Privacy by Design Permission System for Mobile Applications

Karina Sokolova*†, Marc Lemercier*

*University of Technology of Troyes
Troyes, France
{karina.sokolova, marc.lemercier}@utt.fr

Jean-Baptiste Boisseau†

†EUTECH SSII
La Chapelle Sain Luc, France
{k.sokolova, jb.boisseau}@eutech-ssii.com

Abstract—**The Privacy by Design concept proposes to integrate the respect of user privacy into systems managing user data from the design stage. This concept has increased in popularity and the European Union (EU) is enforcing it with a Data Protection Directive. Mobile applications have emerged onto the market and the current law and future directive is applicable to all mobile applications designed for EU users. By now it has been shown that mobile applications do not suit the Privacy by Design concept and lack for transparency, consent and security. The actual permission systems is judged as unclear for users. In this paper, we introduce a novel permission model suitable for mobile application that respects Privacy by Design. We show that such adapted permission system can improve the transparency and consent but also the security of mobile applications. Finally, we propose an example of the use of our system on mobile application.**

***Keywords*–permission, permission system, mobile, privacy by design, privacy, transparency, control, Android, iOS, application, development, software design, pattern, mobility, design, modelling, trust**

I. Introduction

Mobile devices gain in popularity. Thousands of services and applications are proposed on mobile markets and downloaded every day. Smart devices have a high data flow processing and storing large amounts of data including private and sensitive data. Most applications propose personalized services but simultaneously collect user data even without the user's awareness or consent. More and more users feel concerned about their privacy and care about services they use. The TRUSTe survey conducted in February 2011 shows that smartphone users are concerned about privacy even more than about the security (the second in the survey results) [1].

Nowadays, people realize the lack of privacy especially while using new technological devices where information is massively collected, used and stored (Big Data notion). The privacy regulation aiming to control personal data use is set up in many countries. European Union privacy regulation includes the European Data Protection Directive (Directive 95/46/EC) and the ePrivacy Directive. United States regulation includes Children's Online Privacy Protection Act (COPPA) and The California Online Privacy Protection Act of 2003 (OPPA). Canada is under the Personal Information Protection and Electronic Documents Act (PIPEDA) concerning privacy.

The Privacy by Design (PbD) notion proposes to integrate privacy from the system design stage [2] to build privacy-respecting systems. PbD proves systems can embed privacy without sacrificing either security or functionality. Some PbD concepts are already included in European data legislation; the notion is considered to be enforced in European Data Protection Regulation, therefore systems made with PbD are compliant with the law. An application made with PbD notion is not only a benefit for users, along with the opportunity to provide a truly personalized service, but also a legal obligation for developers.

The PbD concept was firstly presented by Dr. Ann Cavoukian. She proposes seven key principles of PbD enabling the development of privacy-respective systems. The system should be proactive, not reactive, embed privacy feature from the design, integrate Privacy by Default, respect user privacy, data usage should be transparent to the end user and the user should have access to the mechanism of control of his data. Full functionality and the end-to-end security should be preserved without any sacrifice. [2]

Mobile privacy was discussed in 'Opinion 02/2013 on apps on smart devices' by the Article 29 Data Protection Working Party [3], where the opinion on mobile privacy and some general recommendations were given. The article states that both the Data Protection Directive and the ePrivacy Directive are applicable to mobile systems and to all applications made for EU users. Data Protection Regulation is also applicable to mobile systems. The article defines four main problems of mobile privacy: lack of transparency, lack of consent, poor security and disregard for purpose limitation.

Many reports propose recommendations about mobile privacy improvement repeating basic privacy notions (e.g., data minimization, clear notices) but the exact patterns or a technical solutions are missing [4].

The permission system is embedded in the mobile systems and is a crucial part of mobile security and privacy. Nowadays, permission systems do not follow Privacy by Design notions. Many works are concentrated on analysing and modelling the actual permission systems [5][6][7], on improving actual permission systems to give more control to the user [8][9][10] or to add additional transfer permissions [11], on visual representation of permissions [12], on user perceptions of current permission systems [13][14][15], on the data flow analyses (possible data leakage detection) and the actual permissions enforcement and verifications [16][17][18][19][20][21][22][23]. To our knowledge no work has been conducted on redefining the permission system to fit the Privacy by Design notion.

The remainder of the paper is organized as follows: Section 2 describes current permission systems of iOS and Android and points out problems regarding Privacy by Design. Section 3 introduces our proposal: the pattern of the privacy-respecting permission system. We show that it can cope with the transparency, consent and purpose disregard problems and also improve the security. Section 4 shows the application of our novel permission system to the real mobile application. The paper ends with a conclusion and future works.

II. Existing Mobile Permission Systems

In this section, we present current iOS and Android permission systems and evaluate those systems regarding the PbD notion. We take into account the full functionality allowed by the permission system, privacy by default, transparency and the control notions.

- Full functionality: possibility to use all functionalities available on the platform.
- Privacy by Default: the default configurations of the system are privacy protective.
- Transparency: user should clearly understand what data is used, how and for what purpose.
- Control: user should have a full control over his personal data usage.

We consider the privacy policy to be very important for the proactive and transparent system therefore we present the state of application privacy policy in both systems.

A. Permissions

iOS and Android have different strategies concerning the access to the device data. The iOS platform gives to non-native applications access only to the functionalities listed in privacy settings: location services, contacts, calendar, reminder, photos, microphone and Bluetooth (sensitive data, such as SMS and e-mails are not shared at all). Recently, the connection to Facebook, Twitter, Flickr and Vimeo was added to the platform (iOS7). Full functionality is given up for privacy reasons as applications cannot use the full power of the platform but only a limited number of functionalities.

An iOS application should have permission to access information listed above. By default, the installed application has no permission granted. The application displays a pop-up explaining what sensitive data it needs before to access it. The user can accept or decline permission. If permission is declined, the corresponding action is not executed. If the permission is accepted, the application obtains access to the corresponding data. The user is asked to grant permission only once, but he/she can activate or deactivate such permission for each application in privacy settings integrated by default into the iOS. Thereby iOS maintains transparency, control and privacy by default.

The Android system remains on the sharing principle. Full functionality is preserved: applications have access to all native applications' data and can expose the data themselves. Applications need permission to access the data, but differently from iOS, users should accept the full list of permissions before installing an application. While all permissions are granted, an application has full access to the related data. Some Android permissions tagged as 'dangerous' can be prompted to the user every time the data is going to be accessed, but it is rarely the case. Users see the list of dangerous permissions on the screen before installing the application.

TABLE I
Android and iOS Permission Systems Comparison

	Full Functionality	Default Settings	Transparency	Control
Android	+	-	-	-
iOS	-	+	-/+	+

Android proposes more than 100 default permissions and developers can add supplementary permissions. Multiple works show that users do not understand many of default permissions and fail to judge the application privacy and security correctly using the full permission list [13][15]. Permissions do not clearly show what data is used for and how. Moreover, some other studies show the abusive usage of android permissions by developers [24].

Some users do not check the Android permission list because they need a service and they know that all permissions should be accepted to obtain it. Android permission list looks like a license agreement on a desktop application that everybody accepts but very few actually read [25]. An Android user does not have any control over permissions once the application is installed: permissions cannot be revoked. Android does not include an iOS-like system permission manager (privacy settings) by default, therefore the user has to activate or deactivate the entire functionality to disable the access to related data (e.g., Wi-Fi or 3G for Internet connection; GPS for geolocation) or to use additional privacy enhancing applications.

Both iOS and Android default permission systems mostly inform about data access, but not about any other action that can be completed with the data. For example, no permission is needed to transfer the data. Android and iOS include permissions for functionalities that can be related to personal data transfer, such as Bluetooth and Internet. Permissions can be harmless to users, but there is no indication of whether personal data is involved in a transaction. This decreases the transparency of both platforms.

Android and iOS permissions do not include purpose explanation. An iOS application helps to understand the purpose by asking permission while in use, but if an application has a granted permission once for one functionality it could use it again for a different purpose without informing the user. Android users can only guess what permission is used for and whether the use is legitimate.

Table I shows the system differences regarding four main privacy notions: full functionality, transparency, control and privacy by default. One can see that the current Android permission system is missing in transparency, control and default privacy; iOS sacrifices the functionality and also lack of transparency. Permissions are often functionality-related and users fail to understand and to judge them. Personal data usage is unclear and the purpose is missing.

B. Privacy Policy

Users should choose applications they can trust. Apple ensures that applications available on the market are potentially harmless, although Android users should judge the application for themselves with the help of information available on the market. The AppStore and Google Play provide similar information: name, description, screenshots, rating and user reviews.

The transparency and the proactivity of the system can be improved by including the privacy policy in the store. A user can be informed about the information collected and stored before he downloads the application. Without any privacy policy, the user can hardly evaluate the security and privacy of the application, only the functionality and stability of the system. In their feedback, users often evaluate the functionalities and user interfaces and report bugs, but they rarely indicate privacy and security problems.

iOS does not require developers to include the privacy policy in the application but only in applications directed at children younger than 13 years old. Apple encourages the use of privacy policy in the App Store Review Guidelines and iOS Developer Program License Agreement. Apple specify that developers should insure the application is compliant with all laws of the country the application is distributed in. On viewing the App Store Review Guidelines one can see that all Privacy by Design fundamental principles and data violation possibilities are covered by Apple verification. However, the exact evaluation process used by Apple remains secret and some privacy-intrusive applications may appear in the store. Until recently, Apple authorized the use of device identification. This identification number was not considered private. Many applications used this number to uniquely identify their users therefore many applications were considered privacy intrusive [26].

Google Play Terms of Service do not require any privacy policy to be added to the Android applications. Google provides an option to include the privacy policy but does not verify or enforce it. Google Developers Documentation provides recommendations and warns that the developer has a responsibility to ensure the application is compliant with the laws of the countries in which the application is distributed.

Some developers include license agreement and privacy policy. According to [27] only 48% of the top 25 Android paid applications, 76% of the top 25 Android free applications, 64% of iOS paid applications and 84% of iOS free applications have included the privacy policy. Android includes the permission list in the store and this can be considered a privacy policy, but, as previously discussed, the list is unclear to the final user.

III. Privacy respecting permission system

Mobile phones have significant data flow: information can be received, stored, accessed and sent by the application. Data can be entered by the user, retrieved from the system sensors or applications, come from another mobile application, arrive from the server or from other devices. Data can be shared on the phone with another application, with the server or another device.

We propose to focus permissions on data and the action that can be carried out on this data, rather than on the technology used. The definition of purpose of the data usage is included in our permission system.

Privacy Policy should be short and clear. Users should have a global vision of the data usage and functionalities before they install an application. Users rarely read long involved policies, especially when they want a service and feel they have no choice but to accept all permissions. Our permission system enables a simple policy to be generated with a list of permissions.

A. Privacy by Design Permission System

The permission system is integrated into the mobile operating system; well designed, it makes a proactive privacy-respecting tool embedded into the system.

We model our permission system with an access control model. We choose discretionary access control where only data owner can grant access. The user should be able to control the data, therefore we consider the user is a unique owner of all information related to him.

R_{app} is a set of $rules$ assigned to the application. We define a $rule$ as an assignment of the $\mathcal{R}ight$ over an $\mathcal{O}bject$ to the $\mathcal{S}ubject$.

$$\forall rule \in \mathcal{R}_{app},\, rule = (s,\, r,\, o) \tag{1}$$

where $s \in \mathcal{S}ubject,\, r \in \mathcal{R}ight,\, o \in \mathcal{O}bject$

We define the mobile application as a $\mathcal{S}ubject$. $\mathcal{O}bjects$ are the user-related data, such as e-mail, contact list, name and surname, phone number, address, social networks friend list, etc.

$$\mathcal{S}ubject = Mobile\, Application \tag{2}$$

$$\mathcal{O}bject = \{Phone\#,\, Name,\, Contacts,\, \cdots\} \tag{3}$$

To define Right we have to introduce $\mathcal{A}cation$ and $\mathcal{P}purpose$.

We define a set of actions denoted $\mathcal{A}ction$ as all actions can be carried out on user private data by the application: load, access, process, store and transfer of the data. We define the $\mathcal{A}ction$ as follows:

$$\mathcal{A}ction = \{Load, Access, Process, Store, Transfer\} \tag{4}$$

$\mathcal{P}urpose$ is assigned by the application and depends on the service. For example, purpose could be 'retrieve forgotten password', 'display on the screen', 'calculate the trust score', 'send news', 'automatically retrieve nearest restaurant' and 'automatically attach location to the message'.

$$\mathcal{P}urpose = \{Retrieve\, forgotten\, password,\, \cdots\} \tag{5}$$

We define the set of rights denoted $\mathcal{R}ight$ for all actions except the $Store$ action as follows.

$$\forall r \in \mathcal{R}ight,\, \forall a \in \mathcal{A}ction - \{Store\}\,, r = (a,\, p) \tag{6}$$

where $p \in \mathcal{P}urpose$.

We define the set of rights denoted $\mathcal{R}ight$ with the action equal to $Store$ having an additional parameter $\mathcal{T}ime$ informing about the time storage. We define the period $[0, T]$ as an application lifetime.

$$\forall r \in \mathcal{R}ight,\, if\, a\, = \{Store\}\,, r = (a,\, p,\, t) \tag{7}$$

where $p \in \mathcal{P}urpose,\, a \in \mathcal{A}ction,\, t \in [0, T]$

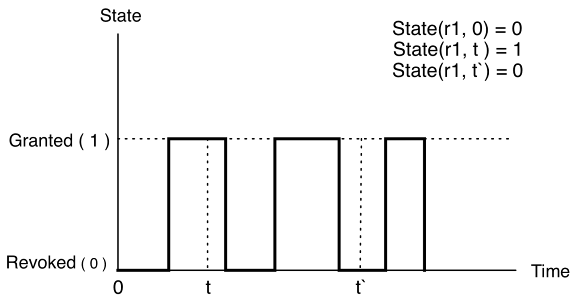

Figure 1. Example of state modification diagram for a given permission

The time storage can indicate the number of days, hours or months data is stored or the time regarding the application lifecycle: until the application is closed, until the application is stopped, until the application is uninstalled. All personal data available during the deinstallation of the application is deleted regardless of defined period, as it cannot exceed the application lifetime.

Each $rule$ should be explicitly asked of the user to be assigned. Thus each $rule$ has a $\mathcal{S}tate$: granted or revoked. To respect the Privacy by Default notion the default $\mathcal{S}tate$ of the permission in installed applications is $Revoked$. We propose to define the $\mathcal{S}tate$ as follows:

$$\forall rule \in \mathcal{R}_{app},\ \forall t \in]0,T]$$

$$\mathcal{S}tate(rule,t) = \begin{cases} Granted, & user\,accepts\,the\,rule \\ Revoked, & user\,declines\,the\,rule \end{cases} \quad (8)$$

$$\forall rule \in \mathcal{R}_{app},\ \mathcal{S}tate(rule,0) = Revoked \quad (9)$$

The State of a rule $r1 \in \mathcal{R}_{app}$ changes over the application lifetime. The diagram in Figure 1 shows an example of the state modification.

A given user should be informed about the use of the permission: the $rule$ should be $defined$ and $displayed$ for each $o \in \mathcal{O}bject$. The Figure 2 shows the recapitulative schema of the rule definition.

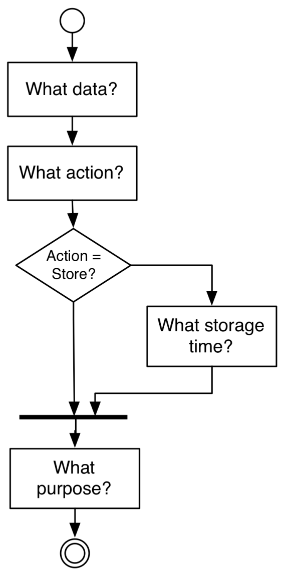

Figure 2. Activity diagram for the rule definition

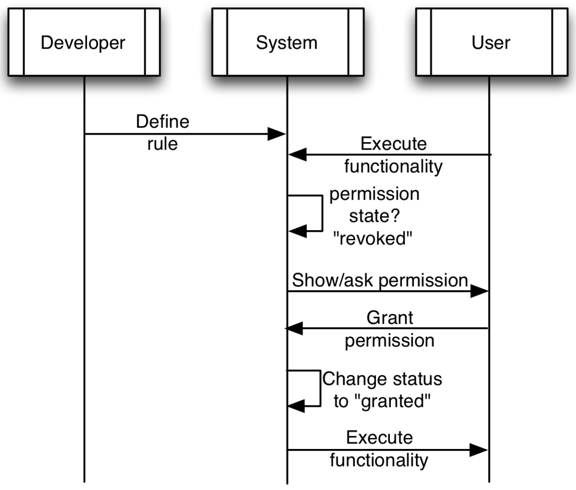

Figure 3. Sequence diagram: first use of one permission

Users should be able to grant or revoke the displayed permission. Finally, user should dispose the settings with all $rule \in \mathcal{R}_{app}$ to be able to $Grant$ or to $Revoke$ individual permissions in later use.

The sequence diagram in Figure 3 shows the pattern in action when the permission is used for the first time.

Thus, we obtain the pattern the developer can follow to design the permission system. The developer should $define$ the permission for all personal data ($\mathcal{O}bject$) used in the application ($\mathcal{S}ubject$).

The permission ($rule$) is stored inside the application with its current $State$. The default $State$ is $Revoked$. Developers should verify that the permission is $displayed$ and $requested$ at least once and that it is $available$ in settings for modification. The simple privacy policy can be generated from the list of defined rules and added to the store.

IV. Application

In this section, we propose an example of permission system made for the application of trust evaluation of friends on social networks named Socializer 1.0 [28]. We choose this application because its service is based on private information and cannot be anonymous, the PbD notion should be integrated into this application. This application needs user friend lists of different social networks (Facebook, Twitter, LinkedIn) and the contact list to view friends and common friends to calculate the overlap of friends in different social networks and contact list and to evaluate the trust of Facebook friends.

Contact list is found on the smartphone, therefore the application needs an $Access$ right.

$$r_1 = (s,\ (Access, \mathcal{P}_{r1}),\ ContactList) \quad (10)$$

where s is a $Subject$ defined as the application Socializer 1.0; $\mathcal{P}_{r1}$ is a purpose defined as a set of p_1, p_2 and p_3: $\mathcal{P}_{r1} = \{p_1, p_2, p_3\}$; p_1= view the list of contacts on the screen; p_2= calculate the overlap; p_3= calculate the trust.

Social networking friends lists should usually be retrieved from the server of a given social network thereby the load and store actions should be defined. The Facebook friends list with the contact list is essential to assure the overlap and trust functionality.

$$r_2 = (s,\ (\mathit{Load}, \mathcal{P}_{r2}),\ FacebookFriendList) \quad (11)$$

$$r_3 = (s,\ (\mathit{Store}, \mathcal{P}_{r2}, t_1),\ FacebookFriendList) \quad (12)$$

where t_1 is a storage time defined as: while the application is installed; $\mathcal{P}_{r2}$ is a purpose defined as a set of p_2, p_3 and p_4: $\mathcal{P}_{r1} = \{p_2, p_3, p_4\}$; p_4= view the Facebook friend list on the screen.

For each Facebook friend, the list of common friends with the user is necessary for trust calculation.

$$r_4 = (s,\ (\mathit{Load}, p_3),\ FacebookCommonFriendLists) \quad (13)$$

A list of friends from other social networks improves the scores of overlap and trust.

$$r_5 = (s,\ (\mathit{Load}, \mathcal{P}_{r5}),\ TwitterFriendList) \quad (14)$$

$$r_6 = (s,\ (\mathit{Store}, \mathcal{P}_{r5}, t_1),\ TwitterFriendList) \quad (15)$$

where $\mathcal{P}_{r5}$ is a purpose defined as a set of p_5 and p_6: $\mathcal{P}_{r5} = \{p_5, p_6\}$; p_5= view the Twitter friend list on the screen; p_6= improve the overlap and trust score with Twitter friends;

$$r_7 = (s,\ (\mathit{Load}, \mathcal{P}r_7),\ LinkedInFriendList) \quad (16)$$

$$r_8 = (s,\ (\mathit{Store}, \mathcal{P}r_7, t2),\ LinkedInFriendList) \quad (17)$$

where $\mathcal{P}_{r7}$ is a purpose defined as a set of p_6 and p_7: $\mathcal{P}_{r7} = \{p_{6'}, p_7\}$; $p_{6'}$= improve the overlap and trust score with LinkedIn friends; p_7= view the LinkedIn friend list on the screen.

The second functionality of the application is to evaluate the behaviour of Facebook and Twitter friends to indicate potentially dangerous contacts. The behaviour evaluation is calculated by analysing the messages published by the given friend over time. The application needs a permission to load messages.

$$r_9 = (s,\ (\mathit{Load}, p_8),\ TwitterFriendMessages) \quad (18)$$

where p_8 is a purpose defined as 'calculate the Twitter friends behavior'.

$$r_{10} = (s,\ (\mathit{Load}, p_9),\ FacebookFriendMessages) \quad (19)$$

where p_9 is a purpose defined as 'calculate the Facebook friends behaviour'.

The third functionality proposes to view today Facebook and Twitter messages on the screen for the user.

$$r_{11} = (s,\ (\mathit{Store}, p_{10}, t_2),\ TodayTwitterFriendMessages) \quad (20)$$

where p_{10} is a purpose defined as 'view today Twitter messages'; t_2 is a storage time defined as: one day.

$$r_{12} = (s,\ (\mathit{Store}, p_{11}, t_2),\ TodayFacebookFriendMessages) \quad (21)$$

where p_{11} is a purpose defined as 'view today Facebook messages'.

The user has the option to share the scores by posting new messages on Facebook and Twitter. The user can also contribute to the research by sending the trust and behaviour anonymized statistics to the developer. Those actions should be taken with the user's express consent.

$$r_{13} = (s,\ (Transfer, p_{12}),\ FacebookFriendTrustScore) \quad (22)$$

where p_{12} is a purpose defined as 'share results on Facebook'.

$$r_{14} = (s,\ (Transfer, p_{13}),\ FacebookFriendTrustScore) \quad (23)$$

where p_{13} is a purpose defined as 'share results on Twitter'.

$$r_{15} = (s,\ (Transfer, p_{14}),\ TrustAndBehavior) \quad (24)$$

where p_{14} is a purpose defined as 'contribute to the improvement of the methodology'.

The final application has 15 rules that should be accepted by the user.

$$\mathcal{R}_{app} = \{r_1, r_2, r_3, \cdots, r_{14}, r_{15}\} \quad (25)$$

The rules r_1, r_2, r_3 and r_4 have a common purpose, all rules should be accepted to achieve the functionality mentioned in the purpose: 'calculate the trust'. Similarly, r_5 should be grouped with r_6 and r_7 with r_8. The rules from r_9 to r_{15} should be accepted one by one to achieve the aforementioned purpose (to get the functionality). Finally, we obtain 10 permissions to be added to the application to propose control to the user. The Table II recapitulates permissions.

To compare with actual permission systems, (a) iOS requires contact list, Facebook and Twitter access permissions. (b) Android requires 'internet', 'read_contacts' and 'get_accounts' access permissions. Facebook and Twitter connections are managed with APIs that requires permissions to be declared on the platform but the permission management will not be available for users in the mobile application by default. iOS permissions give certain transparency to the user but Android permissions are vague.

We obtained more fine-grained control of the application and the data including permissions to all necessary personal data, actions carried out on this data and corresponding purposes. The recapitulation table (Table II) clearly shows what data are used for what purpose. This kind of table can be added to the privacy policy to improve transparency.

V. Conclusion and Future Work

We modelled a permission system for mobile application regarding Privacy by Design. This permission system is data-oriented, thus the final user can easily understand what personal data is involved. We include the action that is missing from current iOS and Android permission systems, such as load and transfer, that improves transparency of the application.

The novelty is to include the purpose of the data usage into the permission system. The clear purpose will help users to understand better why the data is used and to judge whether this permission is needed. Purpose in permission also forces developers to apply the minimization principle: a developer cannot use the data if he cannot define the clear purpose of usage. The compulsory purpose definition should help guard against the abusive permission declaration 'in case'. Finally, purpose gives the user more fine-grained control, as the same

TABLE II
TABLE RECAPITULATING PERMISSIONS NEEDED FOR THE APPLICATION (LAST COLUMN IS A PERMISSION NUMBER)

Object	Action	Purpose	#
Contacts list	Load	View; Calculate Overlap and Trust	1
Facebook friends list	Load; Store		
Facebook common friends	Load		
Twitter friends list	Load; Store	View; Improve Overlap and Trust	2
LinkedIn friends list		View; Improve Overlap and Trust	3
Twitter messages	Load	Tw. friends behaviour	4
Facebook messages		Fb. friends behaviour	5
Today Tw. messages	Store	View Tw. messages	6
Today Fb. messages		View Fb. messages	7
Trust score	Transfer	Publish to Twitter	8
		Publish to Facebook	9
Trust and behaviour	Transfer	Contribute to research	10

data can be allowed to be used for one functionality but not for another. It is important for our system to integrate clear purpose and not a vague explanation (e.g., 'measure the frequency of application utilization' instead of 'improve user experience').

PbD states that the user should have a control over his data and be Privacy by Default, therefore permissions used in the application are revoked by default. Users should be clearly informed and asked to grant permission. Moreover, users should keep control of permissions during all the application use-time, therefore the permission setting must be available.

Our permission system helps developers to be compliant with the law; it defines what permissions the developer should add to the application, but in the current state it cannot ensure that all necessary permissions are really added. Our pattern indicates to the developer what should be added to the application to be more transparent, but if he decides to transfer data without asking permission, the pattern allows this (even if it goes against European law). The generated privacy policy can give the first indication permitting evaluation if the data usage is reasonable and the purpose is clear. Manual verification of an application can show the anomaly in permission system usage.

We aim to build a framework for the automatic management of a new permission system to simplify the developers work. We target the Android system first as it is more crucial due to the more open communication and data sharing and the vagueness of the current permission system.

The impact of new privacy-respective permission systems on users and developers could be measured by conducting the real-life experience. We aim to measure the impact of integration of the new permission system on design and development time, as well as particular situations and difficulties in applying the pattern. We also aim to evaluate user perceptions. We have an additional hypothesis that the explicative application with high transparency improves the user experience and leads to more positive perception of the same application, therefore the use of our permission system gives benefits to the application owner.

This work can be continued by developing an enforcement system automatically verifying whether all necessary permissions are properly defined. Many works propose systems monitoring mobile data flow, therefore the permission verification system can be based on one of the already proposed systems. Another important aspect for the developer is to be able to prove the application is compliant with the law. The system generating on-demand reports on the data, including the private data usage, can be developed.

REFERENCES

[1] TRUSTe. Consumer Mobile Privacy Insights Report. [retrieved: Apr., 2011]

[2] A. Cavoukian, "Privacy by design: The 7 foundational principles," 2009.

[3] E. data protection regulators, "Opinion 02/2013 on apps on smart devices," EU, Tech. Rep., Feb. 2013.

[4] K. D. Harris, "Privacy on the go," California Department of Justice, Jan. 2013, pp. 1–27.

[5] W. Shin, S. Kiyomoto, K. Fukushima, and T. Tanaka, "Towards Formal Analysis of the Permission-Based Security Model for Android," in Wireless and Mobile Communications, 2009. ICWMC '09. Fifth International Conference on. IEEE Computer Society, 2009, pp. 87–92.

[6] K. W. Y. Au, Y. F. Zhou, Z. Huang, P. Gill, and D. Lie, "Short paper: a look at smartphone permission models," in SPSM '11 Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices. ACM Request Permissions, Oct. 2011, pp. 63–67.

[7] R. Stevens, J. Ganz, V. Filkov, P. T. Devanbu, and H. Chen, "Asking for (and about) permissions used by Android apps." MSR, 2013, pp. 31–40.

[8] A. R. Beresford, A. Rice, N. Skehin, and R. Sohan, "MockDroid: trading privacy for application functionality on smartphones," in HotMobile '11: Proceedings of the 12th Workshop on Mobile Computing Systems and Applications, ser. HotMobile '11. ACM Request Permissions, Mar. 2011, pp. 49–54.

[9] 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," in Proceedings of the 18th ACM conference on Computer and communications security, ser. CCS '11. ACM Request Permissions, Oct. 2011, pp. 639–652.

[10] M. Nauman, S. Khan, and X. Zhang, "Apex: extending Android permission model and enforcement with user-defined runtime constraints," in ASIACCS '10: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security. ACM Request Permissions, Apr. 2010, pp. 328–332.

[11] S. Holavanalli et al., "Flow Permissions for Android," in Automated Software Engineering (ASE), 2013 IEEE/ACM 28th International Conference on, 2013, pp. 652–657.

[12] J. Tam, R. W. Reeder, and S. Schechter, "Disclosing the authority applications demand of users as a condition of installation," Microsoft Research, 2010.

[13] S. Egelman, A. P. Felt, and D. Wagner, "Choice Architecture and Smartphone Privacy: There's a Price for That," in Proceedings of the 11th Annual Workshop on the Economics of Information Security (WEIS), 2012.

[14] M. Lane, “Does the android permission system provide adequate information privacy protection for end-users of mobile apps? ,” in 10th Australian Information Security Management Conference, Dec. 2012, pp. 65–73.

[15] P. G. Kelley et al., “A conundrum of permissions: Installing applications on an android smartphone,” in Proceedings of the 16th International Conference on Financial Cryptography and Data Security, ser. FC’12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 68–79.

[16] A. P. Felt, E. Chin, S. Hanna, D. Song, and D. Wagner, “Android permissions demystified,” in CCS ’11: Proceedings of the 18th ACM conference on Computer and communications security. ACM Request Permissions, Oct. 2011, pp. 627–638.

[17] P. Berthomé and J.-F. Lalande, “Comment ajouter de la privacy after design pour les applications Android? (How to add privacy after design to Android applications?),” Jun. 2012.

[18] M. Egele, C. Kruegel, E. Kirda, and G. Vigna, “PiOS: Detecting Privacy Leaks in iOS Applications.” 2011.

[19] C. Gibler, J. Crussell, J. Erickson, and H. Chen, “AndroidLeaks: Automatically Detecting Potential Privacy Leaks in Android Applications on a Large Scale,” in Trust and Trustworthy Computing. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 291–307.

[20] T. Vidas, N. Christin, and L. Cranor, “Curbing android permission creep,” in In W2SP, 2011.

[21] Y. Zhang et al., “Vetting undesirable behaviors in android apps with permission use analysis,” in CCS ’13: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, Nov. 2013, pp. 611–622.

[22] W. Xu, F. Zhang, and S. Zhu, “Permlyzer: Analyzing permission usage in Android applications,” Software Reliability Engineering (ISSRE), 2013 IEEE 24th International Symposium on, 2013, pp. 400–410.

[23] W. Luo, S. Xu, and X. Jiang, “Real-time detection and prevention of android SMS permission abuses,” in SESP ’13: Proceedings of the first international workshop on Security in embedded systems and smartphones. ACM Request Permissions, May 2013, pp. 11–18.

[24] A. P. Felt, K. Greenwood, and D. Wagner, “The effectiveness of application permissions,” in WebApps’11: Proceedings of the 2nd USENIX conference on Web application development. USENIX Association, Jun. 2011, pp. 7–7.

[25] R. Böhme and S. Köpsell, “Trained to accept?: a field experiment on consent dialogs,” in CHI ’10: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM Request Permissions, Apr. 2010, pp. 2403–2406.

[26] E. Smith, “iPhone applications and privacy issues: An analysis of application transmission of iPhone unique device identifiers UDIDs,” October 2010.

[27] “Mobile Apps Study,” Future of Privacy Forum (FPF), pp. 1–16, Jun. 2012.

[28] C. Perez, B. Birregah, and M. Lemercier, “A smartphone-based online social network trust evaluation system,” Social Network Analysis and Mining, vol. 3, no. 4, 2013, pp. 1293–1310.