Hobson's Choice: Security and Privacy Permissions in Android and iOS Devices

John Haggerty, Thomas Hughes-Roberts and Robert Hegarty*

School of Science and Technology, Nottingham Trent University, Clifton Campus, Clifton Lane, Nottingham, NG11 8NS, United Kingdom

*School of Computing, Mathematics and Digital Technology, Manchester Metropolitan University, Chester Street, Manchester, M1 5GD, United Kingdom

john.haggerty@ntu.ac.uk; thomas.hughesroberts@ntu.ac.uk; r.hegarty@mmu.ac.uk

Abstract. The use of smartphones and tablet devices has grown rapidly over recent years and the widespread availability of software, often from unknown developers, has led to security and privacy concerns. In order to prevent security compromises, these devices use access control as a means by which a user is able to specify an application's ability to interact with services and data. However, the use of access control as a security countermeasure in this environment is severely limited. For example, once permissions are granted to software, they may share data, such as location or unique identifiers with third persons without informing the user, whether or not the application is itself running. This paper presents the results of a comparative study conducted with computing students at two UK universities that identifies the issues surrounding software access control permissions in Android and iOS operating systems. Through this study, we are able to quantify the impact of security access permissions on mobile device security and privacy, even amongst specialist users.

Keywords: Mobile device security, access control, Android, iOS

1. Introduction

The use of smartphones and tablet devices has grown rapidly over recent years. The relatively low cost and high power of such devices has made them attractive to consumers, providing them with a truly mobile computing experience. The widespread adoption of mobile devices has resulted in the availability of thousands of applications for their owners. There are two models of software provision in this market; open and closed. In open markets, such as Google Play, application developers are able to distribute their software for a small fee or free of charge with little control by the owners of the store. In closed markets, such as the Apple App store, the distribution of software is more tightly controlled by the market owners. Software from both markets can be downloaded to a device and installed instantly, making it an attractive service for users and automated, dynamic analysis techniques are employed to identify malware.

The widespread availability of software, often from unknown developers, has led to security and privacy concerns. Many black market software stores exist, where malware is crafted into pirated commercial software and provided to end users free of charge. For example, viruses and Trojan applications can be made readily available by malicious developers through both legitimate and pirate software stores and researchers have demonstrated that it is possible to obfuscate malware from detection techniques (Apvrille and Nigram, 2014). This software can compromise the security of the device and the user by sending messages to premium rate services or stealing information, such as logon credentials or passwords (Delac *et al*, 2011). Alternatively, many applications interact with other online services, such as Google, through authentication tokens (Google, 2015). These tokens are sometimes sent in plain text, and can therefore be intercepted by a malicious application monitoring data sent from the device or a network to which they belong, to steal these credentials and access private information held elsewhere online.

In order to prevent security and privacy compromises, both Android and iOS devices use access control as a means by which a user is able to specify an application's ability to interact with services and data. For example, in the Android operating system, application data is held in isolated environments within the file system and inter-process communications are controlled through access permissions. The access rights depend upon the software to be used and the services that they require. For example, the BBC iPlayer application requires access to internet services to stream multimedia, phone calls so that the software is notified when the phone rings, and system tools to prevent the device entering sleep mode (Madden, 2012). Irrespective of the software and its source, users readily provide applications that they download with a range of permissions within the operating system.

However, the use of access control as a security countermeasure in this environment is severely limited. For example, once permissions are granted to software, they may share data, such as location or unique identifiers with third persons without informing the user, whether or not the application is itself running. The granularity of many requested access rights is too coarse to be useful, for example, the Internet permission provides broad-ranging capabilities without restricting access to specific URLs or domains. In the Android operating system, a user is presented with a list of capabilities, such as network services, location information, access to personal data, etc. upon installation. The user is not able to selectively grant access rights to the application as the only choice is to install the software or not, with all the permissions required by the developers rather than owner of the device. In the iOS operating system, a user will only grant capabilities as they use the software. However, to use the software with the functionality required by the user, they must grant the permissions requested. In both models, the user is presented with Hobson's choice; they are presented with the choice of installing or not installing, or using or not using, the software, with the potential cost to the security and privacy of information if they choose to install or run an application. Permission revocation is not an option unless the software is removed wholesale and users often fail to understand the risk posed by composite permissions. More importantly, the decision to grant permissions is being made by users that are potentially unaware of the security implications of making such a choice.

This paper presents the results of a comparative study conducted with computing students at two UK universities that explores issues surrounding software access control permissions in Android and iOS operating systems. Through this study, we are able to quantify the impact of Android and iOS security access permissions on mobile phone device security and privacy, even amongst specialist users. In particular, the results of this survey identify the permissions that participants are prepared to accept in installing applications from unknown sources. Moreover, it identifies whether these permissions, if wide ranging or inappropriate for the application, are questioned by the user. This paper will therefore quantify the problem of access control in the mobile environment and discuss appropriate countermeasures to mitigate the issue.

This paper is organized as follows. Section 2 discusses related work. Section 3 posits the methodology used for the comparative survey. Section 4 presents the results of the survey and discusses their significance to mobile device security. Finally, we make our conclusions and discuss further work.

2. Related Work

The most common operating systems in the mobile device market are Android and iOS, which combined accounted for over 90 per cent of smartphones in late 2014 (IDC, 2015). Both of these operating systems are based on the Linux kernel to provide hardware abstraction; Android builds on Linux whilst iOS is derived from OS X, a variant of BSD UNIX. On top of the kernel are the native libraries, which provide some of the common services for applications and other programs. Running processes rely on virtualization, whereby a virtual machine (VM) runs an application in its own instance. On top of this layer is the application framework, where code running for and on the VM provides service to multiple operations.

Due to the popularity of such devices running this architecture, mobile security and privacy has received much attention. In particular, their widespread use has introduced a range of new threats as well as transposing issues long associated with more traditional computing devices. For example, Delac *et al* (2011) present an attacker-centric threat model for mobile devices. This model realizes that the sources of threats to mobile devices are wide and varied, such as through Bluetooth connections, access to the Internet or networks, or USB peripherals. Erturk (2012) identifies the issue of privacy-invasive adware on mobile devices that target open-source platforms such as Android. Frank *et al* (2012) suggest that the Android operating system and Internet access through social network services such as Facebook can provide third-party applications with access to user’s private data as well as perform sensitive operations such as post online messages or make phone calls. Many of these potential issues do not have to be developed by very experienced software developers engineers or developers. For example, Mylonas *et al* (2011) suggest that all smartphone platforms could be used by average developers for attacking privacy or harvesting data without the user's knowledge or consent.

A number of countermeasures to these wide and varied threats have been proposed. For example, Batyuk *et al* (2011) posit a scheme whereby applications available via the Android market place are assessed for security vulnerabilities and a report is made available to users. This scheme also reverse-engineers applications to adjust security settings according to the user’s requirements. Ghosh *et al* (2012) propose a scheme for user privacy based on contextual information analysis to maintain user privacy when using applications that access and share device location and surroundings data. Yuhao Luo et al (2013) posit a method for the protection of user data by introducing a secure, enhanced kernel and data-at-rest encryption. In this way, they aim to provide data protection rather than address application privileges. Encryption has since been implemented and deployed in recent versions of the Android and IOS operating systems, with Android employing SELinux to provide Mandatory Access Control. Fazeen and Dantu (2014) propose a model for the identification of Android applications' intentions to identify malware through permission requests.

However, the issue remains that the principal defence strategy employed in both Android and iOS devices is based on the granting of access rights to applications, often obtained from unverified sources. This is compounded by the current practice of only allowing access to an application if, and only if, the user grants all the capabilities presented to them by that software.

Due to the complexity of applications running on powerful mobile devices, we posit that it is difficult for users to determine the implications of the often loosely scoped permissions. This problem is worsened by the security implications of composite permissions; with permissions that seem harmless in isolation, presenting a security risk in combination with other permissions. As mentioned previously the Internet permission is loosely scoped, permitting malicious applications to leak information obtained from other permissions granted by the user, this 'gateway permission' requires much finer grain control. Tighter granularity for key permissions such as Internet access would reduce the ability of malicious applications to breach user privacy. If URL or domain specific permissions were provided, the increase in transparency would enable users to make more informed decisions about the applications they install, and the destination of data leaving their devices.

The above challenges have emerged as consumers shift from a conventional computing environment, using PCs and laptops to a mobile environment in which they use smartphones and tablets. Unfortunately it is not immediately apparent to many users, that mobile devices contain much more sensitive personal information than conventional computing devices. Coupled with this conventional computing environments do not typically require the end user to grant permissions, rather a user agrees to a EULA (often without reading it).Thus, even experienced computer users are not well equipped to deal with the threats to privacy posed by the transition from a conventional to mobile computing environment. A further 'at risk' group are users who have little to no experience of computing devices; this group has stumbled into computing due to the low cost and high availability of smart phones, and may be unaware of the threat computing devices pose to their personal privacy.

While malware is usually found in software from unverified sources, there have been instances of malware in the app stores of both iOS and Android (Porter Felt *et al,* 2011). The user's judgment may be impaired through the false sense of security that using an official app store provides. Both the major app stores deploy automated (e.g. Google Bouncer) and manual vetting of applications, however the purpose of such vetting is to identify malware. These vetting processes do not take account of the user's privacy, with the choice of whether to accept permissions being delegated to the end user. Furthermore, the way in which consent is obtained differs between platforms with some requiring full consent at installation and others during run-time. (Porter Felt *et al*, 2012). The question is therefore raised: does the way in which permissions are requested influence the extent of consent and general user consent? Given the prominent role the user plays in managing the security of a personal mobile device, understanding end-user perceptions is vital in designing appropriate solutions.

In the next section, we present the methodology of a survey of computing students at two UK universities that attempts to quantify the security challenges that this model raises.

3. Survey methodology

The aim of the survey is to begin exploring how users approach permissions on a mobile device by comparing the two prevalent methods of application permission acceptance in iOS and Android devices. Participants are therefore presented with an application's permissions request and asked what their reaction would be: accept or reject. They are then asked to provide an explanation for their choice.

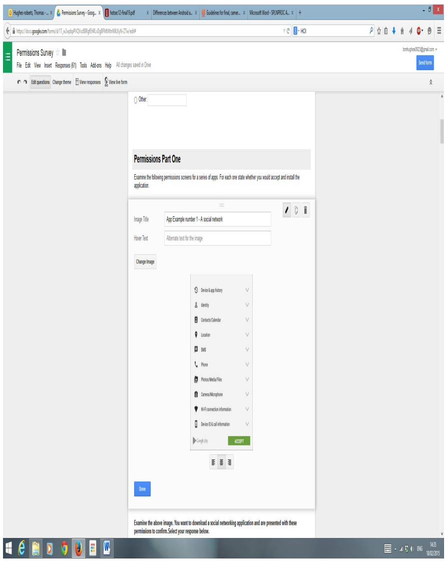
The first part of the survey presents the permission screens as they would within an Android OS. That is, in order to download and install the app they must agree to all the required permissions. The second part of the survey presents the permissions as they would be found within iOS where acceptance is required in increments and based on the task at hand. To that end the participant is presented with a sentence detailing the context of the permissions request; for example, "you wish to make an in-app purchase, the app requires permission to access…"

Table 1 provides an overview of the applications present in the survey, the permissions of which are taken from actual software on the market.

Table - Part 1, Applications Overview

App	**From (at time of survey)**	**Notes**
Social Network	Facebook	
Messenger	Facebook Messenger	
Invasive Flashlight	High Powered Torch	
Less invasive Flashlight	LED Torch	Only if rejected invasive Flashlight
File Store	Dropbox	
Game	Angry Birds Transformers	
Invasive Camera	Google Camera	
Less Invasive Camera	Camera 1080	Only if rejected invasive camera
Banking	HSBC	
Loyalty Card	Game Rewards	

For each of these applications, the participant is presented with the generic name of the application (e.g. "you wish to download a social networking application") and an accompanying permissions screen, an example of which can be seen in Figure 1.


Hughes-roberts, Thomas - ...
Permissions Survey - Goog...
hotsec12-final19.pdf
Differences between Android a...
Guidelines for final, camer...
Microsoft Word - SPLNPROC A...
Permissions Survey
File Edit View Insert Responses (61) Tools Add-ons Help All changes saved in Drive
Send form
Edit questions Change theme View responses View live form
Device & app history
Identity
Contacts/Calendar
Location
SMS
Phone
Photos/Media/Files
Camera/Microphone
Wi-Fi connection information
Device ID & call information
Google play
ACCEPT
Done
ENG
14:35
18/02/2015

○ Other:

Permissions Part One

Examine the following permissions screens for a series of apps. For each one state whether you would accept and install the application.

Image Title: App Example number 1 - A social network

Hover Text: Alternate text for the image

Change Image

Examine the above image. You want to download a social networking application and are presented with these permissions to confirm.Select your response below.

Figure - Example permissions screen

The second half of the survey took three of these applications and split their permissions based on a specific usage of the software. Table 2 provides and overview of the applications presented to the respondent and the contextual statements behind the permission request.

Table - Part 2, Apps overview

App	Context	Permissions
App 1 – Social Network	Install and Launch	Device Profile Contacts/Calendar Wi-Fi
	Post a picture on timeline	Photos/Media/Files Camera/Microphone Location
	Text Control over account	SMS Phone
App 2 – Game	Install and Launch	Profile Wi-Fi Phone state
	Save Game	Photos/media/files
	Make an in-app purchase over SMS	In-app purchases SMS Phone
	Play-for-Free	Location Phone state
App 3 – Banking App	Install and Launch	Identity

		Phone state
	Find local branch	Location
	Remote check deposit	Camera/Microphone
	Transfer to Contact	Phone Contacts List

Following the review of the two methods of permissions acceptance participants are asked which approach they prefer in managing their privacy.

Responses to each set of permission requests are compared against each other to examine if the greater degree of control in the granular permissions approach does indeed lead to a more selective acceptance.

Participants on computing courses at two Universities were approached to take part in the study which was completed within a scheduled lab session. In total, 60 participants took the survey (50 male, 10 female); it is noted that there is a bias in the sample in terms of gender due to the sampling method chosen and as such this cannot be considered a true representation of a full population. Of this sample, the same number of participants used iOS as Android as an operating system: 45.8% each with the remainder using other OS's. This does not reflect the market shares in 2014 as suggested by IDC (2015) where Android accounts for 84.4 per cent of the mobile market compared to iOS's 11.7 per cent in the last quarter.

The following section summarizes the results from this survey and presents points for further discussion from the data obtained.

4. Survey Results

The breakdown of acceptance for whole permission requirements (part 1 of the instrument) is broken down in table 1.

Table - Whole Acceptance Summary

App Part 1	**% Accept**
Social Network	50%
Messenger	66.7%
Invasive Flashlight	18.3%
Less invasive Flashlight	61.2%
File Store	71.7%
Game	48.3%
Invasive Camera	28.3%
Less Invasive Camera	90.7%
Banking	60%
Loyalty Card	30%

Interestingly, despite only 50% of participants stating that they would accept the permissions for the social networking application, 80.7% of participants admitted to having the social networking app Facebook on their phone (which these permissions were taken from). Furthermore, if this particular response is broken down then the majority of change is from iOS users where 74.1% would reject the permissions presented to them but 85.2% have the app on their phone. The change is much less pronounced in users of Android where 75% of participants stated they would accept the permissions with a similar amount (71.4%) admitting to having the app on their phones. Indeed, it is the Android users who would be more used to this style of permission acceptance.

Perhaps unsurprisingly, there was a large increase in the number of acceptances in the two applications where a less invasive version was offered if the first was rejected. Participants stated that the requests were much more reasonable: “it needs this”, “only requires relevant access for the flashlight to function”. This would suggest that attention was paid to the permissions requested within the context of the survey and an assessment made as whether or not they are reasonable. However, a number of participants still felt access to the camera was unreasonable: “it would still have access to my camera and microphone, there are other ways of turning on the flashlight”. This is despite the flashlight being governed by the same set of permissions as the camera in Android suggesting there is a lack of understanding of the technical aspect of permissions; this is further discussed later in the paper. Furthermore, a number of participants did state that the reason they accepted was “I always do it without thinking” and this was carried through to the survey for a number of example applications.

Table - Part 2 Permissions Overview

App	**Context**	**% Accept**
App 1 – Social Network	Install and Launch	66.7%
	Post a picture on timeline	75%
	Text Control over account	65%
	Accept to Each	40%
App 2 – Game	Install and Launch	74.1%
	Save Game	72.4%
	Make an in-app purchase over SMS	48.3%
	Play-for-Free	52.6%
	Accept to Each	24.6%
App 3 – Banking App	Install and Launch	82.1%
	Find local branch	89.3%
	Remote check deposit	29.8%
	Transfer to Contact	61.4%
	Accept to Each	19.3%

Results from Table 4 would suggest that participants, when given the opportunity, will be more selective of their acceptable permissions. On each of the above sample scenarios the total amount of complete answers is less than the comparable app in the previous section. For example, 60% of participants confirmed that they would accept the banking app's permissions when all are required for install compared to only 19.3% who confirmed that they would accept each of the permissions when given separately. This would espouse the benefits of more granular control over permissions on a mobile device. Indeed, the majority of participants (66.7%) preferred this style of permission management. However, they would still need to accept all permissions in totality were they to be able to use the software.

However, the likelihood of acceptance appears to be coupled with understanding. For example, 89.3% of participants allowed the banking app to access the devices location to find the local branch. Reasons for acceptance appeared to suggest that the request reasonable: "It needs it", "Seems helpful", "Needs to locate you and find the nearest branch". When understanding is lacking the likelihood of response is less. For example, in the same app the permission requires access to a camera to take a picture of check in order to make a remote deposit. Only 29.8% of participants agreed to this particular access: "Why would it need my camera", "Camera not needed", "I don't think it needs these" etc. This reflects the approach some app developers are taking, by explaining the reason behind permissions in the app market, in order encourage users to install applications with obscure looking permissions.

5. Conclusions and further work

Mobile devices and their associated software stores have become a ubiquitous part of society. Users from a wide spectrum of backgrounds frequently interact with the permissions systems employed by app stores during the installation and updating of mobile apps. Mobile devices are much more integrated into our daily lives than fixed computers facilitating a wide variety of tasks (e.g. diary, contacts, navigation, and photography). As a consequence they contain large amounts of personal data. Permissions are requested to enable users to determine what personal data applications have access to.

The findings of the survey illustrate that the respondents are aware of software permissions, however in many instances accept the permissions requested for the majority of applications. It is obvious that a user's decision about whether to install an application or not is governed by more than just their permissions.

Existing usage of a service is likely to increase the chances of an application being installed. The motivation for the user is likely that of acceptance; "the provider already has access to my data, as I'm already a user of the service". Alternatively the reputation of the provider is assessed with the user making a value judgment based on the size/reach/popularity of an application provider and utility of the application. For example many users rejected the generic social network application, yet admitted to having software installed from one of the established social networks.

User understanding plays a large role in the acceptance of application permissions. It was clear from the survey responses that some effort was made by users to understand why permissions are requested. However awareness of the broad granularity of permission is limited. Additionally users demonstrate their lack of understanding of the underlying fundamentals of how some applications work, refusing to install a banking application that required camera access in order to scan/photograph a cheque. This limited understanding identified in the sample group of computer specialists is likely much larger in the general population. At present the user is provided with a list of the functionality each permission provides, rather than the reason that the permission is required, leaving the user to guess why each permission is needed. It would be beneficial if a description of the reason each permission request is being made was presented to the user during the installation process. The description could be derived from the requirements stage of the software development process. However, such additions to the application must not add to the complexity of the interaction or risk being similarly ignored by the user. The question is therefore raised: how can users be made aware of the context of permission requests without adding to the complexity of the system?

In spite of the additional factors considered by users installing applications, and misunderstandings related to the permissions model, a more granular approach to application permissions resulted in users being more selective about the software that they install, i.e. rejecting invasive applications.

We therefore conclude that increased choice, realized through a more granular approach to application permissions provides improvements to user privacy.

Further work is required to determine how users interact with an application permission during a software update. The value judgment made by users is likely to vary when they are prompted to grant permissions for software that they have used frequently for an extended period of time.

6. References

7. Apvrille, A. and Nigam, R. (2014), "Obfuscation in Android malware, and how to fight back", *Virus Bulletin*, July 2014, pp. 1-10.
8. Batyuk, L., Herpich, M., Camtepe, S.A., Raddatz, K., Schmidt, A.D. and Albayrak, S. "Using Static Analysis for Automatic Assessment and Mitigation of Unwanted and Malicious Activities Within Android Applications", *Proceedings of the 6th International Conference on Malicious and Unwanted Software*, 18-19 October, 2011, Fajardo, Puerto Rico, pp. 66-72.
9. Delac, G., Silic, M. and Krolo, J. (2011), "Emerging Security Threats for Mobile Platforms", *Proceedings of MIPRO 2011*, 23-27May, 2011, Opatija, Croatia, pp. 1468-1473.
10. Erturk, E. (2012), "A Case Study in Open Source Software Security and Privacy: Android Adware", *Proceedings of the World Congress on Internet Security*, 10 - 12 June, 2012, Ontario, Canada, pp. 189-191.
11. Fazeen, M. and Dantu, R. (2014), Another Free App: Does It Have the Right Intentions?", *Proceedings of the 12th Annual Conference on Privacy, Security and Trust*, 23-24 July, 2014, Toronto, Canada, pp. 283-289.
12. Frank, M., Dong, B., Porter Felt, A. and Song, D. (2012), "Mining Permission Request Patterns from Android and Facebook Applications", *Proceedings of the 12th International Conference on Data Mining*, 10-13 December, 2012, Brussels, Belgium, pp. 870-875.

13. Ghosh, D., Joshi, A., Finin, T. and Jagtap, P. (2012), "Privacy Control in Smart Phones Using Semantically Rich Reasoning and Context Modeling", *Proceedings of the Symposium on Security and Privacy Workshops*, 24-25 May, 2012, San Francisco, CA, USA, pp. 82-85.
14. Google (2015), “Using OAuth 2.0 for Server to Server Applications”, available at https://developers.google.com/accounts/docs/OAuth2ServiceAccount, last accessed 10 February, 2015.
15. IDC (International Data Corporation) (2015), “Smartphone OS Market Share, Q3 2014”, available at http://www.idc.com/prodserv/smartphone-os-market-share.jsp, last accessed 10 February, 2015.
16. Madden, D. (2012), "BBC Internet Blog - BBC iPlayer Android App Update", available at http://www.bbc.co.uk/blogs/legacy/bbcinternet/2012/02/bbc_iplayer_android_update.html, last accessed 10 February, 2015.
17. Mylonas, A., Dritsas, S., Tsoumas, B. and Gritzalis, D. (2011), "Smartphone security evaluation The malware attack case", *Proceedings of the International Conference on Security and Cryptography*, 18-21 July, 2011, Seville, Spain, pp. 25-36.
18. Porter Felt, A., Egelman, S., Finifter, M., Akhawe, D. and Wagner, D. (2012), “How to Ask for Permission”, *Proceedings of HotSec ‘12*, 7 August, 2012, Bellevue, WA, USA,.
19. Porter Felt, A., Finifter, M., Chin, E., Hanna, S. and Wagner, D. (2011), “A survey of mobile malware in the wild”, *Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, 17-21 October, 2011, Chicago, IL, USA, pp 3-14.
20. Yuhao Luo, Dawu Gu and Juanru Li (2013), "Toward Active and Efficient Privacy Protection for Android", *Proceedings of the International Conference on Information Science and Technology*, 23-25 March, 2013, Yangzhou, Jiangsu, China, pp. 925-929.