Studying Users' Adaptation to Android's Run-time Fine-grained Access Control System

Panagiotis Andriotis[a,*], Gianluca Stringhini[b], Martina Angela Sasse[b]

[a] *University of the West of England, Department of Computer Science and Creative Technologies, Frenchay Campus, Bristol, U.K.*

[b] *University College London, Department of Computer Science, London, United Kingdom*

Abstract

The advent of the sixth Android version brought a significant security and privacy advancement to its users. The platform's security model has changed dramatically, allowing users to grant or deny access to resources when requested by applications during run-time. This improvement changed the traditional coarse-grained permission system and it was anticipated for a long time by privacy-aware users. In this paper, we present a pilot study that aims to analyze how Android users adapted to the run-time permission model. We gathered anonymous data from 52 participants, who downloaded an application we developed and answered questions related to the run-time permission model. Their answers suggest that most of them positively accepted the new model. We also collected data that describe users' permission settings for each installed application on their devices. Our analysis shows that individuals make consistent choices regarding the resources they allow to various applications to access. In addition, the results of this pilot study showcase that on a second data collection round (occurred one month after the first phase of our experiments), 50% of the respondents did not change a single permission on their devices and only 2.26% of installed applications (on average) presented altered permission settings.

Keywords: Privacy, Android, Usability, Acceptance, Controls, Permissions

1. Introduction

When Android Developers released the Developers Preview of the Marshmallow version (v6.0) during summer 2015, a major change at the permission system was introduced; the sixth version initiated the run-time permission model. The previous versions are listing at instalation time the resources that the application to be installed is going to utilize. After reviewing the requested permissions (which were presented as groups, e.g. Contacts) the user can choose to accept or deny the installation. This binary model (accept-reject) has been criticized at the past as being ineffective to provide meaningful information about the way the application to be installed will affect user's privacy [1, 2]. In addition, it limits users' ability to manage the applications' accessibility to their private data. Therefore, the transition from this model to a new one, that would allow users to control the resources that applications were allowed to use (following the iOS paradigm) was anticipated for a long time.

The run-time permission model (aka ask-on-first-use (AOFU) [3]) is based on the principle of least privilege and assumes that applications will be able to function at a basic level, even if the users do not provide access to resources that might affect their privacy. Therefore, applications designed to adhere to this model must request access to sensitive resources during run-time. These actions will (in theory) keep users informed about what an application is trying to do in the background and will provide limited contextual information [4].

According to the official documentation for Android Developers[1], there are two basic categories of

*Corresponding author

Email address: panagiotis.andriotis@uwe.ac.uk (Panagiotis Andriotis)

URL: www.andrio.eu (Panagiotis Andriotis)

[1] http://bit.ly/2d4AdGH

permissions; *normal* and *dangerous*. The documentation notes that the system automatically grants access to resources that applications requested via *normal* permissions, because access to these resources is considered to be of low risk. On the other hand, if an application needs to access users' private information, or other sensitive data stored on the device, then the associated permissions with these actions are considered as *dangerous*. Hence, applications designed to function properly under the AOFU permission model, need to request user's permission during run-time, in order to access sensitive information. Therefore, it lies with the users' discretion if they will accept or deny access to sensitive resources. Additionally, Android users are able to revoke access to resources via the Settings application under this model. Currently, there exist nine groups of dangerous permissions: Calendar, Camera, Contacts, Location, Microphone, Phone, Sensors, SMS, Storage.

Recent research work indicated that permission requests are vital in terms of conducting resource usage risk assessment and identifying malicious behaviors [5]. Permission requests are also used to assess the app's quality and installation risk, based on patterns identified in high-reputation applications in the Android market place [6]. Therefore, app permissions play a major role in users' privacy and security. The fact that the number of decisions that smartphone users must make (regarding the acceptance of these permissions) can be unrealistically high [7], urged researchers lately to propose automated permission recommendation schemes. Some systems use crowdsourcing methods [8] and others employ machine learning models that incorporate contextual information aiming to predict users' privacy decisions [3]. In order to achieve that, Wijesekera et al. [3] used modified versions of the Android operating system to acquire application usage data and the Experience Sampling Method (ESM) [9] to collect ground truth data about users' privacy preferences. Additionally, Liu et al. [7] deployed rooted devices which were modified and enhanced with the Android permission manager "Apps Ops" [2, 10]. Hence, prior work was based on experiments conducted with modified devices, specifically instrumented to gather privacy related data.

Knowing that under the coarse-grained permission model, users are not allowed to intervene with the access control system (since applications can access all resources on a mobile device after the installation process), we investigate in this paper how Android users adjusted their privacy preferences under the fine-grained run-time permission model (AOFU). We consider the following questions in this pilot study. a) Which are the sensitive resources Android users allow more often to be accessed on their phones? b) Are they strict or selective when applications request access to specific sensitive data? c) Do they make consistent choices when they grant or deny access to these resources? d) Are these choices time persistent?

To this end, this paper[2] presents the results of a pilot study that assesses users' adaptation to the Android run-time permission model. For the needs of this study we developed and distributed an application at the official Android marketplace (Google Play) aiming to collect anonymous data related to the permissions that were granted (or denied) by users at that time ('Permissions snapshot'). We did not monitor users' actions for a long time because we assumed that this choice would discourage many people to voluntarily download the app and participate to the study. This is a different approach from prior work [3, 4, 7] as we aim to gather permission information from devices that were actually used by participants in their daily lives and were not running a modified version of the operating system. Our data collection method is not intrusive or pervasive and does not introduce biases related with asking security and privacy questions (privacy nudges). Thus, we chose to obtain snapshots of the permission settings from the participants' devices. Our application collected permission data twice (in an one-month period) and only when the users were informed and agreed to provide them, according to the ethics approval agreement. The aim of our pilot study is to examine users' perceptions of the provided security and privacy, and at the same time, to investigate how Android users adapted to the AOFU permission model. This work studies and presents security and privacy preferences of Android users of the fine-grained permission model. The contributions of this paper are the following:

- We collected data derived from devices that were running the Android Marshmallow operating system, hence the permission data came from a sample of 52 participants who were ac-

[2]This is an extended version of our work [11] presented on December 2016 at the 8th IEEE International Workshop on Information Forensics and Security (WIFS) 2016.

tually using these devices for a considerable period of time[3].

- We present comparative views of users' permissions settings and other privacy preferences associated to the use of popular social media. Moreover, we showcase which sensitive resources were used from our participants more frequently.

- We demonstrate that our participants presented a consistent behavior regarding the resources they allow to be accessed by social media and communication applications. Furthermore, this pilot study shows that the granted permissions to installed applications from the same participants after a period of one month were not dramatically altered. This result verifies similar findings presented recently [7].

The rest of this paper is organized as follows. The next section discusses the methodology we used to derive data and reconstruct permission settings for each participant. Section 3 presents the acquired results focusing at the beginning on the survey answers; then it analyzes our findings from the collected permission data. In section 4 we discuss limitations of our study, proposing at the same time directions for future work. We review related work in section 5 and conclude this paper in section 6.

2. Methodology

This section presents the methodology we used to collect and analyze data. Data collection was carried out in two phases using an application we developed, which was distributed via the official Android marketplace (Google Play) following the example of other recent research works [12]. The application, named "Permissions Snapshot", initially served as a survey instrument, but it also collected anonymous permission data from the devices that were using it. The participants needed to download the application on their devices, answer six multiple choice questions about their experience with the run-time permission model and then send permission data to our server. At the second phase (after a period of one month) the same participants were asked to send permission data again, as explained in more details in the following section. Before we distribute the application on Google Play and publicize it, we obtained approval to proceed with this project from the UCL Ethics Committee (Project ID Number: 8945/001).

2.1. Survey and Questionnaire Design

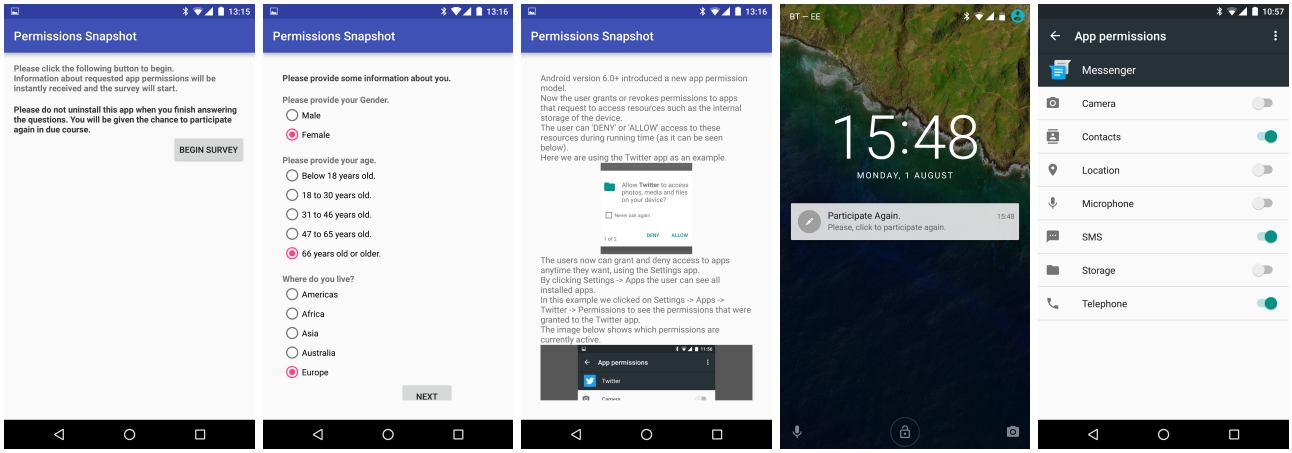
The application we developed targeted Android Marshmallow users (SDK 23+) and could not be installed on devices that run an older version of the operating system (OS). This means that the collected data came from participants who were already familiar with the sixth Android version (Marshmallow). During the data collection period (June - August 2016), the most modern version of the OS was the sixth; however, the seventh version ('Nougat') was released as a "Developers Preview" version.

Our application did not collect personal information apart from the package names of the installed applications on the device and the requested permissions. The participants were informed about this action after reading the 'Information Sheet', which was provided at the 'Description' section of the installation page on Google Play. The 'Information Sheet' was also shown when the application was launched by the user. "Permissions snapshot" did not collect application usage data because that would entail additional effort from users to turn on the usage statistics feature on their phones. This action cannot occur automatically or programmatically on modern Android versions. We chose to refrain from engaging users in this action, having in mind that the participants (volunteers) could be discouraged by any additional complexity, or even worse, become sceptical about their involvement with the study and, therefore, quit before answering any questions.

In order to avoid duplicate entries from participants (and devices), "Permission Snapshots" requested from the system (during run-time) to get the `ANDROID_ID` of the device[4]. This number was used by our application to compute and return a hashed value that could be used from us to maintain users' anonymity and, at the same time, identify possible duplicate entries during analysis. Before releasing the data to the public, we replaced these hexadecimal strings with random numbers to further anonymize them.

[3]The anonymized dataset can be found online at the following address: http://dx.doi.org/10.14324/000.ds.1520825

[4]The `ANDROID_ID` is a hexadecimal number granted by the system, which uniquely distinguishes devices.



(a) Begin survey (b) Demographics (c) Tutorial (d) Notification (e) Permission Groups

Figure 1: Screenshots Showcasing "Permissions Snapshot" Application and System's Functionality.

Next, we discuss how participants were interacting with our application. At the first phase, when users launched the application, they had to read the 'Information Sheet' which described the aim of the study and the steps that would follow. Participants had to tap on a check box to indicate they gave their consent to share anonymous data and participate to the study. The users were always informed about the current and the next step of the process via short texts provided by the application (see figure 1a). After users agreed to provide their consent to share anonymous data, the application received detailed information about the permissions that the system granted (or not) for each installed application at that specific time. In order to achieve that, our application utilized the `PackageManager` class and received data via its `getPackageInfo` public method. This was an instant procedure on contemporary devices.

After permission data were stored locally on the device, the user was able to participate to the survey. Initially we gathered basic demographic data, as figure 1b demonstrates. Then the participants had to answer six multiple choice questions (by clicking on radio buttons). Each question was presented on a single `activity` and the user had to click the 'Next' button to proceed with the next question. It should be noted that all questions had predefined answers in order to make analysis easier for us and distinguish participants who were just skipping questions by clicking the 'Next' button. When the six questions were answered, the participants sent the responses (along with the demographic and the permissions data) to our server by clicking a button. Respondents were able to withdraw and uninstall the app any time, before they send their data to the server. At the final step of the first phase, the application showed to the participant a short tutorial that discussed the changes at the permission model (figure 1c). Our application's users were told that they could uninstall it at that phase. However, we suggested them to keep the application on their devices, because they would have the chance to participate to the survey again in the near future.

The questionnaire should be short and simple, because according to [13], these types of questionnaires usually attract higher response rates than long and complex ones. Also, we needed the completion of the survey to be an effortless and pleasant process, given that our participants would volunteer to provide responses. We chose closed format questions because they are easy to fill in. Moreover, it would be easier for users to click on radio buttons instead of typing answers to open questions on their phones. The questions had predefined answers to assist us clean up the data from responses coming from users that where just clicking the "Next" button without reading the questions. However, we did not include verifier questions because the questionnaire was short and we did not want to confuse or disorient the user by asking again the same question. We ensured that questions were easily readable on the majority of devices using facilities

provided by the Android Studio (an integrated development environment for Android developers).

When designing the questionnaire, we followed the conventional wisdom which contains advice about how to optimize question order [14]. We ordered the questions according to the following logic: Early questions should be easy and pleasant to answer [15], and questions on the same topic should proceed from general to specific [14, 16]. Hence, participants were first asked how long they were using the current OS version and then if they had noticed the permission model change. Then we asked the respondents if they believed that the aforementioned change affected positively their ability to manage shared data. With the following two questions (number 4 and 5) we intended to see the users' familiarity with the new model and if popup messages, appearing during run-time, were causing fatigue [17]. Finally, with the last question we intended to validate that users prefer fine-grained permission models as proposed in [18]. The questions are as follows:

1. How long have you been using the current version of the Android operating system? (Android Marshmallow or version 6.0+)
   Offered answers: a) 0 - 6 months, b) 7 - 12 months, c) More that 1 year, d) Don't Know.
2. Have you noticed any changes at the current app permission model, compared to the previous versions of the operating system? (For example, have you ever seen any messages from any app that asks to give it your permission in order to access specific resources, e.g. the Camera?)
   Offered answers: a) Yes, b) No, c) Don't Know.
3. Do you think that now you have more control as a user on the personal data you share? (E.g. Contact list, SMS, etc.)
   Offered answers: a) Yes, b) No, c) Don't Know.
4. Please state if the following quote is Correct or Wrong, or choose "Don't Know" if you are not sure. "I can revoke or grant any app permissions anytime from the Settings app of my device."
   Offered answers: a) Correct, b) Wrong, c) Don't Know.
5. Please state if you Agree or Disagree with the following quote. "I believe that the new runtime permission model is irritating because it asks me too many questions during running time (such as: Allow APP to take pictures and record video?)."
   Offered answers: a) I Agree, b) I Disagree.
6. If you have used older versions of the Android operating system (e.g. if this is not your first Android device) you have probably noticed that now some apps request your permission to perform some actions, e.g. to access your SMS. Previously you just downloaded the apps and no more requests were made from the device during runtime. Do you prefer the new permission model (runtime permission requests) or the old model?
   Offered answers: a) I prefer the new runtime permission model, b) I prefer the old model, c) I don't have any specific preference.

For the second phase of data collection, the application generated a notification message after a period of one month. Participants who did not uninstall the application after the completion of the first phase, received a notification on their devices (and wearables) to participate again to the survey (as seen at figure 1d). Hence, there was a chance for participants to upload permission data again, following the same procedure they used at the first phase of data collection, i.e. by clicking a button. Therefore, we could use the second set of data to compare them with the first set. After the second set of data was delivered to our server, the users were gracefully adviced to uninstall the application. Interestingly, taking into account data provided by the Google Play Console[5] (and considering also a few pilot trials we ran on various devices prior the release of our application), 15 devices (out of 61 that downloaded our application) appeared to have our application installed four months after we released it for the first time to the Google Play store. This means that approximately 24.6% of the users did not uninstall the application, despite the fact that they were prompted to do so.

2.2. Permission Data Acquisition

This section presents our methodology to acquire permission data from devices. Our intention was to correlate the data in order to provide permission information in a similar fashion with the original system. For example, figure 1e shows permission

[5] https://play.google.com/apps/publish/

information for the application 'Messenger'. Information for granted permissions for each application on Android devices can be obtained using the Settings application as follows: Settings → Apps → (application) → Permissions. Using the Settings application, users can grant, deny or revoke permissions to the following groups; Calendar, Camera, Contacts, Location, Microphone, Phone, Sensors, SMS, Storage.

Using the flag `GET_PERMISSIONS` when calling the `getPackageInfo` method of the `PackageManager`, we can get information about the status of each permission for every application (`requestedPermissions` and `requestedPermissionsFlags`). Note that "Permissions Snapshots" did not gather information about system's applications. After experimenting with these settings during a pilot study conducted prior to the official release of the application, we concluded that for each requested permission, the system returns public arrays of `String`s and `int`s denoting the current permission settings on the device. In our log files we denoted a granted permission with *3* and a denied permission with *1*.

According to the Android Developers' documentation[6], the first time an application requests to access sensitive resources (such as the SMS list on the device, using for example the permission `READ_SMS`), it displays to the users a dialogue message in order to get explicitly their permission to do so. However, if users allowed at the past access to a sensitive group (e.g. accepted `READ_SMS` from the SMS group), then when the application requests a different permission from this group (e.g. `SEND_SMS`), the system will *immediately grant* it.

For our analysis we considered these characteristics to reconstruct the permission settings for each application (and each permission group). For example, if for the package `org.telegram.messenger` (the 'Telegram' application) the `PackageManager` returned "`android.permission.READ_CONTACTS, 1`" and "`android.permission.WRITE_CONTACTS, 3`", this means that the application was granted permission to access the 'Contacts' group. Thus, after a user had chosen to see the permission settings of this application from the Settings app on the device, she would see that the Contacts switch would be turned to the 'On' position (as figure 1e shows).

We treat different versions of the same application uniformly, because we did not expect to see massive changes among various versions, given that data collection was done in a short period of time.

3. Results

The study was publicized at university mailing lists, social media and forums for a short period of time (June - August 2016). We stopped collecting data after the end of August 2016 and we finally received responses from 52 participants. It should be noted that no compensation or prizes were offered at this campaign. Our respondents contributed to this study as volunteers. In addition, during the specific period, the Android Developers Dashboards[7] reported that only 10% of the Android users that visited Google Play had installed the Marshmallow version on their devices. Therefore, finding volunteers for this study was extremely difficult at the given period. Having that in mind we followed a lightweight approach in data collection and decided to apply the snapshots model instead of constantly monitoring users' devices. We believe that this approach would not discourage participants to provide anonymous data. In addition, we expected that our study would probably attract more tech-savvy users, given that new Android versions' adoption is usually slow and Nexus phone users are getting software updates faster than others.

Three users from our sample provided the predefined answers of the questionnaire, hence their demographic data and responses were excluded from the presented results. However, their device data were included in the analysis, because they could not be manipulated or falsified from these users; these are device-dependent data. Also, two participants stated that they were below 18 years old; therefore, we excluded their data and their responses from our dataset and did not take them into account in the analysis. The minimum age for an Android user to participate to our study was set to be the age of 18. Additionally, a particular file contained permission data, but no demographics or responses. However, these permission data were included in the analysis. Finally, one file (derived from device No 24) contained only permission data of our application and the responses to the

[6] http://bit.ly/2d4AdGH

[7] http://bit.ly/1kjKifB

six questions. We included these responses to the demographic data too.

3.1. Demographics and Questionnaire

In total, 46 responses to the six questions were included to the analysis. 85% of the participants were male and 15% were female. More than half of the users were between 18 to 30 years of age (52%), 37% were 31 to 46 years old, 9% were between 47 and 65 years old and 2% were above 66 years old. The majority of the respondents were residents of Europe (74%); the rest were from America-Canada (17%) and 9% of the respondents claimed they were from Asia.

Regarding the responses at the first question, 52% of the participants had been using the Android Marshmallow version for 0 - 6 months. 30% of the participants had been using the OS for 7 - 12 months and 11% claimed they had been using the system for more than a year. The latter response may seem inaccurate because Marshmallow was released during the last quarter of 2015. However, this finding might indicate that a few respondents were mobile software developers who were using the Developer Previews of the system; these were released earlier, during summer 2015. This explanation is reasonable if we take into consideration that the application was also advertized to Google+ Communities related to Android. The rest of the participants (7%) clicked the "I Don't Know" option.

Considering the second question, 89% of the participants replied that they had noticed the change at the permission model. 7% said they did not notice any difference and 4% chose the "I Don't Know" answer. This finding shows that the majority of the participants were familiar with the change at the permission model.

The consequent questions were related to the security and usability of the new permission model. At the third question, 65% of the respondents replied positively; they thought that the users could effectively control the personal data they share under the run-time permission model. 17% replied negatively and 17% said "I Don't Know". The responses at question 4 demonstrated that the participants of our study were familiar with the new capabilities the Settings app provide. 78% chose the "I Agree" response, 7% clicked on "I Disagree" and 15% said "I Don't Know". Considering the usability of the model and the security fatigue that might cause, the responses at the fifth question showed

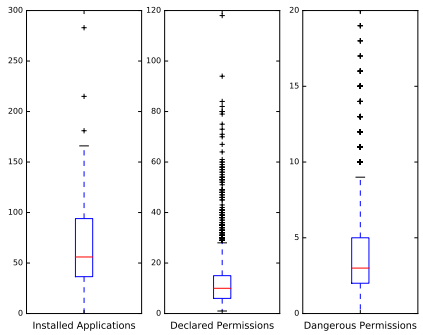


Figure 2: Boxplots Showing Distribution of the Number of Installed Applications, Declared Permissions and Dangerous Permissions per Device.

that Marshmallow users were not overwhelmed by security dialogue messages produced by the system. 89% of the participants replied "I Disagree" and only 11% clicked on the "I Agree" choice.

At the final question the participants were asked if they prefer the run-time permission model or the old one. 78% of them stated they prefer the run-time permission model and 9% said they prefer the previous model. Additionally, 13% clicked the "I don't have any specific preference" option. The answers to the last question indicate that users reacted positively to the changes, confirming previous results [18].

To conclude this section, the answers of the majority of users in our sample suggest that they think they can control the data they share more efficiently under the run-time permissions model. Also the participants in our study were not frustrated by the dialogue messages the system issues and, finally, they preferred the run-time permissions model, compared to the previous one.

3.2. Data Analysis

As discussed previously, we received permission data from 52 devices but we finally analyzed data derived from 49 devices. Recall that one device returned permission data only from our application and 2 participants declared they were younger than 18 years old; thus, these permission data were excluded from analysis. The following information is provided to describe our dataset in details.

3.2.1. Installed Applications

The average number of installed applications per device was approximately 71 and the standard deviation was 52.69. The maximum number of applica-

Table 1: Most Popular Applications in the Dataset.

Applications - Packages	Installations	%
com.facebook.orca	37	75.5
com.whatsapp	32	65.3
com.twitter.android	31	63.3
com.google.android.apps.translate	23	46.9
com.facebook.katana	22	44.9
com.spotify.music	22	44.9
com.skype.raider	20	40.8
com.ubercab	20	40.8

tions found on a single device was 283 and the minimum was 19. The number of unique applications that were seen in the collected dataset was 1,983 applications. Figure 2 shows the number of installed applications per device (first boxplot). The most popular applications among the participants of our study can be seen at Table 1.

3.2.2. Declared and Dangerous Permissions

Android applications contain declared permissions in the `AndroidManifest.xml` file. This information is available to the `PackageManager` during run-time. Using the methodology described in Section 2.2 we gathered the declared permissions for each application in our dataset. The average number of permissions declared in the `AndroidManifest.xml` file of each application was approximately 12.39 (with standard deviation = 10.94). The average number of dangerous permissions per application was approximately 3.85 (with standard deviation = 3.21). Distributions of these data are provided in figure 2.

The aforementioned numbers indicate that (on average) 31% of the declared permissions in the `AndroidManifest.xml` file of each application belonged to dangerous permissions groups. The application `com.baidu.appsearch` declared 118 permissions in its manifest file and 19 of them belonged to dangerous groups. Moreover, the application 'Signal Private Messenger' (`org.thoughtcrime.securesms`) declared 20 permissions that belonged to dangerous permissions groups. These applications presented the highest number of declared and dangerous permissions in our dataset, respectively. Table 2 shows applications that declared the highest number of permissions and Table 3 those with the highest number of dangerous permissions in our dataset. Note that some applications can be seen more than once, having different number of declared or dangerous permissions, because the dataset contained different versions of them.

Table 2: Applications that Declared Highest Number of Permissions.

Applications Packages	Permissions	
	Declared	Dangerous
com.baidu.appsearch	118	19
com.qihoo.appstore	94	19
com.sec.android.easyMover	84	12
com.sec.android.easyMover	82	12
com.wandoujia.phoenix2	80	9
com.kingroot.kinguser	80	4
com.lenovo.magicplus	79	18
com.boying.store	75	12

Table 3: Applications that Declared Highest Number of Dangerous Permissions.

Applications Packages	Permissions	
	Declared	Dangerous
org.thoughtcrime.securesms	57	20
net.dinglisch.android.taskerm	51	19
com.baidu.appsearch	118	19
com.qihoo.appstore	94	19
com.kms.free	49	18
com.lenovo.magicplus	79	18
com.lenovo.leos.cloud.sync	53	18
com.sec.chaton	58	17

3.3. Permission Groups

We mentioned previously (in Section 2.2) that we grouped the declared permissions to simulate the user interface of the Settings app when it shows the granted permissions per application (Figure 1e). According to the Android Developers documentation[8], there exist 9 groups of dangerous permissions at API level 23 and above (up to API 27 as of January 2018): Calendar, Camera, Contacts, Location, Microphone, Phone, Sensors, SMS, Storage.

3.3.1. Dangerous Permissions Requests per Device

First, we estimated how many applications on average request access to dangerous permissions groups per device. Hence, we measured the appearance of dangerous permissions in the manifest file of each application per device. In this step we did not consider if the particular groups were granted access permission by the user. We demonstrate the results

[8] http://bit.ly/2d4AdGH

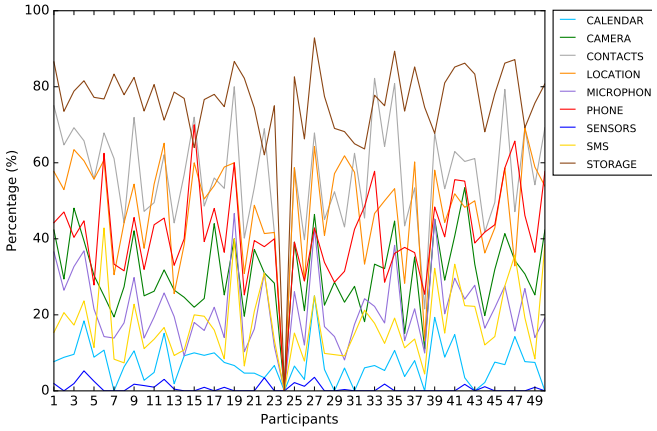


Figure 3: Dangerous Permissions Groups Requests per Device.

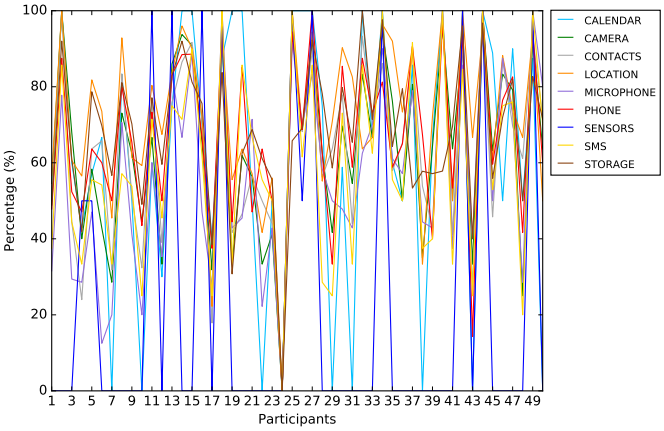


Figure 4: Dangerous Permission Groups Accessibility (%) per Device.

Table 4: Average Use of Dangerous Permission Groups.

Permission Groups	Average Use (%)	S.D.
Calendar	7.34	5.16
Camera	30.46	9.36
Contacts	58.02	12.78
Location	49.67	12.20
Microphone	21.63	9.37
Phone	40.56	10.56
Sensors	0.93	1.19
SMS	16.51	9.23
Storage	76.64	7.18

at Figure 3, which presents the average dangerous permissions requests per device (concatenated as dangerous permissions groups). Figure 3 shows that the Camera, Contacts, Location, Phone and Storage groups were the most requested dangerous groups per device.

In addition, Table 4 presents more generic information, showcasing the average use of dangerous permissions (gathered as groups) and their standard deviations in our dataset. We can see that indeed requests to access the device's storage were made by almost 77% of the applications in our dataset. Other popular resources for applications were the Contact list and the Location; access to these resources were requested by the half of the applications in our dataset.

3.3.2. Permission Groups' Accessibility

Figure 4 provides a rough representation of the permission groups' accessibility. This figure demonstrates the percentage of permission groups that appeared to be accessible from the installed applications per device. The methodology described at Section 2.2 was used to estimate these percentages. As figure 4 shows, users tend to have a stable behavior (considering their security preferences) when dealing with different applications on the same device. For example, more than 80% of the applications on device No 34 could access the distinct permission groups.

However, this indication (that users seem to have a stable behavior when granting access to resources on their devices to various applications) might be skewed by the fact that there existed numerous applications per device that were probably not designed to adhere to the new permission model. Thus, during the transitional period our experiments took place, applications like 'Snapchat' appeared to have access to all dangerous resources, in order to maintain backward compatibility. Such applications can be easily identified if the user tries to revoke access to resources via the Settings app; the system will issue the following message, which is rather preventive in our opinion: "This app was designed for an older version of Android. Denying permission may cause it to no longer function as intended".

Despite that figure 4 provides probably a noisy representation of the accessibility to sensitive resources, it forms an early estimation that users tend to persistently grant access to specific resources and deny it to others. In the next sections we will study closely users' privacy preferences for popular applications.

Table 5: Installations of Social Media Applications in our Sample.

Applications - Packages	Installations	appr. %
com.facebook.orca	37	76
com.whatsapp	32	65
com.twitter.android	31	63
com.facebook.katana	22	45
com.skype.raider	20	41
com.instagram.android	14	29
com.linkedin.android	13	27
org.telegram.messenger	7	14

Table 6: Percentage of Devices with Fine-tuned Permission Settings for Applications in the $\mathcal{S}$ set.

Applications	All On (%)	All Off (%)	F-Gr (%)
Messenger	24	22	54
WhatsApp	44	0	56
Twitter	23	19	58
Facebook	23	9	68
Skype	35	15	50
Instagram	7	29	64
LinkedIn	38	15	46
Telegram	0	14	86

3.4. Fine-grained Permissions on Social Media and Communication Applications

This Section presents users' security preferences, considering fine-grained permission settings for popular social media applications. We focused on social media and messaging applications because they were installed by numerous participants. Other application categories (such as business or travel) are not studied here because we did not have adequate data from enough users that would allow us to make safe conclusions.

3.4.1. Users with Fine-Grained Permission Settings

Table 1 showed the most popular applications in the dataset. Note that most of them are social media and messaging applications. Thus, we focused our study on eight applications and instant messengers: (Facebook) Messenger (`orca`), Facebook (`katana`), Whatsapp, Twitter, Skype, Instagram, Linkedin and Telegram. We will refer to this set of applications as the 'social' set $\mathcal{S}$. Despite Telegram was not among the most installed applications, it was added in this section because it is known for its 'end-to-end' encryption capabilities. Hence, it would be interesting to see users' preferences for this application. Table 5 shows the popularity of the aforementioned applications in our sample (percentage of installations). Facebook Messenger appeared to be more popular than the Facebook application itself, indicating that there exists a considerable number of users that prefer to use only the Messenger application.

Table 6 shows the percentage of devices on which applications from $\mathcal{S}$ were allowed (All On) or "denied" (All Off) access to *all* permission groups, or appeared to have fine-grained settings, respectively (F-Gr). We should note here that in the "denied" category we also included permissions that were never invoked by the apps. According to this table, on average, approximately 60% of devices did not allow access to at least one sensitive resource. However, given that we did not store applications' usage statistics, we cannot assume that access to this resource was requested and eventually was denied by the user. Note that the last row on table 6 shows that no participant granted all permissions to the Telegram application. This finding makes sense considering that Telegram's target group is privacy aware users, which are probably more selective or cautious when dealing with permission settings.

3.4.2. Analysis Based on the F-Gr User Group

To avoid misleading conclusions arising from the possibility that participants might not have used a specific application from the social set $\mathcal{S}$, we only considered (in this and the following section) participants with fine-grained permission settings. Indeed, this assumption might exclude permission data from users who denied access to all resources when they ran the specific applications or they did not ever use the app. However, we believe that the F-Gr user group will give us a better understanding on how users, who care about permissions, act. Figure 5 (a, b, c) shows graphically how our participants tuned their permission settings for the Facebook Messenger, WhatsApp and Uber applications respectively. Green color indicates that permission was granted to the dangerous group and red color indicates that permission was denied or revoked by the user to the dangerous group. Ochre cells indicate that permissions for the specific group were not found on the device. For example, user No 3 had probably installed an older version of the Uber application, which did not include the use of the device's microphone (figure 5c). Results are not affected by the fact that we do not differentiate between older versions of the same application be-

(a) Facebook Messenger (b) WhatsApp (c) Uber

(d) Facebook (e) Twitter (f) Skype

Figure 5: Permission Settings of Users in the F-Gr Group for Various Applications. Green Color Indicates the Resource was Accessible, Red Color Indicates the Opposite. Ochre Color Shows that the Group was not Declared on this Device.

cause these versions were eventually excluded from the F-Gr group (with 2 exceptions, shown in Figures 5c, 5f).

Figure 5 indicates that there exist similarities at users' settings if we observe each subfigure vertically. For example, the majority of Facebook Messenger's users did not provide access to Phone, SMS and Location (94.74%, 89.48% and 73.69%, respectively). On the other hand, Storage, Microphone and Camera were granted access by a large number of participants (84.21%, 84.21% and 73.69%, respectively).

Similarly, Figures 5d and 5e demonstrate users' settings for Facebook and Twitter. Figure 5d showcases that Facebook users did not allow access to Calendar, SMS, Microphone or Phone (93.33%, 93.33%, 80% and 80%, respectively). On the contrary, Facebook had access to the Storage and Location groups in 93.33% and 60% of the devices, respectively. Analogous numbers can be identified among Twitter users. Storage and Location were accessible in 73.33% and 80% of the devices, but in general users seem to be more rigorous regarding giving access to other dangerous permission groups. Note that we do not suggest here that the same user grants or denies the same permissions on Facebook and Twitter.

WhatsApp users in our sample did not allow access to Location and Phone groups (77.78% and 94.44%) but they were more keen to allow access to the SMS group, compared to Messenger users (66.67% and 10.52%). All Uber users allowed access to the Location group (100%) and the majority of applications (except Uber) present high accessibility of the Storage group (84.21%, 100%, 38.89%, 93.33%, 73.33%, 90% respectively). This is an indication that most applications require access to the device's storage in order to provide their basic functionality; hence, users allow them to access this group. Given that users are keener to permit this request entails security risks, because malicious applications may request access to the Storage group in the right context and then misuse it.

Users' preferences may be also linked with their level of trust to the specific application or they may only indicate how users interact with these applications. For example, the fact that the Camera group is in general marked red in both applications might denote that the majority of our participants did not primarily use Facebook or Twitter to take pictures. Thus, the demonstrated accessibility in Figure 5 could possibly be linked with the level of functionality of each application and also with the level of trust the users show to them. Further research work should be done to investigate this question. However, visual schemes like those presented in Figure 5 can be read both vertically and horizontally and provide insights about the use of applications under the run-time model. A vertical view demonstrates how the majority of users tuned their permission settings for each application. A horizontal view shows individual users' preferences. For example, user No 17 seems to have similar behavior when using Messenger and WhatsApp because Phone, Location and SMS groups are marked with red color in both cases. Common permission settings across a number of different applications from the same category can eventually create users' privacy profiles.

3.4.3. A Generic Comparison for Granted Permissions per Application

Figure 6 presents in a concise graph the percentage of users that granted permissions to dangerous groups per application. As previously stated, we only consider participants from the F-Gr group, as seen at Table 6. Figure 6 shows for example that all users allowed the LinkedIn application to access their Contact List (100%). However, applications like Messenger and Facebook demonstrate lower rates of accessibility to this permission group (47% and 33%, respectively). Also, users of messaging applications like Messenger, WhatsApp, Skype and Telegram seem to be more keen to grant access to the Microphone permission group. At the same time, Facebook, Twitter, Uber and Instagram present high numbers of accessibility to Location services.

We can also extract additional common characteristics from figure 6 regarding users' preferences and permission settings. For instance, if we focus on the Storage group we can see that most of the users allow the applications (of our social set $\mathcal{S}$) to access the storage of their devices. However, users' preferences indicate that they are more reluctant to grant storage access to applications like Uber. This finding is probably linked with the functionality of each application and the reasons we use them. Another generic observation can be made for the SMS group. We derive that users are hesitant to allow applications to access their SMS list with the exception of WhatsApp and Telegram. A possible explanation for this result is that both applications provide end-to-end encryption, providing higher levels of trust

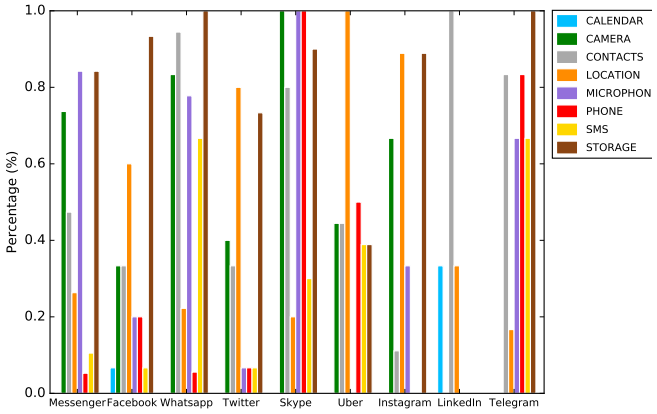


Figure 6: Percentage of Granted Permissions per Application (from $\mathcal{S}$ set) Considering F-Gr Group Participants.

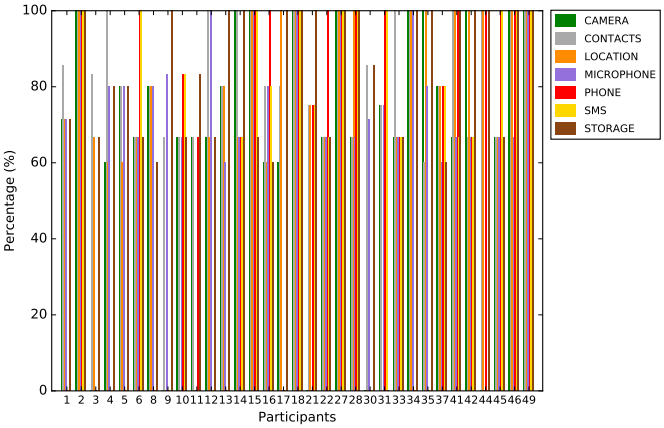


Figure 7: Users' Behavioral Consistency Considering Applications from the $\mathcal{P}$ Set.

to their users. In general, these two applications present high levels of accessibility (70% and above) in almost all dangerous groups. Furthermore, one could note that WhatsApp and Telegram's Location accessibility is not high (approximately 20%). The gathered data are not sufficient to allow us to explain why this occurred. However, the answer to this phenomenon might be hidden in the interplay of trust and functionality. Trust in this context is not only linked with the use of encryption; trust can be also derived from the developer of the app.

3.5. Users' Behavioral Consistency

In this section we study if users present a consistent behavior when they are using different applications on their devices. In other words, we examine if the same participant follows the same behavioral patterns when using the fine-grained permission system on a variety of applications on the same device. In order to do that, we identified the users in our sample who had installed at least 3 of the applications listed at Table 5 on their devices. Note that LinkedIn and Telegram applications were excluded from the analysis for this section, because the former application declares access to only 3 dangerous groups and the latter was installed only on 7 devices. Thus, we are particularly interested in permission-aggressive applications in this context. We will refer to this set of applications as the 'popular' set $\mathcal{P}$. As the goal of this section is to highlight consistent behaviors regarding permission settings preferences, we did not take into consideration the Calendar permissions group, because it is declared only on Facebook. Hence, we evaluate similarities to the way users grant or deny access to specific resources considering at least 3 popular social media applications installed on the same device.

3.5.1. Behavioral Similarity

We define the term 'behavioral similarity' to assess common characteristics of user's behavior related to the permission settings on their devices. *Behavioral similarity* or *behavioral consistency* is defined here as the quality of preserving similar user behavior when various applications (at least 3) request access to a specific permission group. For example, if the user has granted (or denied) access to the Microphone group to at least 60% of social media applications (installed on the same device), then this is considered in this work to indicate behavioral consistency.

Figure 7 demonstrates the behavioral consistency of 33 users in our sample. Here we did not exclude users from the 'All On' and 'All Off' groups, in order to provide a more generic view. The inclusion of these users will probably result in very high values of consistency in rare occasions where all considered applications belong to the 'All On' group. Given that we did not store information that would reveal if installed applications were compiled to abide the new model, we included these users in the analysis. In general, users had installed at least 3 applications from the popular applications set, as discussed previously. The goal of our analysis here is to evaluate if a user who denies (or grants) access to a specific dangerous group for one application, maintains the same behavior, when asked to provide access to the particular dangerous group by another application.

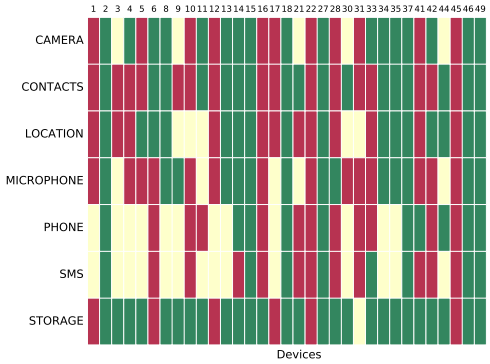


Figure 8: Descriptive Table Showing Participants' Permission Settings Consistency for Applications from the $\mathcal{P}$ Set.

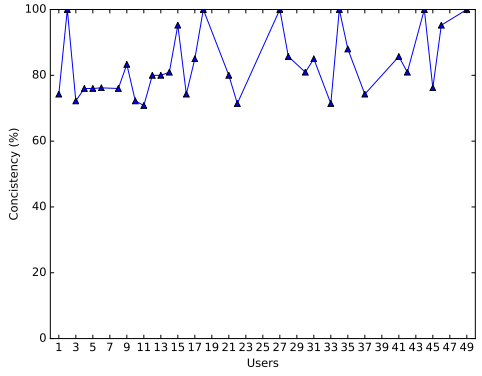


Figure 9: Average Behavioral Consistency of Users (33 Participants).

The calculation of a user's behavioral consistency was performed as follows. Considering the number of applications (from the popular set $\mathcal{P}$) found on a single device, we highlighted which of them were granted or denied access to each permission group. For example, user No 5 had installed 5 applications from the popular set. Access to Camera, Contacts and Microphone was denied to 4 applications among them (80%). On the contrary, access to Storage was granted to 4 applications (80%) and access to Location was granted to 3 applications (60%), as figure 7 shows. We consider as consistent behaviour either the persistent denial or the persistent permittance to a specific resource. Another example is User No 3, who only presented similarities at the Contacts, Location and Storage groups settings (83.3%, 66.67%, 66.67%, respectively). Again, given the lack of usage statistics, we cannot decide if a group was asked to be given access; these permissions are included in the "denied" category.

Figure 8 complements figure 7 and shows the persistence per user in denying or allowing access to resources when prompted by applications of set $\mathcal{P}$. Ochre cells indicate that the user did not present consistent behavior for the specific dangerous group or the applications did not request access to these resources. Figure 8 can be utilized to extract user profiles according to their privacy and security settings. For example, users such as No 13, 34, 35 seem to have similar behaviors when granting access to social media applications. As future work, we aim to identify such privacy profiles.

Finally, we estimated the average values of behavioral consistency per user and the results can be seen at Figure 9. The overall average behavior consistency in our sample is 83.26% and the standard deviation is 10.05. Without a doubt, behavioural consistency seems to be high in our dataset. This probably occurs because the set $\mathcal{P}$ consists of a small number of popular applications, basically from the social media and messaging category. Thus, the elements of similarity in functionality and trustfulness are present in this set. Further work should be done in the future to examine if behavioral consistency is maintained when we compare applications from different categories and if these high numbers of consistency remain the same when we are dealing with larger groups of users.

To conclude, this section presented our analysis on 33 participants, which indicates that users of popular social media in our sample demonstrated consistency, allowing or denying access in a uni-

Table 7: Percentage of Users that Selectively granted permissions to Applications

Participant	**No2**	**No4**	**No5**	**No8**	**No13**	**No25**	**No26**	**No27**	**No32**	**No35**	**No40**	**No46**
Installed apps (2nd phase)	36	43	90	96	250	47	162	34	39	49	83	37
New apps added	4	5	16	1	36	1	14	8	6	2	4	9
Apps removed	2	0	5	0	1	0	18	2	0	0	0	1
Installed apps (1st phase)	34	38	79	95	215	46	166	28	33	47	79	29
Apps with Changed Permissions	0	3	4	0	1	0	0	0	0	2	2	2
Apps with Changed Permissions (%)	0	7.89	5.06	0	0.46	0	0	0	0	4.25	2.53	6.9

Table 8: Details of Revised Permission Settings per User.

Participants	**No4**	**No5**	**No13**	**No35**	**No40**	**No46**
Application 1	*Reddit* Storage ON	*Google Docs* Storage ON	*YouTube* Camera ON	*Google Translate* Camera ON	*OfficeSuite+PDF Editor* Storage ON	*Uber* Microphone ON
Application 2	*Google+* Storage ON	*Root Explorer* **Contacts OFF**		*Google Keep* Contacts ON	*Firefox* Storage ON	*Acrobat Reader* **Camera OFF**
Application 3	*Shazam* Location ON Microphone ON	*Instagram* Microphone ON Camera ON				
Application 4		*Facebook* Location ON Contacts ON				

form fashion to resources which are protected with dangerous permissions.

3.5.2. Behavioral Consistency over Time

The second phase of experiments aimed to analyze if participants' preferences and permissions settings change after a period of time. The application we developed set an alarm to the device, which would go off one month after the application was first launched. In addition, we used the Google Cloud Messaging service to further enhance communication with the devices that still had our application installed (via push notifications).

As mentioned at Section 2.1, we indicated the changes of the permission model to the users by showing a small tutorial at the last part of the first phase. Participants were reminded that they could revoke access to sensitive resources using the Settings application. We suggested the participants to keep the application on their devices after the first phase was completed and participate again, when they see a notification on their phones (or on their Android smartwatches). The notifications were not persistent; they were designed to be erased (clicking the 'Clear All' option on the Notification drawer) if the user did not want to participate for a second time to our experiments. As already noted, our participants were volunteers and we did not want the application to be very pervasive and aggressive.

The participants sent their data during the second phase on a voluntary basis; just like the first phase, no compensation was provided. During the second phase, 24.49% of the participants (12 out of 49 participants that constituted the 'first phase' set of users) send us their permission data. Note that numerous participants uninstalled our application after the first phase, thus we did not get responses from them. We gathered our findings regarding the alterations of permission settings during the second phase at Table 7.

Table 7 shows that the respondents during the second phase installed additional applications on their devices. The average number of installed applications is approximately 80 per device. It was 74 at the first phase (rounded in both cases), if we take into account only these 12 devices. This is indeed a rough estimation, but it shows that participants installed additional applications on their phones in the one-month period between the two phases of our experiments. Hence, they were actually using their devices in this period.

The last row of Table 7 shows the percentage of installed applications having at least one altered permission setting, compared to the first phase settings. On average, we derive that (per device) only 2.26% of the installed applications presented altered permission settings. As a matter of fact, 50% of the respondents' devices did not present any changes at their permission settings. We must stress again that at the end of the first phase, all participants saw a tutorial underlining that Marshmallow users were able to change any permission, for any application, anytime they wanted using the Settings application.

In total, 14 applications were found having altered permission settings in our dataset. Among them, user No 5 revoked access to the Contacts group for 1 application and user No 46 revoked access to the Camera for 1 application (presented in bold at Table 8). On the contrary, 14 applications presented altered permission settings providing access to the following dangerous groups: 5 applications were given access to the Storage group, 3 applications were allowed to access the Microphone and the Camera groups and 2 applications were able to access the Location and the Contacts groups. Details can be seen at Table 8. Table 8 indicates that users at the second phase of our experiments were again more keen to allow access to the Storage group. Additionally, note that only users No 5 and No 46 changed permissions to social media and communication applications from set $\mathcal{P}$.

It would be ideal to repeat the experiments more than once in order to make conclusions that are more robust; hence, we could obtain more than one additional snapshot. A longitudinal study in this context would provide insightful details of users' actions. However, we believe that this strategy would discourage our volunteers to participate and they would be disengaged from the beginning of the experiments. Thus, we chose to proceed with the less pervasive approach of taking one additional permission snapshot from the participants' devices. Also, it would be interesting to have the ability to interview participants who returned a second set of permission data in order to better understand why users adjusted their apps' privacy settings a month later or why they did not do that. However, received data were anonymous, hence we do not have this opportunity. We recognize that this is a limitation of our pilot study but future work will account for that.

As a conclusion, results presented in this section demonstrated that users' initial permission settings

did not radically change in the second snapshot. Half of the users from the second snapshot did not change any permission settings for the installed applications. The other users just changed a small part of the initial settings. Also, the results show that our participants are more willing to provide access to their devices' storage, when an application initiates such a request.

3.6. ChatON Tokens

While analyzing collected permission data we made an interesting observation. Permission data derived from devices having installed the 'ChatON' application – i.e. devices No 2 and No 40 – included the following permission: `com.sec.spp.permission.TOKEN_` . This was followed by a unique token (a 256 character hexadecimal number). We downloaded the application on a rooted Nexus 7 (2012) tablet that was running a custom ROM (Marshmallow version) and experimented with it. We used this device to send permission data and we marked the (token) number that was attached in the `com.sec.spp.permission.TOKEN_` permission for the given device. Then the device was restarted and the ChatON application was relaunched. The token was the same, which means that this number is persistent. Despite that the ChatON service is no longer available[9], this token can be used by third party applications (just like 'Permissions Snapshot' did) to identify a device, just by using the public `PackageManager` class. This forms an example of bad coding practice which should be avoided, because it introduces vulnerabilities related to privacy; these tokens might be used for the identification of a device.

4. Discussion

Android mobile OS introduced a significant enhancement to its access control system with the advent of the Marshmallow version. Previous work [18] showed that users would prevent permission requests from applications if they had this ability. The former generation of the Android OS did not provide this opportunity to the users. Therefore, all permission requests from an application were accepted by default, if users agreed to install it on their devices. Our work, validates the results of [18] using data derived from actual devices running Android 6.0+. In addition, the results we presented in this paper (regarding permission groups' accessibility rates) were confirmed by another research work, which gathered data from a different group of participants [19].

The current study shows that it is possible to gain information about users' permission settings utilizing publicly accessible classes such as the `PackageManager` class. This methodology allows third-party applications to acquire information about how individuals tune their privacy settings in their personal digital ecosystems. Additional data (such as `firstInstallTime, lastUpdateTime, targetSdkVersion`) can be acquired using the `PackageManager` to enrich our knowledge about installed applications. Collected information can be assembled to depict users' privacy profiles, as Figures 5 and 9 demonstrated. Therefore, this work demonstrated that under the run-time permission model it is possible to create anonymous user profiles. These profiles could be used by online vendors to further personalize their digital application stores. In addition, personal privacy profiles can be used by artificial intelligence agents, which are embedded in the most modern mobile operating systems, in order to optimize recommendations to individual users.

The results of our pilot study initiate a discussion about how effective the new model is in providing additional security to the user. We have seen that most applications request access to device's storage and the majority of users allow them to access it. However, we have noticed that, on average, 60% of the participants used fine-grained settings on their devices, which is a positive step towards security and privacy enhancement. On the other hand, data derived from a small set of 12 participants who submitted permission data for a second time, showed that only one permission was revoked on participants' No 5 and No 46 devices, respectively. This finding complements and validates recent work which examines the lack of continuance of secure behavior from users [20].

4.1. Limitations

Although we attracted a considerable number of participants for this pilot study, we did not make extensive use of all collected data. This happened because we did not get detailed information about how often an application is used on the specific device, or if it was actually launched at least once be-

[9] http://bit.ly/MMG2pP

fore data collection occured. Therefore, we included at the second part of our analysis only participants who had used fine-grained settings for their applications, excluding others that might continuously prevent applications to access sensitive resources. Thus, we could not reliably identify if there exist users that consistently deny access to specific applications. However, in future work we will account for this problem, engaging respondents to provide usage statistics. This will be achieved by recruiting participants from crowd sourcing platforms, such as Amazon Mechanical Turk or Microworkers. The current work can be considered as a pilot study on real users' devices that initiates a discussion about how Android users adjusted to the fine-grained access control paradigm.

Additionally, although we distributed 'Permissions Snapshot' via Google Play, our study was conducted during a period when the majority of Android users owned devices that were running older versions of the OS. Therefore, the participants of this study were mainly males and probably most of them were tech savvy individuals, owning Google Nexus smartphones and tablets. Unsurprisingly, our sample included users having super user privileges on their devices. However, we also received data from participants with different profiles (e.g. older people or users with limited number of installed applications), hence the behavioral patterns demonstrated in this paper should not be linked with specific groups of people.

Moreover, we should acknowledge the fact that when it comes to questions related to privacy enhancement, most of the users would probably agree that they prefer more privacy. Thus, the reader should take into account that users often lean towards utility than privacy [12] and should be careful when making extrapolations from the results. Further work in the future will include a larger and more diverse sample of users, given that the run-time permission model is currently utilized in 3 Android versions (Marshmallow, Nougat, Oreo). We also plan to assess users' privacy and security profiles using frameworks like [21].

5. Related Work

Previous work highlighted privacy- and security-related concerns associated with Android's install-time application permission system, such as the proliferation of over-privileged applications [22]. In a recent study, Pearce et al. [23] showed that approximately 49% of the Android applications at the Android Market were over-privileged because of the use of advertising libraries. Moreover, the authors indicate that such pervasive over-privileging constitutes a threat to users' privacy.

Prior work that studies user-defined privacy and security controls on mobile devices showed that users (on average) have to make over a hundred permission decisions per device [7]. In addition, researchers demonstrated than users are often unaware of the number (or the context) of permissions they granted to applications at the past [24]. Jeon et al. [25] proposed a framework to address security issues that arise from the (old) coarse-grained permission model. They used a taxonomy of four main Android permissions groups and developed a fine-grained permission model to increase the security of existing apps without affecting functionality and without requiring platform modifications.

A recent paper by Wijesekera et al. [18] showed that during a study, at least 80% of the participants indicated that they would prevent at least one permission request of their experimental application if they were aware of its purpose and functionality. The participants also stated that they would block over a third of permission requests if they had this choice. However, other studies highlighted that most of the users do not pay attention to system messages related to permission requests [26]. Additionally, researchers demonstrated that users were often surprised by the ability of applications to collect personal data in the background and share data with third parties [27]. Advertising libraries for example have been consistently examined for data exposure and leakages; a recent study revealed a trend in advertising networks to become more aggressive in collecting reachable user data [28].

In the previous years, a number of security extension mechanisms have been proposed to overcome privacy constrains of the previous permission model [29]. Some approaches add access control processes to provide the user the ability to choose permissions and reduce run-time constrains for each application [30]. Other methodologies introduce fine-grained access controls for Android applications aiming to enforce control over the assignment of permissions through explicit policies [31]. Moreover, FlaskDroid [29] extended the latter framework to provide a flexible, fine-grained access control mechanism for diverse security policies. Furthermore, TaintDroid [32] was designed to track tainted

data from essential sources on Android and detect unauthorized data leakage.

In order to protect mobile devices from unlimited data sharing, several systems have been proposed. Some approaches focus on location services [33], offering its users the ability to adjust location accuracy when using location-based applications. At the same time, on-device or service-based obfuscation techniques have been advised as a methodology to maintain users' privacy [34]. Such schemes according to [35], that utilize abstract location descriptions, may cause location sharing to increase. Beresford et al. [36] presented their system, named MockDroid, which essentially feeds 'empty' resources or 'fake' information to apps that require access to sensitive data [25]. Another data protection mechanism that relies on obfuscation is AppFence [37]; this system substitutes shadow data in place of data that the user wants to keep private. In addition, there exist methods based on crowdsourcing, which aim to utilize contributors' protection decisions in order to provide application specific privacy recommendations for iOS devices [38].

Fine-grained permissions were indeed first introduced earlier on another platform (iOS 6). However, our current work assesses for the first time security settings of users based on data derived from their own devices, which were not modified for the needs of this study. In this work we assume that participants do not share their devices with other people and we intent to highlight behavioral patterns related with their perceptions of security. In this paper we presented preliminary results based on a small set of users. However, we believe that the approach we used in tandem with an improved version of the survey app, which collects more contextual information (following principles presented in other research works [3]), is promising and will assist us to identify privacy preferences of Android users.

6. Conclusions

To conclude, in this paper we discussed the results of our pilot study that investigates Android users' security and privacy settings and preferences under the run-time permission model. The responses from our participants indicate that these users maintain a positive view about this change. Most of them claimed they prefer the new system and they believe that now they can control easily the sensitive data they share. We also demonstrated that, in general, one third of the requested permissions in our sample belonged to dangerous permission groups. Additionally, we showed that Storage, Contacts, Location, Phone and Camera were the most requested dangerous permissions groups from the installed applications in the participants' devices. Furthermore, considering the accessibility of particular applications to resources, we highlighted the persistence of users to allow access to groups that are directly related to their main functionality (e.g. Camera for Instagram or Contacts for LinkedIn). Moreover, we showed that (in general) users' behavior is consistent regarding the resources they allow to social media applications to access. However, additional research work needs to be done to assess users' trust to specific applications. Finally, after collecting permission data from the same sample of participants after a period of one month, we found out that 50% of the respondents had not changed any permission settings on their devices. This might occured either because they did not request additional functionality from the apps, or because they did not use the apps during the one month period. Furthermore, only 2.26% of the installed applications (on average) presented different permission settings. Hence, we conclude that although Android (starting from version 6.0+) provides the ability to its users to control more efficiently resources used by applications, our participants did not utilize this functionality widely. In the future we wil address the limitations of this pilot study by capturing usage data to stress the context within the users' decisions are made.

References

[1] S. Rosen, Z. Qian, Z. M. Mao, Appprofiler: A flexible method of exposing privacy-related behavior in android applications to end users, in: Proceedings of the Third ACM Conference on Data and Application Security and Privacy, CODASPY '13, ACM, New York, NY, USA, 2013, pp. 221–232. doi:10.1145/2435349.2435380.

[2] P. Andriotis, T. Tryfonas, Impact of User Data Privacy Management Controls on Mobile Device Investigations, Springer International Publishing, Cham, 2016, pp. 89–105. doi:10.1007/978-3-319-46279-0_5.

[3] P. Wijesekera, A. Baokar, L. Tsai, J. Reardon, S. Egelman, D. Wagner, K. Beznosov, The feasibility of dynamically granted permissions: Aligning mobile privacy with user preferences, in: 2017 IEEE Symposium on Security and Privacy (SP), 2017, pp. 1077–1093. doi:10.1109/SP.2017.51.

[4] L. Tsai, P. Wijesekera, J. Reardon, I. Reyes, S. Egelman, D. Wagner, N. Good, J.-W. Chen, Turtle guard: Helping android users apply contextual privacy prefer-

ences, in: Symposium on Usable Privacy and Security (SOUPS), 2017.

[5] B. Rashidi, C. Fung, E. Bertino, Android resource usage risk assessment using hidden markov model and online learning, Computers & Security 65 (2017) 90–107.

[6] M. Frank, B. Dong, A. P. Felt, D. Song, Mining permission request patterns from android and facebook applications, in: Data Mining (ICDM), 2012 IEEE 12th International Conference on, IEEE, 2012, pp. 870–875.

[7] B. Liu, M. S. Andersen, F. Schaub, H. Almuhimedi, S. A. Zhang, N. Sadeh, Y. Agarwal, A. Acquisti, Follow my recommendations: A personalized privacy assistant for mobile app permissions, in: Twelfth Symposium on Usable Privacy and Security (SOUPS 2016), USENIX Association, Denver, CO, 2016, pp. 27–41.

[8] B. Rashidi, C. Fung, T. Vu, Dude, ask the experts!: Android resource access permission recommendation with recdroid, in: Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on, IEEE, 2015, pp. 296–304.

[9] S. E. Hormuth, The sampling of experiences in situ, Journal of personality 54 (1) (1986) 262–293.

[10] Xposed Module Brings Back App Ops on Android 4.4.2 to Give Your Control of Your Application Permissions, `https://goo.gl/MvkJ8U`, Accessed: 21-01/2018.

[11] P. Andriotis, M. A. Sasse, G. Stringhini, Permissions snapshots: Assessing users' adaptation to the android runtime permission model, in: Information Forensics and Security (WIFS), 2016 IEEE International Workshop on, IEEE, 2016, pp. 1–6.

[12] P. Andriotis, G. Oikonomou, A. Mylonas, T. Tryfonas, A study on usability and security features of the android pattern lock screen, Information and Computer Security 24 (1) (2016) 53–72. `doi:10.1108/ics-01-2015-0001`.

[13] P. J. Dorman, J. Slattery, B. Farrell, M. S. Dennis, P. A. Sandercock, A randomised comparison of the euroqol and short form-36 after stroke, Bmj 315 (7106) (1997) 461.

[14] P. V. Marsden, J. D. Wright, Handbook of survey research, Emerald Group Publishing, 2010.

[15] J. A. Krosnick, Questionnaire design, in: The Palgrave Handbook of Survey Research, Springer, 2018, pp. 439–455.

[16] J. A. Krosnick, Survey research, Annual Review of Psychology 50 (1) (1999) 537–567. `doi:10.1146/annurev.psych.50.1.537`.

[17] S. Parkin, K. Krol, I. Becker, M. A. Sasse, Applying cognitive control modes to identify security fatigue hotspots, in: Twelfth Symposium on Usable Privacy and Security (SOUPS 2016), USENIX Association, Denver, CO, 2016.

[18] P. Wijesekera, A. Baokar, A. Hosseini, S. Egelman, D. Wagner, K. Beznosov, Android permissions remystified: A field study on contextual integrity, in: 24th USENIX Security Symposium (USENIX Security 15), USENIX Association, Washington, D.C., 2015, pp. 499–514.

[19] P. Andriotis, S. Li, T. Spyridopoulos, G. Stringhini, A comparative study of android users' privacy preferences under the runtime permission model, in: International Conference on Human Aspects of Information Security, Privacy, and Trust, Springer, 2017, pp. 604–622.

[20] P. J. Steinbart, M. J. Keith, J. Babb, Examining the continuance of secure behavior: a longitudinal field study of mobile device authentication, Information Systems Research 27 (2) (2016) 219–239.

[21] N. K. Malhotra, S. S. Kim, J. Agarwal, Internet users' information privacy concerns (iuipc): The construct, the scale, and a causal model, Information systems research 15 (4) (2004) 336–355.

[22] A. P. Felt, E. Chin, S. Hanna, D. Song, D. Wagner, Android permissions demystified, in: Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS '11, ACM, New York, NY, USA, 2011, pp. 627–638. `doi:10.1145/2046707.2046779`.

[23] P. Pearce, A. P. Felt, G. Nunez, D. Wagner, Addroid: Privilege separation for applications and advertisers in android, in: Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, ASIACCS '12, ACM, 2012, pp. 71–72. `doi:10.1145/2414456.2414498`.

[24] H. Almuhimedi, F. Schaub, N. Sadeh, I. Adjerid, A. Acquisti, J. Gluck, L. F. Cranor, Y. Agarwal, Your location has been shared 5,398 times!: A field study on mobile app privacy nudging, in: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI '15, ACM, New York, NY, USA, 2015, pp. 787–796. `doi:10.1145/2702123.2702210`.

[25] J. Jeon, K. K. Micinski, J. A. Vaughan, A. Fogel, N. Reddy, J. S. Foster, T. Millstein, Dr. android and mr. hide: Fine-grained permissions in android applications, in: Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, SPSM '12, ACM, 2012, pp. 3–14. `doi:10.1145/2381934.2381938`.

[26] A. P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, D. Wagner, Android permissions: User attention, comprehension, and behavior, in: Proceedings of the Eighth Symposium on Usable Privacy and Security, SOUPS '12, ACM, New York, NY, USA, 2012, pp. 3:1–3:14. `doi:10.1145/2335356.2335360`.

[27] J. Jung, S. Han, D. Wetherall, Short paper: Enhancing mobile application permissions with runtime feedback and constraints, in: Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, SPSM '12, ACM, New York, NY, USA, 2012, pp. 45–50. `doi:10.1145/2381934.2381944`.

[28] S. Demetriou, W. Merrill, W. Yang, A. Zhang, C. A. Gunter, Free for all! assessing user data exposure to advertising libraries on android, NDSS 2016.

[29] S. Bugiel, S. Heuser, A.-R. Sadeghi, Flexible and fine-grained mandatory access control on android for diverse security and privacy policies, in: Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13), USENIX, Washington, D.C., 2013, pp. 131–146.

[30] M. Conti, V. T. N. Nguyen, B. Crispo, CRePE: Context-Related Policy Enforcement for Android, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 331–345. `doi:10.1007/978-3-642-18178-8\_29`.

[31] M. Ongtang, S. McLaughlin, W. Enck, P. McDaniel, Semantically rich application-centric security in android, Security and Communication Networks 5 (6) (2012) 658–673. `doi:10.1002/sec.360`.

[32] W. Enck, P. Gilbert, S. Han, V. Tendulkar, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, A. N. Sheth, Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones, ACM Trans. Comput. Syst. 32 (2) (2014) 5:1–5:29. `doi:10.1145/2619091`.

[33] M. Benisch, P. G. Kelley, N. Sadeh, L. F. Cranor, Capturing location-privacy preferences: quantifying accuracy and user-burden tradeoffs, Personal and Ubiquitous Computing 15 (7) (2011) 679–694. doi:10.1007/s00779-010-0346-0.

[34] B. Henne, C. Kater, M. Smith, M. Brenner, Selective cloaking: Need-to-know for location-based apps, in: Privacy, Security and Trust (PST), 2013 Eleventh Annual International Conference on, 2013, pp. 19–26. doi:10.1109/PST.2013.6596032.

[35] K. Tang, J. Hong, D. Siewiorek, The implications of offering more disclosure choices for social location sharing, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '12, ACM, New York, NY, USA, 2012, pp. 391–394. doi:10.1145/2207676.2207730.

[36] A. R. Beresford, A. Rice, N. Skehin, R. Sohan, Mockdroid: Trading privacy for application functionality on smartphones, in: Proceedings of the 12th Workshop on Mobile Computing Systems and Applications, HotMobile '11, ACM, New York, NY, USA, 2011, pp. 49–54. doi:10.1145/2184489.2184500.

[37] P. Hornyack, S. Han, J. Jung, S. Schechter, D. Wetherall, These aren't the droids you're looking for: Retrofitting android to protect data from imperious applications, in: Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS '11, ACM, New York, NY, USA, 2011, pp. 639–652. doi:10.1145/2046707.2046780.

[38] Y. Agarwal, M. Hall, Protectmyprivacy: Detecting and mitigating privacy leaks on ios devices using crowdsourcing, in: Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys '13, ACM, New York, NY, USA, 2013, pp. 97–110. doi:10.1145/2462456.2464460.