

In order to acquire user location, you need to ask for user permission. Typically, it is done in a preferences or settings context inside `AndroidManifest.xml` located in the application root folder [7].

In terms of the process and how Android acquires the user permission, Android does it once during the application installation, after it inspects the manifest file [8]. After that, it is the application job to let user know or to ask the user the permission to use the location information. In case if the application doesn't provide this kind of control, the user will not be able to prevent a certain application from using location information, unless turning off the whole location service from Android settings.

To abstract the view and to understand the process of how privacy controls are implemented in Android platform, we create the following code to describe how Android privacy control method is working.

```
01 // get LocationManager instance
02 LocationManager locationManager = (LocationManager)
03             this.getSystemService(Context.LOCATION_SERVICE);
04 // Define a location listener to receive location updates
05 LocationListener locationListener = defineListener();
06
07 // add the location listener to the Location Manager
08 locationManager.requestLocationUpdates(LocationManager.NET
09 WORK_PROVIDER, 0, 0, locationListener);
```

Figure 3: Current Privacy Control method in Android. The code snippet shows how to acquire a location information

As it illustrated in Figure 3, Android doesn't requires the application to ask for user permission. It is done only once during the application installation, and once the permission is issued, the user can't control it later. It assumes that you already give the application the required permission during the installation, and in case you don't, the app will fail during the run time [8].

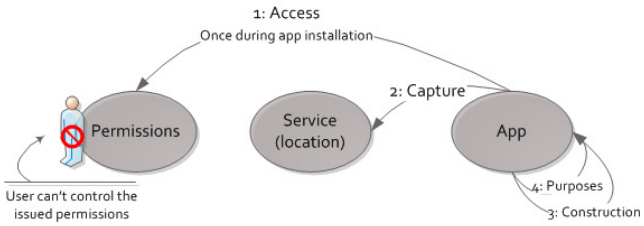

Figure 4: Android Privacy Control Model of location service.

In Figure 4, we compare the privacy control methods in Android with the four components proposed in the Bellotti and Sellen's framework. We found that the Access component for Android is coarse-grained, as the permission is given once at the app installation time. In the Capture component, Android helps the application to get the required information by providing the appropriate APIs. However, Android did not export APIs to support the implementations of the Construction and Purposes components, and it leaves them to the app developers.

Figure 5 shows a suggestion for what the privacy control method supposed to be based on the Bellotti and Sellen's framework.

```
01 // Define the action that will applied on the location
02 // information
03 String actionArray[]={"sendToOurserver","StoreItFor10Days"};
04
05 // get LocationManager instance
06 LocationManager locationManager = (LocationManager)
```

```
07            this.getSystemService(Context.LOCATION_SERVICE);
08
09 // Define a location listener to receive location updates
10 LocationListener locationListener = defineListener();
11
12 // (Purpose Component) tell android about the purpose of
13 // this request
14 locationManager.requestPurpose("to update the map");
15
16 // (construction Component) tell Android about the action
17 // you will apply to the info
18 locationManager.actionWillApplied(actionArray);
19
20 // (Accessibility Component) Check if the user has given the
21 // app the right permission to
22 // get the user location
23 if(locationManager.checkUserPermission(AppID)){
24   // (Capture Component) add the location listener
25   // to the Location Manager
26   locationManager.requestLocationUpdates(LocationManager.NET
27   WORK_PROVIDER, 0, 0, locationListener);
28 }
```

Figure 5: Suggested Privacy Control method for Android. The code snippet shows how to acquire a location information in the suggested method

The improved programming model in Figure 5 addresses all the four components of the Bellotti et al. framework. First, the app should tell Android the purpose of the information request, and the actions that will be applied on that information. Then, it asks for user permission. At this point Android is capable to generate a full informative popup to ask for user permission. Finally, if the user gives the required permission, the app should continue and capture the information, otherwise the app should stop.

2) Privacy Control Methods of iOS

On the iOS platform, location service is provided by the "CLLocationManager" class, and it works in two modes, delegation mode and direct call mode. Delegation mode is used when there is no information about the current location. In this case, you call the "startUpdatingLocation" method after assigning a "delegate" object to the "CLLocationManager" object [9]. Then, your "delegated" object will be called back and updated with the new information about the current location. In the direct call mode, an app can query the "location" property to get the most recently user location.

In order to acquire user location, we need to check if the location service is enabled for our application. We can do that by inquiring the permission component by calling the "locationServicesEnabled" method in the "CLLocationManager" class. If the application doesn't have the permission yet, it still can call the "locationServicesEnabled" method. The system will then ask for the permission from the user each time the app asks for location information. iOS stores user decision about whether or not to give a certain application a permission to access location service. A user can manage these settings for each app.

To abstract the view and get more understanding of how privacy controls are implemented in iOS platform, we create the following code to describe how iOS privacy control method is working.

```
01 // get CLLocationManager instance
02 locationManager = [[CLLocationManager alloc] init];
03
04 // Define a location delegate object to receive location
05 // updates
06 locationManager.delegate = self;
07
08 // (Accessibility Component) check if the user enable the
09 // service
10 if(locationManager.locationServicesEnabled()){
11    // start receiving the location information updates
12    locationManager.startUpdatingLocation();
13 }
```

Figure 6: Current Privacy Control method in iOS. The code snippet shows how to acquire location information.

Figure 6 shows that iOS will ask for user permission every time an application tries to get the user location information. Compared to Android which only asks for user permission once at the app installation time, this mechanism is more fine-grained.

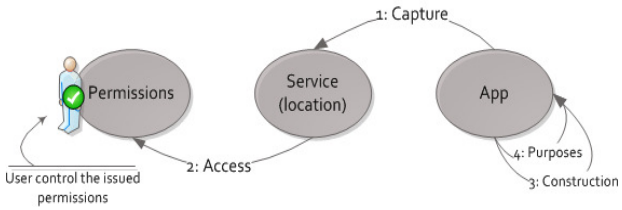

Figure 7: iOS Privacy Control Model of location service.

In Figure 7, we analyze the iOS privacy control methods in the context of four components from the Bellotti et al. framework. The figure shows that in the capture component, the application asks for location information. The access component will then get user permission if it's not acquired before. iOS will asks user permission every time an application tries to get the user location. Based on our study, iOS did not provide APIs for constructing the purposes and construction components in the apps.

Figure 8 shows the suggested method for iOS that handles all the four components of the Bellotti et al. framework.

```
01 // Define the action that will applied on the location
02 // information
03 String actionArray[]={"sendToOurserver","StoreItFor10Days"};
04
05 // get CLLocationManager instance
06 locationManager = [[CLLocationManager alloc] init];
07
08 // Define a location delegate object to receive location
09 // updates
10 locationManager.delegate = self;
11
12 // (Purpose Component) tell iOS about the purpose of
13 // this request
14 locationManager.requestPurpose("to update the map");
15
16 // (construction Component) tell iOS about the action
17 // you will apply to the info
18 locationManager.actionWillApplied(actionArray);
19
```

```
// (Accessibility Component) Check if the service is enabled
// for the current app
if(locationManager.locationServicesEnabled()){

  // start receiving the location information updates
  locationManager.startUpdatingLocation();

}
```

Figure 8: Suggested Privacy Control method for iOS. The code snippet shows how to acquire a location information in the suggested method

3) Comparing Android and iOS Privacy Control Methods

We have shown that for location service, the iOS app asks the user for permissions each time application requests information from the service. This implies that iOS allows users to change permissions without uninstalling the application. On the other hand, Android asks for the user permission only once during application installation. This permission cannot be changed after app installation.

Compared to the Bellotti and Sellen's framework, both Android and iOS have the weakness on not providing the purpose and construction components. For example, iOS tells you that App A needs an access to your location service without telling you the purpose or what will happened to the released information. In such situation the user can't make a clear decision of giving the required permission.

B. Bug assessment results

Among the 48 bugs that we classified, we found that 14 Android and 5 iOS privacy bugs are caused by the weakness of the privacy control methods identified above, shown under Column *Privacy Control Method* in the following table. We manually analyzed these bugs and found that with our suggested programming models, these privacy issues can be fixed.

Platform\Cause	Privacy Control Method	Others
Android	14	10
iOS	5	19

Table 7: Causes of Privacy Bugs

Here, we provide two examples of such bugs. In Android bug# 10340 [14], the user complained that he/she cannot revoke a permission that is already given to an app. As we show in Section IV, in the current Android privacy model, the user is not able to revoke a permission that is given to any app unless the app explicitly provides that for him/her. With the suggested privacy model, the user will be able to revoke the permission even after app is installed. Moreover, he/she will be able to get more information, e.g., why the app needs the information and where it will be stored, to help them decide whether he/she should give the permission.

In iOS bug# 15940113 [13], the user asks why the location service is always running in iPhone. In the current iOS model, the user has no idea why a certain service is running even if the user already gave some apps the permission to access his location. If iOS applies the suggested privacy model, iOS will provide a place where the user will be able to browse all the location service requests and know why these requests are being invoked. Such logging systems help users know why a certain service is running.

In addition to the privacy issues caused by the weak privacy control methods currently implemented in Android and iOS, we also find privacy bugs that are caused by the OS components. See Column *Others*. Studies of these bugs will be our future work.

V. Limitations and Future Work

This work has the following limitations. First, we have compared the privacy control methods currently implemented on the Android and iOS platforms against the Bellotti and Sellen's framework. This framework describes minimum standards of information privacy for ubiquitous environments. To more systematically evaluate privacy control methods for different platforms, we need to identify a more complete set of privacy requirements.

Second, our preliminary study only focuses on location service. The weakness found in the privacy control methods might not be able to be generalized across different types of mobile apps. For example, iOS may ask for user permissions for location, but the permission mechanism maybe work differently for accessing other information such as address books. Therefore, we need to study a more variety of apps from different platforms to gain a more comprehensive view on how permission works for different mobile phone platforms.

Third, analyzing privacy control methods provides a simple angle for addressing privacy challenges. In fact, there are other determined factors that can impact the decisions of platform designers for solving privacy problems. For example, there are tradeoffs between privacy and other software qualities such as usability. Android platform designers may choose to sacrifice the flexibility of privacy controls for simpler user experiences. Similarly, the process of launching the mobile phone apps should also be considered when designing privacy solutions. For example, iOS app has application admitting process in Apple iTunes Store. The restricted process validates and checks for app bugs before an application gets introduced to the market. Contrarily, Google has a more loose process than Apple. Thus, to evaluate privacy for smartphone platforms and understand the privacy bugs, we should perform a comprehensive analysis on all factors that potentially impact the privacy.

VI. Conclusions

In this paper, we studied privacy control methods, i.e., how privacy at application level is controlled, for the Android and

iOS platforms. Our findings indicate that neither the iOS nor the Android platforms implemented the four components indicated by the Bellotti and Sellen's framework of information flow. We found privacy bugs related to such weakness of the platforms. We proposed an improved privacy control methods for both Android and iOS. The initial evidence shows that the programming model we proposed would be helpful for addressing users' privacy concerns. Our future work includes considering a full set privacy requirements, studying more varieties of mobile applications and performing a more in depth investigation on privacy bugs reported on the Android and iOS platforms.

REFERENCES

[1] Rong Tan; Junzhong Gu; Jing Yang; Peng Chen; , "Designs of privacy protection in location-aware mobile social networking applications," Pervasive Computing and Applications (ICPCA), 2010 5th International Conference on , vol., no., pp.62-68, 1-3 Dec. 2010

[2] Guanling Chen; Rahman, F, "Analyzing Privacy Designs of Mobile Social Networking Applications," Embedded and Ubiquitous Computing, 2008. EUC '08. IEEE/IFIP International Conference on , vol.2, no., pp.83-88, 17-20 Dec. 2008

[3] Jian Liao; Peiwei Huang; , "Improved mechanism for mobile location privacy," Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on , vol., no., pp.4 pp.-810, 7-7 Nov. 2005

[4] G. Russello, M. Conti, B. Crispo, and E. Fernandes, "MOSES: supporting operation modes on smartphones," in Proceedings of the 17th ACM symposium on Access Control Models and Technologies, 2012, pp. 3–12.

[5] Renegar, B.D.; Michael, K.; Michael, M.G.; , "Privacy, Value and Control Issues in Four Mobile Business Applications," Mobile Business, 2008. ICMB '08. 7th International Conference on , vol., no., pp.30-40, 7-8 July 2008

[6] V. Bellotti and A. Sellen. "Design for privacy in ubiquitous computing environments." Proceedings of the Third Conference on European Conference on Computer-Supported Cooperative Work, pp. 77–92, Milan, Italy, 1993.

[7] The AndroidManifest.xml File , http://developer.android.com/guide/topics/manifest/manifest-intro.html

[8] Location Strategies, http://developer.android.com/guide/topics/location/strategies.html

[9] Getting the User's Location, https://developer.apple.com/library/ios/#documentation/UserExperience/Conceptual/LocationAwarenessPG/CoreLocation/CoreLocation.html#//apple_ref/doc/uid/TP40009497-CH2-SW1

[10] Joon-Myung Kang; Sin-seok Seo; Hong, J.W.-K.; , "Usage pattern analysis of smartphones," Network Operations and Management Symposium (APNOMS), 2011 13th Asia-Pacific , vol., no., pp.1-8, 21-23 Sept. 2011

[11] Nan Xu, Fan Zhang, Yisha Luo, Weijia Jia, Dong Xuan, and Jin Teng. 2009. "Stealthy video capturer: a new video-based spyware in 3G smartphones." Proceedings of the second ACM conference on Wireless network security (WiSec '09). ACM, New York, NY, USA, 69-78

[12] Sébastien Gambs, Olivier Heen, and Christophe Potin. 2011. "A comparative privacy analysis of geosocial networks".Proceedings of the 4th ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS (SPRINGL '11). ACM, New York, NY, USA, 33-40

[13] iOS Discussion Board: https://discussions.apple.com

[14] Android Bug Post: http://code.google.com/p/android/issues/

[15] E. Smith, "iPhone applications & privacy issues: An analysis of application transmission of iPhone unique device identifiers (UDIDs)," pp. 1–19, 2010.

[16] M. Egele and C. Kruegel, "PiOS: Detecting privacy leaks in iOS applications," 2011.

[17] T. Vidas, N. Christin, and L. Cranor, "Curbing android permission creep," Proceedings of the Web, 2011.

[18] "Apple On iPhone Location Tracking: It's A Bug!". [Online]. Available: http://www.huffingtonpost.com/2011/04/27/apple-on-iphone-location-tracking_n_854252.html. [Accessed: 29-October-2012]

[19] "Google Is Tracking Android Users' Location Data, Say Researchers". [Online]. Available: http://www.huffingtonpost.com/2011/04/22/google-android-and-apple-track-your-location_n_852529.html [Accessed: 2-November-2012

[20] Xusheng Xiao, Nikolai Tillmann, Manuel Fahndrich, Jonathan De Halleux,and Michal Moskal. 2012. User-aware privacy control via extended static-information-flow analysis. In Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering.