To Allow, or Deny? That is the Question

Panagiotis Andriotis[1][0000−0002−3490−3123] and
Atsuhiro Takasu[2][0000−0002−9061−7949]

[1] University of the West of England,
Computer Science Research Centre, Bristol, U.K.
panagiotis.andriotis@uwe.ac.uk
[2] National Institute of Informatics,
Digital Content and Media Sciences Research Division, Tokyo, Japan

Abstract. The Android ecosystem is dynamic and diverse. Controls have been set in place to allow mobile device users to regulate exchanged data and restrict apps from accessing sensitive personal information and system resources. Modern versions of the operating system implement the run-time permission model which prompts users to allow access to protected resources the moment an app attempts to utilize them. It is assumed that, in general, the run-time permission model, compared to its predecessor, enhances users' security awareness. In this paper we show that installed apps on Android devices are able to employ the systems' public assets and extract users' permission settings. Then we utilize permission data from 71 Android devices to create privacy profiles based on users' interaction with permission dialogues initiated by the system during run-time. Therefore, we demonstrate that any installed app that runs on the foreground can perform an endemic live digital forensic analysis on the device and derive similar privacy profiles of the user. Moreover, focusing on the human factors of security, we show that although in theory users can control the resources they make accessible to apps, they eventually fail to successfully recall these settings, even for the apps that they regularly use. Finally, we briefly discuss our findings derived from a pen-and-paper exercise showcasing that users are more likely to allow apps to access their location data on contemporary mobile devices (running version Android 10).

Keywords: Human factors · Live analysis · Mobile Computing · User Profiling · Location · Android 10

1 Introduction

We live in an ever-changing digital world and we constantly benefit from technological advancements. Mobile computing is an unambiguous example of the merits we enjoy. Mobile devices are integral parts of our routines and, as a consequence, they hold and transmit voluminous amounts of our personal data. Unsurprisingly, since the earliest years of the mobile computing era we encountered malicious entities attempting to circumvent security and privacy controls that

were set to protect users' information. The lessons learned from recent years, regarding users' inability to effectively protect themselves from intrusive and malevolent actors, led system developers to the implementation of privacy controls that would allow users to easily monitor and regulate the apps' accessibility to sensitive data, sensors and system resources.

Four years after the introduction of the current access control system on Android, which uses request dialogues during run-time (ask-on-first-use, a.k.a AOFU [22]), and a few months after the advent of the anticipated finer-grained model for managing location accessibility while the app is in the foreground (version 10), there still existed approximately 25.2% users that access the Android Play Store on devices running legacy versions of the OS (5.1 and below), as reported on [4]. However, three quarters of Android users (who visit the Play Store) are now familiar with the AOFU model.

This paper reports the results of a study we conducted recruiting 71 participants to voluntarily provide access to the permission settings on their Android devices. For this cause we developed an app that is able to instantly gather appropriate information while it is active, i.e. while it runs on the foreground. Given that any benign app in the Android ecosystem is capable of performing similar data collection, we consider our prototype as a potential tool, able to perform an endemic live digital forensic analysis and extract the current state of the permission settings on the device. This information might be useful for the analyst as it will provide the capability to perform user profiling and acquire some fundamental information about the user's security awareness. Furthermore, we conducted a survey (using the app we deleveloped) asking our volunteers to answer two basic questions, aiming to investigate the following research questions.

- **RQ1**: Which sensitive resources on their devices users aim to protect more frequently?
- **RQ2**: Do they change their privacy-related perceptions when they are dealing with their favorite apps?
- **RQ3**: Are users aware of the permission settings on their devices?
- **RQ4**: Can we categorize users according to their privacy/permission settings?

Therefore, the main contributions of this paper are as follows:

a) We gather system related information from the actual devices used daily by our participants, resulting in the acquisition of high quality permission data which are further used to create representative privacy profiles.
b) We demonstrate that Android users are sceptical about providing access to sensitive resources such as their SMS, microphones and contact lists. However, they become more permissive with their favorite apps; this action is related to the anticipation to gain benefits from the advanced functionality and is based on the foundations of trust.
c) We show that users who are now familiar with the AOFU model are still not fully aware of the resources their favorite apps are accessing on their devices.

d) Finally, we demonstrate that the finer grain location settings introduced in Android 10 will probably positively affect users' intention to allow apps to access their location data.

The rest of this paper is structured as follows. Section 2 discusses recent related work on users' acceptance of the AOFU model. In section 3 we present the methodology we used to collect permission settings from live devices and we also discuss elements of our survey design and implementation. Section 4 reviews the collected information and section 5 analyzes the results. Finally, we discuss our findings in section 6 and draw our conclusions in section 7.

2 Related Work

Prior work that investigates mobile phone permission controls and dialogues has shown that not only users do not pay attention to them, but they also cannot comprehend them [9]. The ask-on-install (AOI) permission model (used on legacy Android OS versions) might also cause frustration to users who feel they do not have control on the personal data they share [8]. Additionally, the AOI system presents the inherit disadvantage that users are not given any contextual information about how and when apps access their sensitive resources [20].

These drawbacks undermine users' secure interaction with the system and therefore novel approaches have been adopted to address them. The AOFU model was long-anticipated and it was initially well-received by Android users [1, 3]. However, users' engagement in decision making when they are dealing with the access control management of their mobile devices might lead to the problem of habituation [23]. In addition, although the AOFU model provides some context in the foreground and allows users to make informed decisions, especially at the beginning of the apps' lifespan, it can also be error prone [11, 22]. However, users appreciate the fact that they have dynamically been made part of the security chain in the AOFU model and they take into account the "when" and "why" an app requests permissions [11, 21].

We have seen numerous papers investigating users' adoption and acceptance of this model [2, 6, 7, 10, 16, 18, 19, 21, 23]. Andriotis et al. [1, 3] recently introduced a method to acquire snapshots of permission settings from Android devices and showed that, in general, users make consistent choices when it comes to allowing access to specific sensitive resources. Although malicious actors can employ side-channel attacks to gain unauthorized access to sensitive resources bypassing the Android system's controls [17], users have a more positive attitude towards the run-time permission model [18]. In order to simplify and enhance its effectiveness, various researchers suggest the accumulation of users' privacy profiles [13, 14].

To this end, we use an updated approach of the aforementioned methodology [1–3] to acquire permission snapshots via an app that has to be installed on the users' device. Our scenario/threat model accounts for the fact that any installed app can periodically acquire similar snapshots (while in use) to effectively

create representative users' privacy profiles. Therefore, the app can conduct a live forensic analysis to perform a comprehensive user profiling.

3 Methodology

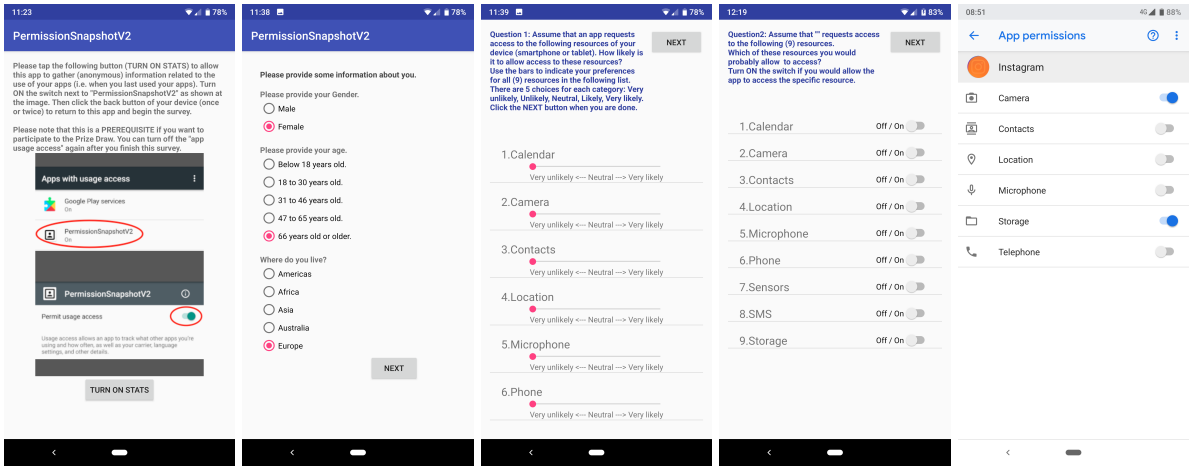
We follow the data collection approach discussed in [3] to collect users' permission settings. First, we develop an Android app that will be used as a survey instrument and at the same time it will collect the current permission settings of the running device. We distribute our application on Google Play and request participants to download the app on their devices. We target users with devices that run Android 6.0 and above, i.e. they implement the run-time permission model. The participants were recruited after viewing our call on various online platforms such as popular social media and university email lists. We did not compensate the respondents for their engagement but we gave them the chance to be included in a prize draw, if they were willing to provide their email address to communicate with them in case they won a prize. We finally got responses from 71 individual Android users from around the world. The project was carried out after ethical approval was acquired by the our RBI (FET Faculty Research Ethics Committee of the University of the West of England (FET.17.03.027)).

3.1 Permission Settings Collection

Our redesigned app utilizes the `PackageManager` and employs the `GET_META_DATA` flag to query the participant's device and acquire a list of application information. Then we use the method `getPackageInfo` with the `GET_PERMISSIONS` and `FLAG_SYSTEM` flags to retrieve non-system applications, i.e. apps that were installed by the users from online app stores. This distinction on the acquired data provides a more accurate representation of users' permission settings because we target only apps that were installed by them. Therefore there is a better probability for creating more representative profiles because we are primarily based on apps that have been used at least once (details in Section 5). This is a fundamental improvement compared to the previous work [1–3].

To ensure that this hypothesis stands true, we employ the `UsageStatsManager` to collect app usage information provided by the Android system itself. Furthermore, we store locally on the phone for each app the `requestedPermissions` and `requestedPermissionFlags`, along with additional information such as the: `versionCode, firstInstallTime, LastUpdateTime,` and `targetSdkVersion`.

The `PackageManager` returns the integer 3 if the permission has been granted and the integer 1 if not. Following this methodology we are able to acquire a snapshot of the user's permission settings. Note that –using this methodology– if the integer that was returned is 1, we do not know if a dangerous permission has been requested by the app in the past. We can only infer that the specific app does not have permission to access the given resource currently. Therefore, following this approach we are able to reconstruct current permission settings on the device for each installed app. In other words we are able to reconstruct



(a) Usage Stats (b) Demographics (c) Question 1 (d) Question 2 (e) Groups

Fig. 1. Screenshots Showcasing "Permissions Snapshots V2" Application and System's Functionality.

information given by the system when the user engages the *Settings* app and requests to see the "App permissions" from the "App info" utility, as seen in Figure 1e.

3.2 Questionnaire Design

The data collection process and the users' engagement lifecycle is described in detail below. First, participants download the app on their devices. After launching the app, consent is given to the app by the user to collect permission settings information. Then the participant is asked to allow the app to collect usage statistics from the device (Figure 1a). This functionality must be explicitly given by the user on our targeted devices. However we provide the users the capability to skip this step, if they do not feel comfortable providing this amount of data to a third party (i.e. to our app).

Afterwards the participants are asked to provide basic demographics (Sex, Age, and Residency as seen in Figure 1b) and answer 2 questions (Figures 1c, 1d). The first question asked the following: "Assume that an app requests access to the following resources of your device (smartphone or tablet). How likely is it to allow access to these resources? Use the bars to indicate your preferences for all (9) resources in the following list. There are 5 choices for each category: Very unlikely, Unlikely, Neutral, Likely, Very likely". A five-point Likert scale [12] was implemented as a slider to store participants' preferences for each dangerous permission ranging from "Very unlikely" to "Very likely".

Next our survey app asked the participants to provide the name of an app they regularly use: "The second (and last) question is related to your favourite

app. Please tap on the following text field and provide the name of an app that you regularly use on this device. Then hit the NEXT button to see the second question". After providing the name of their preferred app, they read the second question: "Assume that "your_favourite_app" requests access to the following (9) resources. Which of these resources you would probably allow to access? Turn ON the switch if you would allow the app to access the specific resource." The activity contained a sequence of switches representing the state of access privileges the specific user would be willing to provide to the specific app, as seen in Figure 1d.

Finally, respondents were instructed how to participate in the prize draw and turn off this app's Usage Access privilege after submitting their answers. The rationale behind the design of our short questionnaire is to identify if there exist deviations between the users' ideal privacy preferences (Question 1) and the amendments they are willing to do when they need to enjoy certain functionalities of their favorite apps (Question 2).

4 Data Analysis

The majority of the respondents of our call provided complete survey answers. Additionally the majority provided their app usage data to our app allowing access to the `UsageStatsManager`. However, there was a small proportion of users that did not allow this action. Additionally, we identified responses from 3 participants that seemed ambiguous. For example, these respondents sent more than one responses while our app was available on Google Play. Therefore, their data were completely removed from the dataset. More details are given in Section 5 below.

Data analysis has been performed in two stages. First we accumulated valid survey responses from participants as explained in Section 5. We refer to this group of participants as G_s in this paper. Then, we compiled another set of data depending only on the permission settings that were sent from the devices to us. These data do not depend on the survey responses as they are actual representations of the permission settings on the participants' devices the given time (Permissions Snapshots); we call G_d this group in this paper.

In order to translate permission data from each device and reconstruct the permission group settings for each app as shown to any user from the Android system (e.g. Figure 1e), we are using the same methodology presented in [3]. We focus on dangerous permissions groups and simulate the way Android handles run-time permission requests to allow or deny access to sensitive resources. Hence, permission settings for each app are represented as a sequence of nine "Allow" or "Deny" decisions. The number nine represents the number of dangerous groups, according to the official classification. This classification depends on the Android API level. While collecting our data, the highest available API level was 27 (i.e. devices running up to version O, codenamed as Oreo). The given time, there existed nine dangerous groups. From API level 28 (Android Pie) another group was added (namely `CALL_LOG`) which practically included some of the

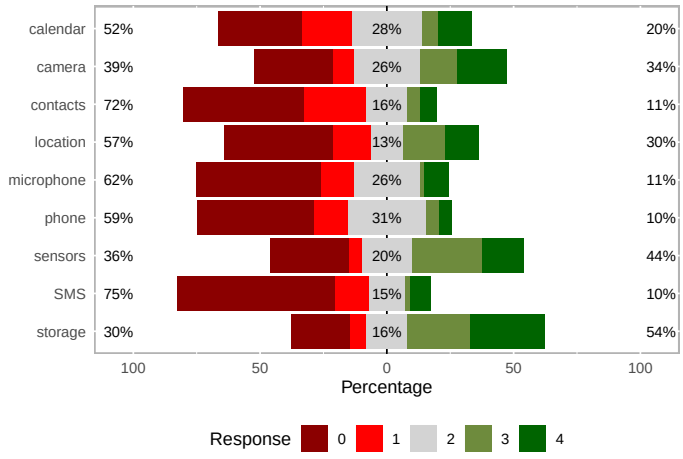


Fig. 2. Responses to Question 1: (0: Very unlikely - 4: Very likely).

older permissions from the `PHONE` group. Since our app was running on devices with OS versions up to Android Pie, we present results based on the nine group classification.

5 Results

We received responses from 71 individuals. Among them, 60 allowed access to the `UsageStatsManager`, 8 did not turn on the Usage access switch when requested by our app, and 3 provided ambiguous responses, therefore their data were removed from our study. Additionally, data from one participant were rejected because she claimed she was below 18 years old. Based on the ethical approval terms, respondents had to be 18 years old and above to participate.

5.1 RQ1: Which sensitive resources on their devices users aim to protect more frequently?

For the first part of this study (regarding the questionnaire responses) we analyzed survey data provided by 61 participants (group G_s). We rejected the answers from individuals who either they did not provide the name of an app for question 2 or they provided a name that could not be found in the corresponding packages provided by their devices' `PackageManager`. Additionally, some of these participants provided responses baring the default answers only, which made us consider they did not sincerely answer the questions; hence their responses were also removed.

As shown in Figure 2 respondents in general are reluctant to allow access to their devices' resources. However, they are more positive to allow access to apps

requesting to use their devices' "Storage" and "Sensors". This finding aligns with recently presented work [2]. The figure also demonstrates that Android users are hesitant to allow apps accessing their: a) SMS messages (75% negative answers), b) their contact lists (72% negative answers), and, c) their devices' microphones (62% negative answers). Another noteworthy finding is that responses related to the camera access were almost equally divided (39% negative, 34% positive). On the other hand, participants are disinclined to allow access to their location data (57% negative answers).

Another point related to location data access is that the answers for the certain permission group presented strong polarity between negative and positive views (13% neutral answers). The decreased percentage of neutral views for this group showcases that mobile device users are aware of the importance of their location data, therefore they have clear views when it comes to sharing them with third parties. Compared to similar previous studies [2, 3] we identify analogous behavior considering users' acceptance of possible requests from the system. In these studies most of the users do not intend to allow access to their SMS, microphones, contact lists, phone logs and location. Our results showcase that these trends haven't changed a lot since the arrival of the run-time permission model three years ago. Additionally, compared to the previous studies we can see that users nowadays have a stronger perception about which protected resources are willing to allow external apps to access.

5.2 RQ2: Are users changing their privacy-related perceptions when they are dealing with their favorite apps?

Next we investigate if users change their behavior when their *favorite apps* request to access protected resources. For this case we focus on the G_s group and gather the answers of the second survey question to compare them with the answers from the first question. We consider as positive the "Likely" and "Very likely" answers and as negative the "Unlikely" and "Very unlikely" answers from the first survey question. Then we count the positive answers representing the resources (i.e. the dangerous groups) they are more positive to allow an app to access. This information is derived from their answers to the first question. Similarly, we count the resources they would allow their favorite app to access, according to their answers to the second question.

Figure 3 shows how participants answered. The blue line shows the number of resources they feel more comfortable to allow an app to access in general, and the orange line shows their responses for their favorite app. We can see that in most cases users would allow more resources to be accessed by their favorite apps compared to their generic response provided to the first question. 18% of the participants provided the same number of positive answers and number of accessible sensitive resources.

In general, from Figure 3 we can infer that users are inclined to allow access to a larger number of sensitive resources when prompted by their favorite apps.

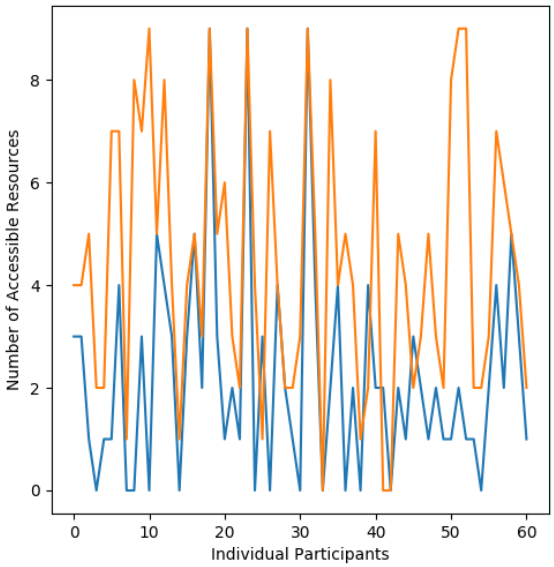


Fig. 3. Deviations among users' survey answers (comparing Q1 vs Q2)

5.3 RQ3: Are users aware of the permission settings on their devices?

The next part of our survey data analysis reflects on the differences between the users' answers on the second question and the actual settings we found on their devices. Therefore, we now evaluate users' answers by comparing them with the users' actual interaction with their favorite apps. This comparison is temporal and adheres to the time we acquired the permission snapshot. Therefore, this is a snapshot that depicts the users' interaction with the system dialogues until that moment.

We are still studying the responses from group G_s in this section. However, due to inconsistencies in some of the users' responses we had to consider only those which did not cause any confusions. For example, one user suggested her favorite app was "messaging" and, at the same time, we found permission settings for more than one messaging applications on her device. Hence, it was not feasible to know the application she was referring to. These ambiguous answers were removed for this part of the study and therefore we report data derived from 47 responses from group G_s.

We use the Jaccard distance (ranges from 0.0 to 1.0) to measure the similarity of two binary vectors for each participant's answer. In general, the Jaccard distance of two vectors equals to 0.0 if the vectors are identical.

The first vector resembles the answers given for question 2 and the second resembles the actual privacy preferences/controls found in the participant's device for the specific app. For example, if the respondent answered that Twitter is her

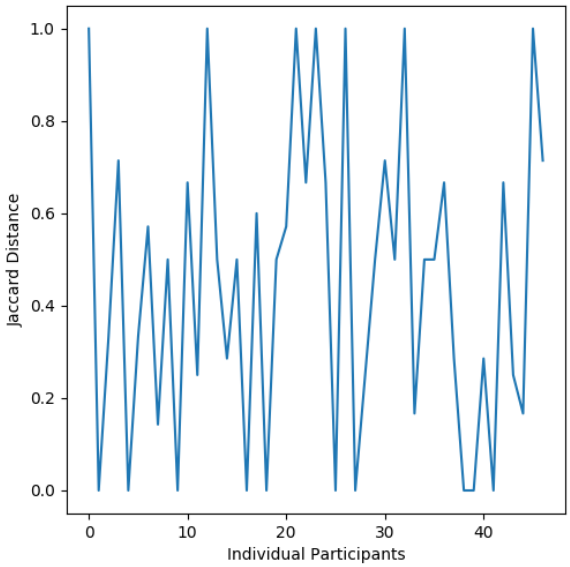


Fig. 4. Jaccard similarity of users' answers to Q2 and their actual permission settings.

most used/favorite app, we represent as $v_1 = (0, 1, 0, 1, 0, 0, 1, 0, 1)$ her answers to the second question, where 0 is "I would not allow access" and 1 is "I would allow access" to the following permission groups: (Calendar, Camera, Contacts, Location, Microphone, Phone, Sensors, SMS, Storage).

The second vector $v_2 = (N, 1, 0, 1, 0, 0, N, 0, 1)$ resembles the actual access settings found on her device for the specific app. Note that some apps do not declare permissions for specific groups; here for example, the Calendar is not used by the app, therefore this group is flagged as N in v_2.

In order to calculate the Jaccard distance we neglect users' choices made for the permission groups flagged as N. Hence, the vectors to be compared are now the following: $v_1 = (1, 0, 1, 0, 0, 0, 1)$ and $v_2 = (1, 0, 1, 0, 0, 0, 1)$. We do that because we do not want to compare users' answers (v_1) with actual settings (v_2) when the particular dangerous permission group is not declared by the app. Hence, we do not impute any missing values. Therefore, the Jaccard distance of v_1 and v_2 is 0.0 in this instance, which means that the user's answer and her actual settings on her device are exactly the same. This can be seen as an indication that the respondent was totally aware of the permission settings on her device related to the specific app.

Figure 4 shows the Jaccard distance between v_1 and v_2 for each participant's entry. In 10 cases the Jaccard distance between v_1 and v_2 was 0.0. This means that only 21.3% of the respondents appeared to have a clear view of the resources they allowed their favorite apps to access. This number is indeed lower if we consider that half of these participants appeared to be 100% permissive when their favorite app requests access to their devices' resources. The average Jaccard

distance derived from the 47 participants is approximately 0.71. The last metric shows that there exist misconceptions about the actual state of the permission settings in our participants' devices despite the fact that there were asked to provide their privacy preferences/settings for their favorite (hence most used) apps.

5.4 RQ4: Can we categorize users according to their privacy/permission settings?

As of October 2019, there existed N = 55 distinct sub-categories on the Google Play App store (e.g. Art & Design, Auto & Vehicles, Beauty, etc.). Our aim here is to create users' privacy profiles based on their acquired snapshots depicting the permission settings for each category.

Modeling User's Settings We accumulate permission settings on each device as follows:

App permission settings for each device are reconstructed from the permission snapshots and resembled by vectors $a = (p_1, p_2, \ldots, p_9)$, for $i \in [1, 9]$ (9 permission groups), where:

a) $p_i = 1$, i.e. the permission was allowed to this app,
b) $p_i = -1$, i.e. the permission was not allowed (or never requested by the app)
c) $p_i = 0$, i.e. the app did not declare this permission.

When there are cases where more than one apps from one category exist on a device, we perform the following basic calculations. For each permission group we count the "Allow" (i.e. "1") and the "Deny (i.e. "0") decisions and find the more prominent value between them. We transform this value to a float number (percentage) representing the probability of this user to allow or deny access to the resource protected by the permission from this specific dangerous permission group. When the prominent number refers to "Allow" decisions the float number is positive, and it is negative in the opposite case. If there exist apps that do not declare a specific permission from a group (e.g. Sensors), we fill this place with a zero. If there are equal "Allow" and "Deny" decisions for a permission group in a category, we assume that the user is positively inclined to allow access (according to our finding from RQ2).

Therefore, for each device we create a sequence (or a feature set in other words) of N = 55 vectors representing the tendency of the user to allow or deny an app from a given category.

In this sub-section we report results gathered from a larger group from our pool of participants (i.e. G_d). G_d consists of data derived from 67 devices. As explained earlier in this Section, we removed data derived from 4 devices. Additionally, we also noted that 7 participants did not provide app usage data. However, we included their permission snapshots in this part of the study because these data are not ambiguous, meaning that they could not be falsified

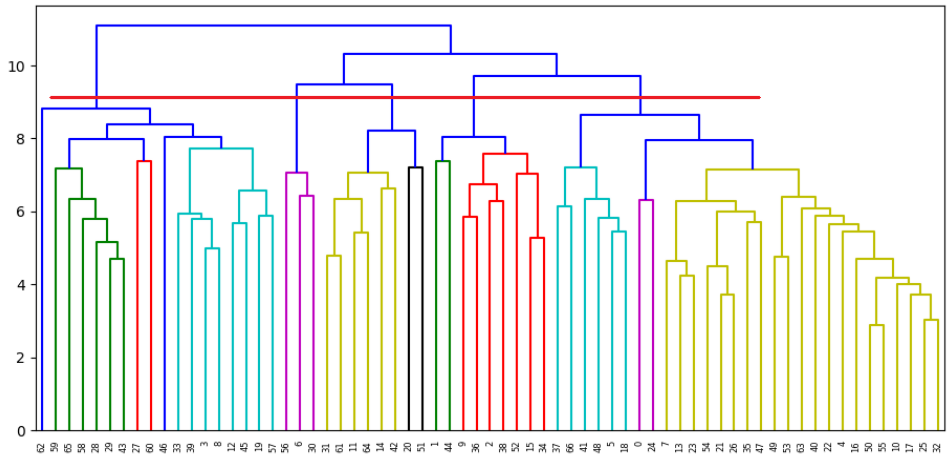


Fig. 5. Dendrogram derived from hierarchical clustering.

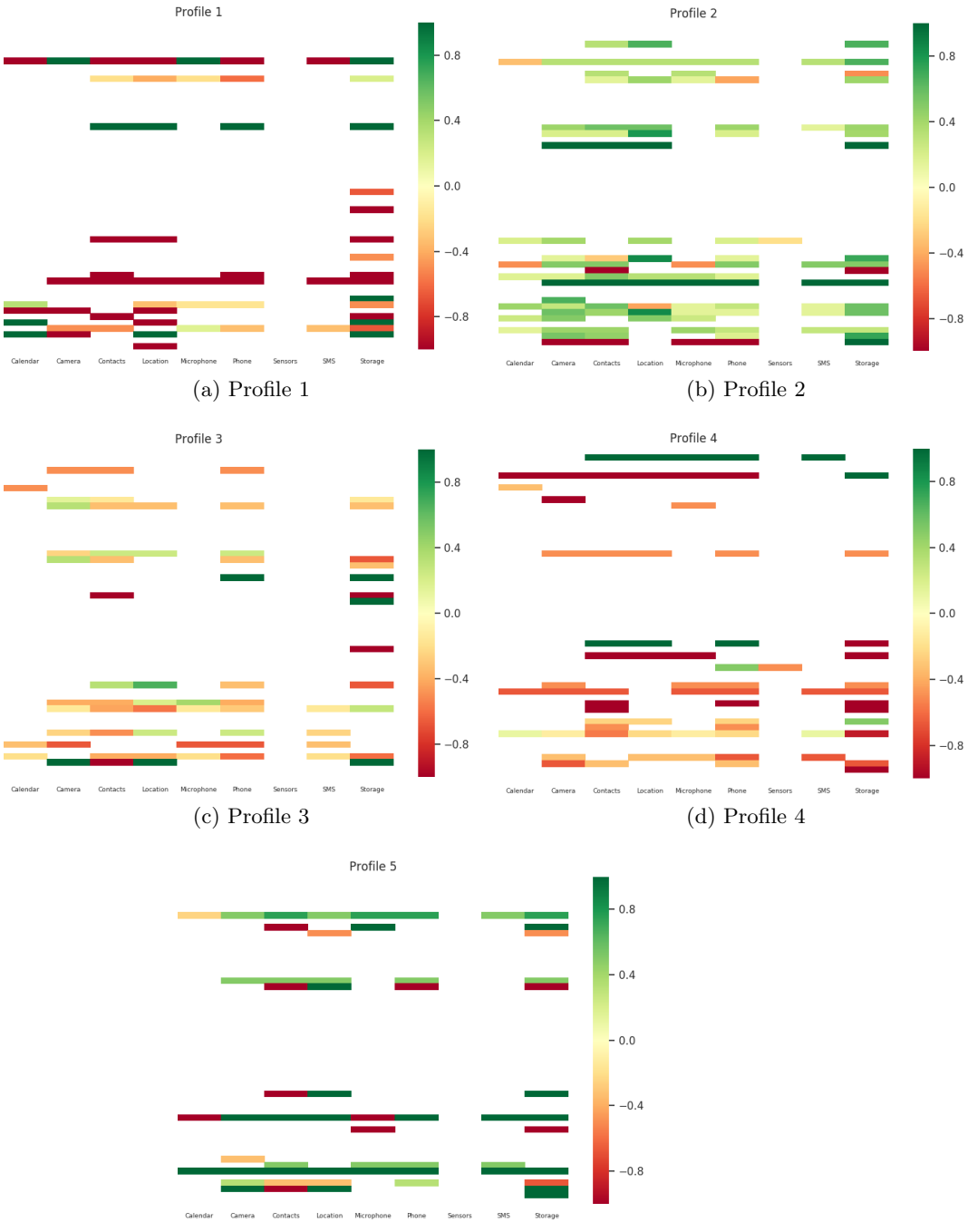
or somehow manipulated, given that they are provided by the Android system itself.

However, there is always a small probability that some permission settings in these devices (we refer here to the 7 participants) will describe apps that have never been used before. This is a reasonable concern which might lead to misinterpretations of a user's intention to allow or deny access to an app from a certain category. Indeed this was also a basic limitation of similar previous work [2].

In order to overcome this limitation we examined the data we derived from devices that provided app usage data. We measured the percentage of installed apps in each of these devices and identified from the app usage data if these apps were invoked at least once. We found that on average 94.22% of the installed apps were run at least once. Therefore, it is safe to generalize and assume that the majority of the data provided from the aforementioned 7 devices contain permission settings from apps that were used at least once.

Clustering Profiles We perform Agglomerative hierarchical clustering using *scikit-learn* [15] to identify clusters in our data (linkage method: *ward*). The same methodology was used by Liu et al. [14] recently to create similar privacy profiles. However, Liu et al. [14] did not consider users' permission settings for all known categories in the Play Store in their work.

We draw a dendrogram to visualize how clusters are formed from our data. After performing a visual inspection, we empirically decide to deviate the users in five big clusters (see the red line in Figure 5). Liu et al. [14] identified 7 clusters in their analysis. However, they admit that the majority of the users in their study is gathered in one big cluster.



(a) Profile 1

(b) Profile 2

(c) Profile 3

(d) Profile 4

(e) Profile 5

Fig. 6. Users' privacy preferences profiles derived from their permission settings.

Figure 6 showcases representative samples of the privacy profiles we created. On the vertical axes we place the different app categories and on the horizontal axes we show the nine groups of dangerous permissions. The color of each cell resembles the user's tendency to allow (green) or deny (red) an app from that category to access a resource from this permission group. White spaces denote the lack of knowledge of the user's reaction to access requests from apps from the given categories.

6 Discussion

As expected from the results presented so far, most of the users tend to protect a number of sensitive resources on their devices. Therefore, the majority of the profiles appear to be more restrictive. Profile 2 and Profile 5 are generally permissive. Profile 2 includes users who tend to allow apps access their Location and Storage. Profile 5 appears to be stricter with particular app categories compared to Profile 2. Profile 1 is restrictive in general, but allows access to Location and Sensors. Profile 3 would not usually allow access to the Calendar, Storage and the Microphone. Finally, Profile 4 appears to be generally restrictive.

Looking at the distribution of the population, we can report the following numbers. Profile 1 includes 18 users (26.9% of our sample), Profile 2 includes 8 users (11,9%), Profile 3 is the most populated with 29 users (43.3%), Profile 4 has 9 users (13.4%) and, finally, Profile 5 is the smallest comprising only 3 users (4.5%).

Compared to the work of Lin et al. [13] and Liu et al. [14] we identify similarities between our generally permissive users (Profile 2) and the "Profile 3 users" of [14] and the "unconcerned" users of [13]. Additionally, Profile 4 in our study is similar to the restrictive "Profile 4" that Liu et al. [14] identified as their protective users, and Lin et al. [13] as their "conservative" users. Finally, the derived clusters from our methodology seem to be more equally distributed compared to those presented in [14].

6.1 Android 10 Location Settings

The permission data collection methodology utilized in this study can be applied on the revamped finer-grained permission model for protection of location data in the most contemporary Android version (i.e. Android 10). The current version was released during Autumn 2019 and it features a new approach to location permission management, featuring two levels of protection. The user according to this updated model has the ability to choose between two location accessibility levels: a) Allow an app to access location data all the time (i.e. even when it is on the background), or b) allow access only when the app is in use (i.e. when it is in the foreground).

Compared to the previous models, the only difference in this occasion is the addition of an extra permission; the `ACCESS_BACKGROUND_LOCATION`. Thus, in order to update users' profiles in the near future to incorporate those users

who updated to the most recent OS version (Android 10), we probably need to introduce a 4th choice in the location permission group: "Always Allow", "Allow when in Use", "Deny", "Not requested". Additionally, we need to account for the fact that more consumers will start using devices running versions 9 and above (i.e. API level 28+). This means that we need to implement different profiles for these users which consist of 10 dangerous permission groups. For the moment these remarks form our plans for future work.

6.2 Pilot Study on Android 10 Location Settings

We attempted to measure users' acceptance of the modernized, tristate location permission system on Android 10, conducting a pen-and-paper exercise as follows. We gathered a random group of 25 undergraduate and postgraduate students (studying Cyber Security and Digital Forensics at the University of the West of England) and asked them to participate in a short experiment. We distributed a short questionnaire and asked them to anonymously answer three questions in 5 minutes.

The questionnaire comprised a screenshot of an app requesting location permission, adhering to the new tristate location permission system introduced in Android 10 [5], followed by 3 short questions. The depicted dialogue message stated: "Allow App 1 to access this device's location?". The message featured the following options: "Allow all the time", "Allow only while the app is in use", "Deny". The participants asked to answer the following questions:

- To comment on the functionality/outcome of each option.
- If the message was clear.
- Which option they would choose.

22 students answered anonymously the questionnaire. We briefly discuss the outcome in this section. 16 participants (i.e. 72.7%) said that the message shown by the system is clear. 3 students (i.e. 13.6%) claimed the opposite, and 3 other students said that "it is misleading", "not very clear", or "a little clear". Therefore, 72.7% thought the message is clear and 27.3% had a different opinion. The most interesting finding however derives from the answers to the last question. 16 students said they would choose the "Allow only while the app is in use" if they were using the app, and only 1 said they would "Allow all the time". Finally, 2 students said they would choose "Deny" and 3 students replied that "it depends on the app". Among these 3 participants, 2 of them said "it depends", but they would probably choose to "Allow only while the app is in use" or "Deny".

Hence, this preliminary study shows that if the users have the choice to allow an app to access the device's location only while the app is in the foreground, they are eventually positively inclined to provide the permission. Also, we saw that almost three quarters of the participants thought the message provided by the system about the tristate location permission was clear enough.

7 Conclusion

We utilized publicly available system information derived from the use of the `PackageManager`, accessible by any installed app on the device[3]. We showed that any app installed on an Android device is able to extract similar information and perform user profiling tasks related to the user's privacy awareness. In this study we gathered permission settings from 71 devices and identified 5 distinct user profiles, related to their inclination to allow or deny access to specific sensitive resources on their devices. We found that 13.4% of users in our sample belong to the most restrictive profile, 16.4% belong to generally permissive profiles and the rest of them are protective, allowing access to certain permission groups (Location, Sensors and Storage).

Moreover, our survey responses, and their comparison with participants' actual privacy controls, demonstrated that users do not feel comfortable with allowing apps to read their SMS, contact lists, and using their microphones. However, the results of this study demonstrated that, as users, we are keener to allow our favorite apps to access restricted resources.

Finally, following a cross-examination of the users' responses with their actual permission settings, we concluded that although users are supposed to have a better overview of the protected resources they allowed their favorite apps to access on their devices, they eventually fail to accurately report which groups are accessible and which are not. Also we identified the inclination of users to allow location access to an app only while the app is in the foreground (feature available on devices running Android 10).

As future work we intend to use our profile categorization methodology to investigate the feasibility of embedding these profiles in recommendation systems to efficiently suggest apps that match users' privacy settings. We believe that online app stores (such as the Google Play app store) have the capability to create more accurate privacy profiles using numerous permission snapshots via longitudinal measurements, because they have constant access to app usage statistics.

Acknowledgment

This work has been supported by the UWE Bristol Vice-Chancellor's Early Career Researcher Awards 2017-2018 and the Great Britain Sasakawa Foundation (No. 5303/2017).

References

1. Andriotis, P., Sasse, M.A., Stringhini, G.: Permissions snapshots: Assessing users' adaptation to the android runtime permission model. In: 2016 IEEE International Workshop on Information Forensics and Security (WIFS). pp. 1–6 (Dec 2016). https://doi.org/10.1109/WIFS.2016.7823922

[3] The dataset is available upon request. Please contact the corresponding author.

2. Andriotis, P., Li, S., Spyridopoulos, T., Stringhini, G.: A comparative study of android users' privacy preferences under the runtime permission model. In: Tryfonas, T. (ed.) Human Aspects of Information Security, Privacy and Trust. pp. 604–622. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-58460-7_42
3. Andriotis, P., Stringhini, G., Sasse, M.A.: Studying users' adaptation to android's run-time fine-grained access control system. Journal of Information Security and Applications **40**, 31–43 (2018). https://doi.org/10.1016/j.jisa.2018.02.004
4. Android Developers: Distribution dashboard. https://developer.android.com/about/dashboards (2019), accessed: 2019-10-13
5. AOSP: Tristate Location Permissions. https://source.android.com/devices/tech/config/tristate-perms (2020), accessed: 2020-01-31
6. Bonné, B., Peddinti, S.T., Bilogrevic, I., Taft, N.: Exploring decision making with android's runtime permission dialogs using in-context surveys. In: Thirteenth Symposium on Usable Privacy and Security ({SOUPS} 2017). pp. 195–210 (2017)
7. Diamantaris, M., Papadopoulos, E.P., Markatos, E.P., Ioannidis, S., Polakis, J.: Reaper: Real-time app analysis for augmenting the android permission system. In: Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy. pp. 37–48. ACM (2019). https://doi.org/10.1145/3292006.3300027
8. Felt, A.P., Egelman, S., Wagner, D.: I've got 99 problems, but vibration ain't one: a survey of smartphone users' concerns. In: Proceedings of the second ACM workshop on Security and privacy in smartphones and mobile devices. pp. 33–44. ACM (2012). https://doi.org/10.1145/2381934.2381943
9. Felt, A.P., Ha, E., Egelman, S., Haney, A., Chin, E., Wagner, D.: Android permissions: User attention, comprehension, and behavior. In: Proceedings of the Eighth Symposium on Usable Privacy and Security. pp. 3:1–3:14. SOUPS '12, ACM, New York, NY, USA (2012). https://doi.org/10.1145/2335356.2335360
10. Hossen, M.Z., Mannan, M.: On understanding permission usage contextuality in android apps. In: IFIP Annual Conference on Data and Applications Security and Privacy. pp. 232–242. Springer (2018). https://doi.org/10.1007/978-3-319-95729-6_15
11. Iqbal, M.S., Zulkernine, M.: Droid mood swing (dms): Automatic security modes based on contexts. In: International Conference on Information Security. pp. 329–347. Springer (2017). https://doi.org/10.1007/978-3-319-69659-1_18
12. Likert, R.: A technique for the measurement of attitudes. Archives of psychology (1932)
13. Lin, J., Liu, B., Sadeh, N., Hong, J.I.: Modeling users mobile app privacy preferences: Restoring usability in a sea of permission settings. In: 10th Symposium On Usable Privacy and Security ({SOUPS} 2014). pp. 199–212 (2014)
14. Liu, B., Andersen, M.S., Schaub, F., Almuhimedi, H., Zhang, S.A., Sadeh, N., Agarwal, Y., Acquisti, A.: Follow my recommendations: A personalized privacy assistant for mobile app permissions. In: Symposium on Usable Privacy and Security (2016)
15. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
16. Raval, N., Razeen, A., Machanavajjhala, A., Cox, L.P., Warfield, A.: Permissions plugins as android apps. In: Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services. pp. 180–192. ACM (2019). https://doi.org/10.1145/3307334.3326095

17. Reardon, J., Feal, Á., Wijesekera, P., Elazari Bar On, A., Vallina-Rodriguez, N., Egelman, S.: 50 ways to leak your data: An exploration of apps' circumvention of the android permissions systems. In: 28th USENIX Security Symposium (2019)
18. Reinfelder, L., Schankin, A., Russ, S., Benenson, Z.: An inquiry into perception and usage of smartphone permission models. In: International Conference on Trust and Privacy in Digital Business. pp. 9–22. Springer (2018). https://doi.org/10.1007/978-3-319-98385-1_2
19. Scoccia, G.L., Ruberto, S., Malavolta, I., Autili, M., Inverardi, P.: An investigation into android run-time permissions from the end users' perspective. In: Proceedings of the 5th International Conference on Mobile Software Engineering and Systems. pp. 45–55. ACM (2018). https://doi.org/10.1145/3197231.3197236
20. Thompson, C., Johnson, M., Egelman, S., Wagner, D., King, J.: When it's better to ask forgiveness than get permission: attribution mechanisms for smartphone resources. In: Proceedings of the Ninth Symposium on Usable Privacy and Security. p. 1. ACM (2013). https://doi.org/10.1145/2501604.2501605
21. Votipka, D., Rabin, S.M., Micinski, K., Gilray, T., Mazurek, M.L., Foster, J.S.: User comfort with android background resource accesses in different contexts. In: Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018). pp. 235–250 (2018)
22. Wijesekera, P., Baokar, A., Tsai, L., Reardon, J., Egelman, S., Wagner, D., Beznosov, K.: The feasibility of dynamically granted permissions: Aligning mobile privacy with user preferences. In: 2017 IEEE Symposium on Security and Privacy (SP). pp. 1077–1093 (May 2017). https://doi.org/10.1109/SP.2017.51
23. Wijesekera, P., Reardon, J., Reyes, I., Tsai, L., Chen, J.W., Good, N., Wagner, D., Beznosov, K., Egelman, S.: Contextualizing privacy decisions for better prediction (and protection). In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems. p. 268. ACM (2018). https://doi.org/10.1145/3173574.3173842