Profiling Mobile Users Privacy Preferences

Aziz Alshehri
Computer Science and Information Technology College
Umm Al-Qura University
Makkah, Saudi Arabia

Fayez Alotaibi
Computer Science and Information Technology College
Shaqra University
Riyadh, Saudi Arabia

Abstract

The evolution of information technology encourages people to use a wide range of different applications for a different purpose. These applications collect large amounts of personal information about their users. Therefore, many users are concerned about how to manage and control the large volumes of personal information in order to protect their privacy. Most mobile privacy researches assume that users have a uniform privacy concerns. Whilst, the current research indicates that users have different privacy attitudes and expectations. Therefore, in order to cater to different user preferences, this paper presents a survey study aimed to cluster the entire user population into a number of subgroups that have similar preferences within the subgroups. Applying privacy profiles as default settings for initial interfaces could significantly reduce the burden and frustration of the user. The result shows that it is possible to cluster users into subgroups and the optimal number of clusters based on the k-means method is four clusters.

1. Introduction

With the rapid growth of the devices, activities, services and information, the enormous amount of private and personal information that is stored has also increased. Therefore, users are becoming increasingly concerned about their personal information, how it is used, by whom and where it is stored [1]. For instance, a Consumer Report found that 92% of British and U.S.

Internet users are concerned about their privacy online [2]. When users became aware of online privacy issues, they were asked what made them most worried about their online privacy: 45% of British Internet users stated that it is personal information being shared between companies [3]. In addition, 89% of users avoided these companies because they believed that companies do not protect their privacy. It was also found that 76% of Internet users limited their online activity in the last 12 months due to these concerns [4]. This evidence indicates that users are sufficiently worried about their online privacy.

Due to the concerns of the users about privacy protection, most mobile operating systems such as Android and iOS provide some privacy safeguards for users [4]. Despite these provisions, there are several usability issues related to the functionality and interface. For instance, Kelley et al. found that users struggle to understand the permissions in Android due to the lack of usability [4]. Therefore, the Federal Trade Commission suggested privacy controls need more improvement to protect users' privacy [3].

A noticed focus has been given to the development of policies, procedures and tools that aid an end-user in managing and understanding their privacy-related information. However, these approaches assume that users can correctly configure all resulting settings and they have uniform privacy requirements. In reality, users do have different privacy concerns and requirements as they have heterogeneous privacy attitudes and expectations [5]. For example, some users consider personal information such as age, address and gender in their profile on a social network being more sensitive than others [6]. Furthermore, in practice it is unrealistic to assume homogeneous privacy requirements across a whole population [7].

Accordingly, there is a need for an approach that considers individual requirements in a centralised and usable manner to meet users' needs. This paper shows that users are diverse, and it is possible to divide users into small groups according to the users' privacy preferences. Applying privacy profiles as default settings for initial interfaces could significantly reduce the burden and frustration of the user. The result shows the optimal number of clusters based on the k-means method is four clusters.

This paper is organised into five sections. Section 2 presents an analysis of background literature. Section 3 shows how the data collected. Section 4 described different methods for determining the optimal number of clusters. The conclusions and future work are presented in Section 5.

2. Background Literature

This section provides an overview of current privacy solutions. In recent years, many researches have been published on privacy in many areas such as mobile applications, web application, and social networks in order to protect users' privacy because privacy exists wherever personal information or sensitive information is disclosed. However, this section merely focuses on a mobile platform due to related to the proposed system.

3. Information Flow Analysers

Many tools have been developed in recent years that aim to analyse and to detect personal information on mobile platforms. These tools analyse mobile apps regarding potential privacy breaches before they leave the system via untrusted apps. Some of the more prominent examples are, Taintdroid [8], AppIntent [9], Little BrothersWatching You [10], and PiOS [11].

Taintdroid was designed to detect sensitive data when it leaves the system via untrusted applications [8]. It was designed based on a dynamic approach which is executed whilst a program is in operation. The system can track the flow of data through four levels: variable, method, message, and file. Although TaintDroid detects the sensitive data, the system assumes that users can correctly configure all the resulting settings. Therefore, this approach could impose an undue burden on the users. In addition, they do not examine the usability related to the interface displayed to users.

In comparison, Balebako et al [10] presented a solution that focuses on the user's awareness of privacy issues. The solution improves the user's understanding of potential privacy leakages. It is built based upon the TaintDroid platform and helps users to know the frequency and destination of data being shared by an application. It also provides many user interfaces which can help to inform users about which privacy sensitive information leaves the phone. However, they do not provide users with control over their personal information to allow them to specify which type of information they prefer not to leave the mobile.

Another tool that aims to analyse programs for possible leaks of sensitive information from a mobile device to the third party is PiOS. It detected privacy leaks related to device ID, location and phone number. Moreover, PiOS considered the address book, browser history, and photos. PiOS uses static analysis to detect data flows. They have analysed more than 1,400 iPhone apps and they found that a majority of apps leak the device ID, which can provide detailed information about the habits of a user. However, PSiOS does not provide users with fine grain control over their personal information.

3.1. Finer Grain Privacy Controls

A number of research prototypes have also offered used fine grain controls in order to prevent potential privacy leakages. For example, AppFence [12], TISSA [13], AntMonitor [14] and ProtectMyPrivacy [5]. There are several techniques were used to enhance information flow control for mobile.

AppFence uses replacing information approach in order to protect sensitive data [12]. AppFence provides users with two privacy controls to protect sensitive resources: shadowing and blocking. Sometimes users do not want to provide application access to sensitive data. Therefore, AppFence sends shadow data instead of the actual data. For example, when an application requires access to user's contacts, AppFence may provide application shadow data that contains no contact entries, contains only those genuine entries not considered sensitive by the user, or that contains shadow entries that are entirely fictional. The second approach for protecting sensitive data is blocking sensitive data from being exfiltrated off the device. AppFence uses TaintDroid information flow tracking to track the sensitive data and prevent information from transmissions of these data out of the device. However, the system does not alert users about how applications use data and whether they will exhibit side effects if privacy controls are applied. In order to know whether side effects impact user-desired functionality, it needs to consult users each time. In this case, the system may place a high level of burden on users.

The Taming Information Stealing Smartphone Applications (TISSA) provides users with fine-grained control over the disclosure of their personal information and consists of three main components [13]. TISSA was designed to protect four types of personal information: phone identity, location, contacts, and call log. The first one is the privacy setting content provider. It contains the current privacy settings for untrusted apps on the mobile device. It also provides users with an interface in order to query the current privacy settings for an untrusted app (e.g., a location manager). In order to protect personal information, TISSA provides users with empty or bogus options for personal information that may be requested by the app. The second component is the privacy-setting manager. It allows users to manage or update the privacy settings for installed apps. The third component contains content providers or services to regulate the access for four types of personal information: phone identity, location, contacts, and call log. For example, when an app requires access to private data, the system will query the privacy settings, and response to the requests according to the current privacy settings for the app. However, it is difficult for an average user to determine which type of permission is high or low

risk for the app because he does not know the reason about permission requirements for individual apps. Additionally, the system does not assist the user to make the right choice in order to reduce the burden on mobile users.

PrivacyGuard [7] and AntMonitor [14] provide fine-grained privacy control and provide ground truth mapping of packets to applications. They used an approach which analyses actual network traffic of Android using VPNService API to intercept traffic. This approach does not require root permissions and is portable to all devices with Android version 4.0 or later. The AntMonitor system consists of three components: an Android application, AnyClient, and two server applications, AntServer and LogServ. Whilst PrivacyGuard runs in its entirety on the local device. The purpose of the client-side analysis is to protect users in real time and provide fine-grained privacy control. However, LogServer works as the central repository to store and analyse all network traffic data and does not have to analyse a large amount of live traffic compared to AntServer. To evaluate AntMonitor system, they recruited student volunteers to use AntClient on their phones. The system collects the packets of the applications that the volunteers selected and stores them at LogServer in order to check whether any of the installed applications are sending the personal data out to the Internet. They found that 44% and 66% of the users have applications that leak their International Mobile Equipment Identity (IMEI) and Android Device ID respectively. However, both PrivacyGuard and AntMonitor assume that average users can correctly specify their personal information to allow the system to detect them when they leave their mobiles. In this case, these solutions do not help users to overcome the burden associated with managing such a large number of data.

ProtectMyPrivacy (PMP) provides users with fine-grained privacy for each app in order to send the anonymized data instead of privacy sensitive information [5]. It detects privacy leaks on iOS Applications. The type of data that PMP protects is a unique device identifier, IMEI, Wi-Fi MAC address and Bluetooth MAC address. Another private data type that PMP protects is the user's address book. It includes names, addresses, phone numbers and emails because some apps upload this information to a server without user's permission. When the app wants to access to the private data, PMP allows the user to deny or allow the app to access private data in real time. Hence, PMP provides user two options to protect his address book: user can allow the app to access his address book or allow PMP to sends an alternative address book, filled with fictitious entries (names, emails and phone numbers). Additionally, they have developed a crowdsourcing system to help users to make informed decisions, which provides app specific privacy recommendations. However, the system just deals with mere access to private data but does not address privacy once the data leaves the app. Moreover, the system does not provide each user with personalized recommendations. Each user has its own privacy preferences. Therefore, it would be helpful to take account of user's profile when the system generates recommendations, in order to make a more personal recommendation.

3.2. Privacy Profiles

A few studies have proposed modelling and predicting users' privacy preferences [15] [16]. Frank et al. cluster 188,389 Android apps and 27,029 Facebook apps to find patterns in permission requests [15]. They used a probabilistic method to extract permission request patterns from Android and Facebook apps. They identified over 30 common patterns of permission requests. However, they looked for permission request patterns in Android apps but they do not identify patterns in user privacy preferences.

Liu et al analysed the permissions users granted to mobile apps on Android and realized that the permission model is too complex and they can find patterns in permission requests [16]. They used machine learning clustering algorithms to split users into a small number of profiles based on their decisions to grant or deny apps access to different permissions. Their result showed that it is possible to significantly reduce user burden while allowing users to better control their mobile app permissions. However, they do not elicit user's privacy preferences in a context where they are not just about the permissions requested by an app but also about the type of information, app categories, data location, time of access data, the entity access to data, data usage and the level of data.

4. Data Collection

Participants were recruited through different platforms such as Email, social network, and some communities' centre. Prior to displaying the survey questions, its aims and structure were briefed confirming that the respondents should be 18 years or older and they are free to withdraw up until the final submission of their responses. In total, 407 completed responses and the total responses are within the range of other surveys in the research domain and close to the expected and targeted figure.

Demographic information was collected including questions related to gender, age, education, and the level of knowledge in order to analyse the data, though the age ratio or any other demographic composition of the participants were not specifically controlled. Among these participants, 30% of them were male; 70 of them were female. Regarding the

age, almost half of the participants were between 25 and 34 which represent 47% of participants. The second large age group was between 35-44 which represent 35%. The vast majority of the population of the participants were aged between 24-44 years old.

5. Cluster Analysis:

As mentioned in the previous section, users' preferences are diverse. Therefore, this section seeks to cluster the entire user population into a number of subgroups that have similar preferences within the subgroups, which can be used to later leverage to create the initial interfaces.

Before clustering the entire user population into a number of subgroups, user's preferences were encoded into a vector and trim the dataset to prevent overfitting. After these pre-processing steps, we obtained a matrix of 46 columns and 407 rows, where each row of the matrix represented a participant. This step yielded a total number of 407 unique participants with 18,722 responses. Each entry of the matrix was a value between [1, 5].

The K-means clustering algorithm is utilized to cluster users' preferences into subgroups. This clustering algorithm is the most popular data mining technique to partition observations into K clusters. Each user is assigned to the cluster with the nearest mean, which itself serves as a representative value of the cluster. K clusters are selected as initial centres; then distances of all data elements are calculated by Euclidean distance formula. Data elements having less distance to centroids are moved to the appropriate cluster. The process is continued until no more changes occur in clusters [k-1]. The K-means clustering algorithm works fast with the Large data set since the time complexity is O(nki) where n is the number of patterns, k is the number of clusters and i is the number of the iterations. The K-means algorithm used Euclidian distance which makes it works well with numeric values with interesting statistic meaning.

Determining the optimal number of clusters in a data set is a fundamental issue in unsupervised data clustering like K-means. Unfortunately, there is no definitive answer to determine the optimal number of clusters. The optimal number of clusters is somehow subjective and relies on the technique used for measuring similarities. However, different approaches for estimating the optimal number of clusters have been proposed. For example, statistical testing methods, visual exploration and Precision of predicting users' preferences.

5.1. Gap Statistic

Gap statistic was perfumed to determine the optimal number of clusters. The gap statistic compares the total within intra cluster variation for different values of k. This statistic avoids the increase or decrease in the monotony of other validation scores with increasing number of clusters

First, the intra-cluster distance is averaged over the k clusters:

$$W\mathrm{k} = \sum_{r=1}^{k} \frac{1}{2nr} \sum_{i,j} D(i,j)$$

Where nr denotes the number of elements of the cluster r.

The Gap statistic is defined as:

$$\textbf{Gap(k)} = \mathrm{E}(\log((W_k)) - \log(W_k)$$

Where E(log(Wk)) is the expected logarithm of the average intra-cluster distance. The optimum number of clusters is the smallest value k. It is clear from the table1 that the smallest sum of errors is the fourth cluster.

Table 1: Clustering Errors

No. of Clusters (K)	Sum of Errors
1	0.0131
2	0.0102
3	0.0096
4	0.0095
5	0.0105
6	0.0103
7	0.0098

5.2. Interpretability and understandability

This approach relies on how the cluster is easy to interpret and meaningful. The clusters that are easy to interpret would also make it easier for users to identify which cluster best matches their preferences. Additionally, visualisation each profile would also make it easier to interpret the cluster. Hence, it helps to determine the optimal number of clusters. To demonstrate the meaning of each cluster, the centroid of each cluster was computed by averaging the feature vectors of instances within the cluster.

Figure 1 shows that when the users' preferences were divided into more than four clusters, it will be more difficult to interpret each cluster. For example, when there are six clusters, cluster 2, 3 and 4 are very similar (close) to each other. The centroid for cluster 2, 3 and 4 are 2.3, 2.63 and 2.71. In addition, both cluster 5 and 6 are almost close to each other which represents unconcerned users. When there are 7 clusters, the centroid for 2, 3, and 4 are also similar

to each other. However, as the number of clusters increases, the similarity between the clusters gradually increases too. Therefore, when the number of clusters is four, it is easy to interpret and meaningful. Lin also found four clusters but they used hierarchical clustering with Canberra distance [17]. They named each cluster based on their characteristics: conservatives, unconcerned, fence-sitters, and advanced users. These names could be used to label each cluster.

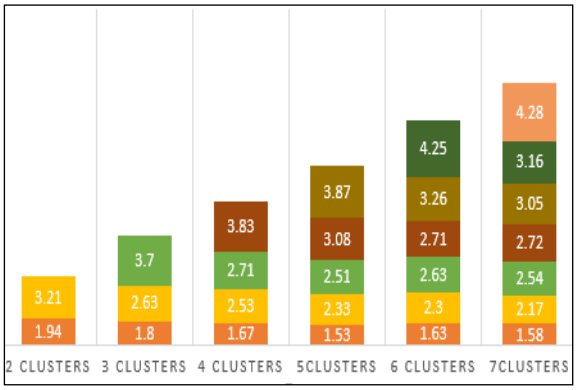

Figure 1: Average of each cluster

Multi classifications were performed to determine the optimal number of clusters. The result presents the accuracy of determining each user to the cluster. In other words, when the classifier has a high prediction of determining each user with their cluster and has high accuracy, this indicates that the number of clusters is good to utilize. Figure 3 shows that both the number of cluster three and four are high accuracy (0.84 and 0.82) and high prediction. On the contrary, when the number of clusters is more than 4, the correct prediction will decrease.

Table 2: The accuracy of predicting users' preferences

No. of Clusters (K)	Accuracy	Lowest Prediction Cluster
3	0.84	Cluster 2=77.8%
4	0.82	Cluster 3=70%
5	0.80	Cluster 1= 68%
6	0.75	Cluster 6=40%

Different methods were performed to determine the optimal number of clusters in a data set. These methods include precision of predicting users' preferences, the interpretability and understandability and the gap statistic methods. The result from the three methods shows that four clusters are the optimal number of clusters.

6. Conclusion and Future Work

This paper showed that it is possible to divide users into different profiles. Choosing a good privacy profile reduces the user configuration effort. The k-means method was utilised to determine the number of profiles. However, in order to determine the optimal number of clusters in the dataset three methods were used statistical testing methods, visual exploration, and precision of predicting users' preferences. The result shows that the optimal number of clusters is four clusters. Classifier has a high prediction of determining each user with their cluster and has high accuracy which represents 82%.

It is evident however that users do have different privacy preferences and needs because users have heterogeneous privacy attitudes. Further research requires to develop a technique on how to assign a new user to the right cluster. Moreover, how to build adaptive interfaces from the privacy preference of the users that seeks to optimise usability and convenience yet improves awareness.

7. Reference

[1] A. I. Anton, J. B. Earp, and J. D. Young, "How Internet Users' Privacy Concerns Have Evolved," IEEE Priv. Secur., vol. 1936, no. February, pp. 21–27, 2010.

[2] TRUSTe, "2016 TRUSTe/NCSA Consumer Privacy Infographic - US Edition | TRUSTe," 2016. [Online]. Available: https://www.truste.com/ resources/privacy-research/ncsa-consumer-privacy-index-us/. [Accessed: 11-Mar-2017].

[3] Federal Trade Commission, "Mobile privacy disclosures - Building trust through transparency," no. February, p. 29, 2013.

[4] P. G. Kelley, S. Consolvo, L. F. Cranor, J. Jung, N. Sadeh, and D. Wetherall, "A conundrum of permissions: Installing applications on an android smartphone," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 7398 LNCS, pp. 68–79, 2012.

[5] Y. Agarwal and M. Hall, "ProtectMyPrivacy : Detecting and Mitigating Privacy Leaks on iOS Devices Using Crowdsourcing Categories and Subject Descriptors," Proceeding 11th Annu. Int. Conf. Mob. Syst. Appl. Serv., vol. 6, no. September, pp. 97–110, 2012.

[6] Y. Song and U. Hengartner, "PrivacyGuard: A VPN-based Platform to Detect Information Leakage on Android Devices," Proc. 5th Annu. ACM CCS

Work. Secur. Priv. Smartphones Mob. Devices - SPSM '15, pp. 15–26, 2015.

[7] Y. Song, "PrivacyGuard : A VPN-Based Approach to Detect Privacy Leakages on Android Devices by," pp. 15–26, 2015.

[8] W. Enck et al., "TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones," ACM Trans. Comput. Syst., vol. 32, no. 2, p. 5, 2014.

[9] Z. Yang, M. Yang, Y. Zhang, G. Gu, P. Ning, and X. S. Wang, "AppIntent: analyzing sensitive data transmission in android for privacy leakage detection," Proc. 2013 ACM SIGSAC Conf. Comput. Commun. Secur. - CCS '13, pp. 1043–1054, 2013.

[10] R. Balebako, J. Jung, W. Lu, L. F. Cranor, and C. Nguyen, "'Little Brothers Watching You': Raising Awareness of Data Leaks on Smartphones," SOUPS '13 Proc. Ninth Symp. Usable Priv. Secur., p. 12:1--12:11, 2013.

[11] M. Egele, C. Kruegel, E. Kirda, and G. Vigna, "PiOS Detecting privacy leaks in iOS applications," Proc. 18th Annu. Netw. Distrib. Syst. Secur. Symp. NDSS 2011, p. 11, 2011.

[12] P. Hornyack, S. Han, J. Jung, S. Schechter, and D. Wetherall, "These Aren't the Droids You're Looking for: Retrofitting Android to Protect Data from Imperious Applications," Ccs, pp. 639–652, 2011.

[13] Y. Zhou, X. Zhang, X. Jiang, and V. W. Freeh, "Taming Information-Stealing Smartphone Applications (on Android)," 4th Int. Conf. Trust Trust. Comput., no. 2011, pp. 93–107, 2011.

[14] A. Le, U. C. Irvine, S. Langhoff, and A. Shuba, "AntMonitor : A System for Monitoring from Mobile Devices," no. 1, pp. 15–20, 2015.

[15] M. Frank, B. Dong, A. P. Felt, and D. Song, "Mining permission request patterns from Android and Facebook applications," Proc. - IEEE Int. Conf. Data Mining, ICDM, pp. 870–875, 2012.

[16] B. Liu, J. Lin, and N. Sadeh, "Reconciling Mobile App Privacy and Usability on Smartphones: Could User Privacy Profiles Help?" 2013.

[17] J. Lin, B. Liu, N. Sadeh, and J. I. Hong, "Modeling Users' Mobile App Privacy Preferences: Restoring Usability in a Sea of Permission Settings," 12th USENIX Secur. Symp., pp. 199–212, 2014.

