Functionalities as superior predictor of applications privacy threats

Alessio De Santo[1], Brice Quiquerez[1] and Cédric Gaspoz[1],

[1] Information Systems and Management Institute, HES-SO // University of Applied Sciences Western Switzerland, HEG Arc, Neuchâtel, Switzerland
{alessio.desanto, brice.quiquerez, cedric.gaspoz}@he-arc.ch

Abstract. Applications are invading our devices whether in our phones, computers and TVs or in our cars, appliances and cameras. Providing great benefits in terms of added functionalities and customization, these applications also put a lot of pressure on our privacy. In order to offer their services, these applications needs access to data stored on the devices or captured by various sensors. Currently all systems have implemented a permissions based framework for granting access to various data, based on the requests made by the applications. However, it is difficult for most users to make informed decisions when they are asked to grant these accesses. In this paper, we present a paradigm shift from a permissions to a functionalities framework. We show that users are consistent in understanding functionalities offered by applications and we propose an ontology for bridging the gap between understandable functionalities and technical permissions.

Keywords: threatening application, privacy, malware, user privacy concerns

1 Introduction

In the last few years, application markets have rapidly become widely popular [22]. Listed in dedicated markets, the user can browse and choose applications from thousands of alternatives. To choose an application, users can mainly rely on the name, the editor, the description, the screenshots and eventually the users' reviews. Application developers must request permission to access device resources. This resources could also include some sensitive personal information. A permission list is presented to the user who has no other choice but to accept them in exchange for the application's functionalities. The permissions are listed with the implicit assumption that the user is able to determine whether the listed permissions are appropriate [23]. In our opinion, access to some features and data of the user's device should be justified by the functionalities offered by the application. However, when the user has to accept a permission without knowing the functionalities involved, it is non-trivial to classify an application as privacy invasive. Previous researchers have identified here a potential issue for user privacy [2, 8, 10].

Á. Rocha et al. (eds.), *New Advances in Information Systems and Technologies*,
Advances in Intelligent Systems and Computing 444,
DOI 10.1007/978-3-319-31232-3_25

2 State of the art

Detection of potential malware, including viruses, worms, spyware and Trojans, has become a big issue in literature today. Malware or malicious software, can be defined as "any software that does something that causes harm to user, computer or network" [25]. With the advent of applications markets, an easy meeting point between consumers and developers has been created. Unfortunately, these markets also provide an easy distribution mechanism for developers with malicious intent to distribute malware [23]. Furthermore, due to the nature of mobile devices, users are exposed to new threats [3]. While being mainly based on existing kernels, the mobile nature of applications makes conventional techniques no longer adapted for detecting and reacting to malware [3]. Several researchers have tried to find a way in protecting the users [2, 7], but it's actually hard to know when an application could be privacy threatening. In fact, we should make a distinction between malicious applications and privacy threatening applications. In our opinion, a malicious application aims to bypass the platform securities to access your personal data. On the other hand, a privacy threatening application could collect data without bypassing any security system, but simply by requiring access to personal information stored on the device.

To address these market-security issues, the various operating system platforms use different approaches. A manual inspection can be run before the application is published on the market. This manual intervention involves teams of experts who are responsible for identifying violations of developer policies or malicious scripting contained in the source code of the application. This first practice allows us to recognize malicious patterns and protect the market from potential malware, but it doesn't necessarily help to determine whether the different information and resources used by the application are appropriate. Therefore, the privacy and consequently the security of the users relies on the user's decision to install an application by accepting a list of permissions. The implicit assumption is made that user is able to determine whether the listed permissions are appropriate or not. This permission list is presented to the user at the time of installation or at the first use of the application, depending of the platform. Through this permission list, the user is asked to grant access to specific smartphone functions such as network access, GPS location, stored personal information, etc. Thus, when accepting a permission, the user will grant, consciously or unconsciously, access to the personal information stored on the device. Once this is granted, the application can access and collect the personal information it needs unnoticed.

While most of previous work has focused on the code [7, 13, 14, 30], and runtime behavior [3, 9, 16, 28], to detect behavioral anomalies and therefore malware, only a few researchers have considered the potential privacy threat of an application through permissions [4, 23, 29]. Pandita et al. [23] summarize the caveat of this work as: *what does the user expect?* Through their research, they have managed to bridge the semantic gap between what users expects an application to do and what it actually does, by analyzing through Natural Language Processing (NLP) whether the application description provides any indication of why the application needs a permission. Nevertheless, a potential privacy threat exists. An application can totally justify the required permissions by furnishing corresponding functionalities, but in counterpart massively sales some confidential personal data. Moreover, not everyone places the same value on the information, as privacy concerns are not the same for every user's profile. Through this research, we would like to suggest a three-level scale (Table 1) to

categorize a potential privacy threatening application with respect to a given user profile. This categorization emerges by contrasting applications descriptions, applications permissions and user privacy concerns. Applications descriptions represent the human perceptible text which will in part influence the installation of the application. By contrast, the application's permissions represent the not-so-human perceptible list of resources accessed by the application. Ideally the application description and therefore the functionalities should be strongly linked to the permissions requested. Although, as we will see through this research, this is not always true. The potential privacy threatening application categorization (Table 1) that we are suggesting brings out the different possible situations.

Table 1. Potential privacy threatening application categorization

Category	*Description*
Red	The application description doesn't provide any indication for why the application needs a permission.
Green	The application description provides indication for why the application needs a permission
Orange	Red + user's non-privacy concerning permissions. Green + user's privacy concerning permissions.

This categorization introduces a new component in the determination of whether an application has to be considered as potentially privacy threatening: the user's privacy concerns. This component represents the sensitivity of each individual user regarding their personal data. This sensitivity may be measured using the De Santo and Gaspoz seven mutually exclusive permission threatening categories [24]. In Table 2, the seven categories are associated with the corresponding Android permissions. These categories of permissions represent all the potentially threatening permissions that could be requested from an application user.

Table 2. User's privacy risk sensibility measurement

Category	Android Permissions
Location	access_coarse_location, access_fine_location, access_location_extra_command, access_mock_location,
System settings and status	access_wifi_state, battery_stats get_tasks, read_phone_state, read_sync_settings, read_sync_stats
Profile and contacts	get_accounts, read_contacts, read_call_log,
Messages and social	read_sms, read_social_stream, read_user_dictionary
User profile and interests	read_profile, subscribed_feeds_read, read_history_bookmarks,
Calendar	read_calendar
Audio, photo and video	record_audio, camera, read_external_storage

By answering a Likert five-scale survey on the users' concerns about each one of these categories, users provide a profile of their privacy concerns. Matching the user profile with the permissions required by an application allows us to categorize applications in a unique and personal way.

3 Functionalities

Recent studies have explored the understanding of smartphone users regarding the permissions that they are required to grant while installing a new application, as well as the resulting privacy concerns [5, 8, 11]. To respond to the users' lack of understanding of the effects of these permissions on their privacy, researchers proposed several options. Kelley et al. [20] showed that when users are presented with a list of permissions before selecting a given application in the store, they tend to choose the application requesting fewer permissions. Other studies found that fine grain controls over the permissions can improve the user's ability to reduce privacy threats by limiting access to permissions providing a real value for the user [2]. However, after developing a permission map to gain a complete understanding of the permission infrastructure, Felt et al. [10] found that many applications are requesting permissions that are not even being used in their codebase. They attribute these to "developer confusion over the permission system (confusion over permission names, related methods, deputies, and deprecated permissions)" [10]. In another study [12] they found that common cases of over-privilege are due to the developer's lack of attention during the release process, leaving permission requests from copied snippets or development testing as well as the fact that they are incentivized to ask for more permissions upfront due to the fact that a change in the permissions requested will trigger a manual update of the application on the user's smartphone. Finally, Kelley et al. "found that participants continued to report that other characteristics of applications are as important as or more important than permissions, including: cost, functionality, design, simplicity, rating, number of ratings, reviews, downloads, size, and others" [20].

Therefore, we have chosen to switch our focus from the permissions to the expected functionalities. Drawing on Christensen et al. [6] "jobs to be done" approach where products should be "hired" by the customers in order to solve a specific problem that they find themselves with, we postulate that users are able to understand the job that they will be able to do with a given application. Thus, instead of forcing users to understand the permissions framework and the consequences for their privacy of choosing one permission, we ask users to identify which functionalities are provided by an application.

Each application comes with a description written by the developer which presents, in plain text, the benefits of the application in order to attract potential users. Besides this description, the application also comes with a list of requested permissions. Whereas users can understand a plain text, we have seen that they have problems understanding the meaning and scope of the permissions. Thus, using functionalities, we can bridge the gap between the intelligible descriptions and the less intelligible permissions. In order to be able to provide a list a functionalities to the users, we first had to identify potential functionalities based on the applications' descriptions and then create an ontology of these functionalities and their relationships with the underlying permissions.

Various techniques were explored for establishing a list of functionalities that would cover a maximum of applications. Two main approaches were retained to deduce the functionalities. The first one, starting from the descriptions, to derive a list of functionalities and the second one, starting from the permissions, to achieve the same. Both approaches were used by selecting samples of applications having a common permission and applying a text mining algorithm to their corresponding descriptions. The result is a concept card by threatening permission representing the more frequently used words. Figure 1 illustrates the concept cards issued for the text mining of 30,000 descriptions using the Android permission "p_read_calendar".

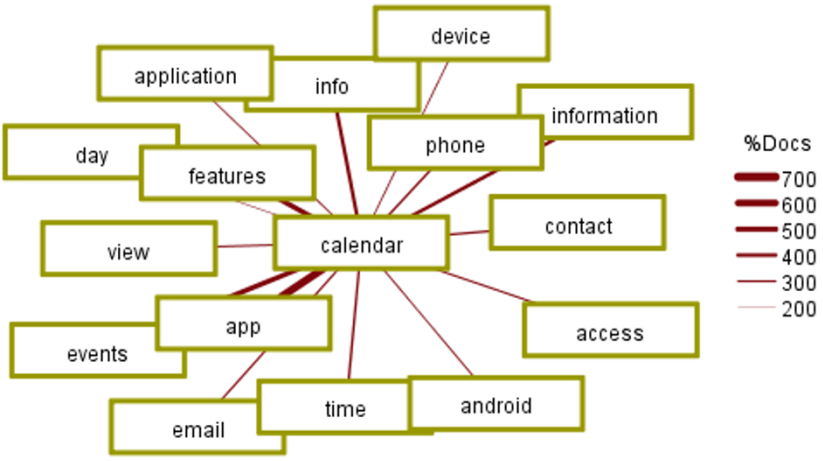

Figure 1. Concept cards

By contrasting the results obtained with each sample of distinct threatening permission (Table 2), we removed common concepts and noise, resulting in a list of the most characteristic concepts for each given permission. This allowed us to achieve a first list of 56 functionalities matching threatening permissions.

The second step was the creation of an ontology comprising applications, functionalities and permissions in order to enlighten the relations between each of them. We use the term ontology not in its philosophical sense as "an explicit, partial account of the intended models of a logical language" [15] but as an explicit representation of concepts and their relationships [27]. Moreover, using an ontology of the domain will allows us to draw inferences in order to make assertions about applications and their permissions. This is due to the fact that "when a number of modalities are specified as being appropriately related, either positively or negatively, to a variable, a rule can be implemented to infer a complementary relationship between the modalities themselves" [1]. Our ontology is based on three concepts: application, functionality and permission. Whereas permissions are already presented in Table 2, Table 3 presents the functionalities associated with the permissions p_localisation and p_camera.

Table 3. Functionalities associated with the permissions p_localisation and p_camera

Functionality	*Definition*
Augmented reality	Displaying information overlays and digital content tied to

	physical objects and locations by means of the camera.
Take a picture	Taking photographs.
Record a movie	Recording movies.
Face recognition	Identifying a person using the device's camera.
Localisation/navigation	Determining the place where you are or your way to a given location or POI.
QR/barcode scanner	Reading printed codes and triggering specific actions based on the content of the code.
Tourism Information/bookings	Providing touristic information (POI, reviews, ...) and/or allowing travel bookings related activities (hotel, transport, ...).
Weather forecast	Providing weather forecasts based on the current location or any location in the world.
Flashlight	Transforming the flash and/or display in a small portable lamp.
Photos and videos Editor	Editing photos and/or videos stored on the device.
Upload/share/view personal photos/videos	Sharing, uploading or viewing photos and/or videos stored on the device.

We defined four types of relations: application *provides* a functionality, functionality *builds on* another functionality, functionality *needs* a permission and application *requests* a permission. These relations provide two links between a given application and the permissions. The first link is the technical link as defined by the developer of the application. An application *requests* some permissions that are technically necessary in order to guarantee its operation. The second link is the functional link. An application *provides* some functionalities that in turn *need* permissions. In contrast to the technical link, the relation between an application and some permissions is not the fact of the developer, but the result of the perception of the usefulness of the application to support the jobs to be done by the user. Therefore, given that the second link is technically independent of the first link, we are able to identify applications that request more permissions that are needed in order to provide the announced functionalities (Figure 2).

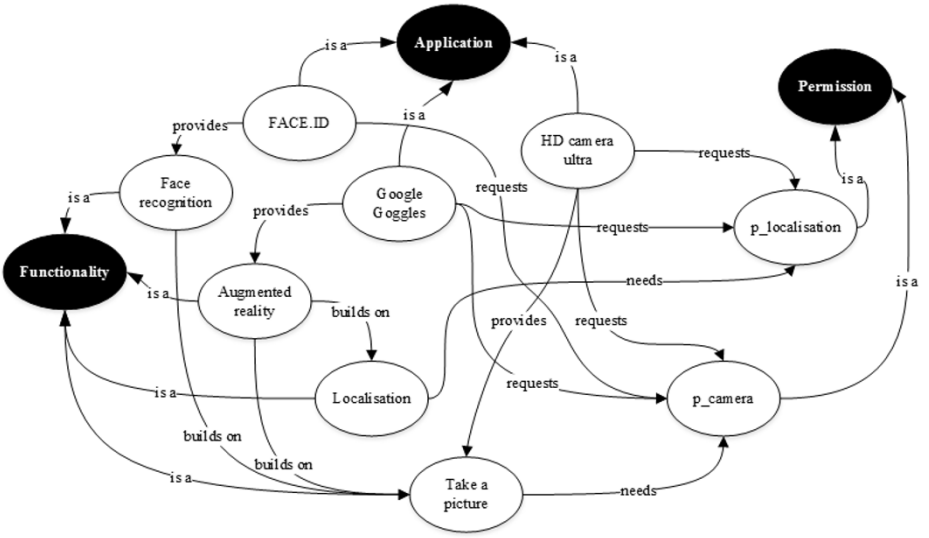

Figure 2. Excerpt of the ontology

Using the provided ontology, given an application providing the functionality of taking pictures and requesting the p_camera and p_localisation permissions, we can deduce that the permission p_localisation is not necessary to support the user in fulfilling its job. At that time, we make no judgement regarding the fact that p_localisation can be useful to add geographical information to the pictures. We only state that it is not needed for fulfilling the job of capturing a scene for future viewing.

Given that we were able to create an ontology of functionalities and their relations with applications and permissions, we should be able to identify applications that request more permissions than are theoretically necessary in order to fulfill their announced functionalities. However, the final missing link to establish is the link between an application and its functionalities. As we have seen, Pandita et al. [23] used NLP in order to extract information about functionalities offered by applications from their description text. This proved to be quite effective, but this approach does not support our main assumption that the user should be able to understand the functionalities provided by an application from its presentation. Therefore, we need to assess if a user is able to correctly assign one or more functionalities to an application, simply by reading its description.

4 Functionalities attribution

Functionalities being elaborated to bridge the gap between applications' descriptions and permissions, it is necessary to assess the comprehension and equivocality of each functionality. In order to avoid attribution errors due to the large amount of possibilities, we focused on smaller subsets of functionalities. However, this research focuses on the 11 functionalities issued from device camera permission (p_camera) and user localisation permission (p_localisation). Before assessment by the general public the comprehension and equivocality of functionalities, we processed a three-step refinement. The result of this process was the ontology presented in chapter 3. This three-step refinement process is based on a sample of 225 applications: 75 applications requiring p_camera permission, 75 requiring p_localisation permission and 75 other applications weren't requiring either of these two permissions.

In the first part of this process we humanized as many as possible of the functionalities issued from the concepts cards. For instance, functionalities such as "compass", "localisation" and "maps" were refined into a less technical one, "navigation".

In the second part of the process, two experts assessed the functionalities through the 225 application sample. However, the entire sample was manually processed to associate each application description with the considered corresponding functionalities. The two experts distinctively conducted the attribution to avoid any mutual influence. Nevertheless, experts commonly refined the functionalities as they were progressing with their assignment, in order to have the same comprehension of the functionalities. The functionalities that required a discussion were refined in order to reach an expert consensus about unequivocal functionalities. In the end, the two experts reached an Iota

coefficient of 0.67, indicating a substantial agreement. After commonly reviewing their attributions the experts agreed on a sample of 211 applications with an Iota coefficient of 0.99 and they excluded applications with poor descriptions or poor clarity. The experts agreed almost perfectly on functionality attribution of this 211 application sample. Within this sample, the functionalities could be attributed in an unequivocal manner according to the experts.

The third step of the process consisted of expanding the functionalities' attribution assessment to any potential applications user. However, we used a crowd platform to manually associate the same 211 applications sample with the same 11 functionalities. When the crowd had a high agreement between the contributors, that would mean that the functionalities were unequivocal to anyone and could therefore be used to universally bridge the gap between the application's descriptions and permissions. A first assessment on 100 applications with 12 contributors allowed us to further refine the functionalities and description of the task, for instance, by renaming the functionality "navigation" as "localization/navigation". The localization of points of interest in the proximity of the user was omitted by some users in the functionality "navigation". The initial "Document Scanner" was also replaced by "QR/Barcode Scanner", and "Tourism Reviews" and "Tourism Bookings" becoming "Tourism Information/Bookings". This kind of incomprehension resulted from contributors' feedback. Full-sized functionality attribution was run on the 211 applications that were completed through 828 trusted judgments by 47 contributors. The results are presented in the next section.

5 Results

The attribution of functionalities relies to some degree on subjective interpretation. Studies measuring the agreement between two or more participants should include a statistic that takes into account the fact that observers will sometimes agree or disagree simply by chance. Janson and Olson's Iota coefficient was used to calculate inter-rater reliability [17, 18]. This Iota coefficient examines chance-corrected agreement and is an extension of Cohen's kappa, as it can be used not only with interval level data but also with multivariate data and with several coders. The interpretation of Iota coefficient is similar to that of the Kappa coefficient, the Iota coefficient being the multivariate version of the Kappa coefficient. An Iota of 1 indicates perfect agreement, whereas an Iota of 0 indicates agreement equivalent to chance. Taking into account that participants had to categorize descriptions using 11 different functionalities, which means that there are 2048 (2^{11}) different categorizations, the Landis and Koch [21] classification could be fairly used to interpret our results.

The results of our experiment on the crowd showed agreement on functionalities going from slight (Take a Picture, Record a Movie) to almost perfect (QR/Barcode Scanner). The multivariate agreement being of 0.54, the overall agreement is moderate on the functionalities perspective.

Table 4. Iota coefficients on functionalities attribution

Functionality	*Iota coefficient*
Augmented reality	0.52
Take a picture	0.17

Record a movie	0.02
Face recognition	0.13
Localization/navigation	0.52
QR/barcode scanner	0.84
Tourism information/bookings	0.48
Weather forecast	0.63
Flashlight	0.74
Photos and videos editor	0.53
Upload/share/view personal photos/videos	0.22
	0.54

It is interesting to notice that the functionalities with the lower rates of agreement are the ones related to the use of the camera and the gallery. We could admit a confusion between those functionalities from the participants. Merging this three categories (Take a picture, Record a movie and Upload/share/view personal photos/videos) which represent in the end the same permission (p_camera) we could increase the agreement rate to a more acceptable level (0.25) which is considered as fair. Further iterations of the assessment could re-label or make further precise this functionality.

Table 5. Iota coefficients on merged functionalities

Functionality	*Iota coefficient*
Augmented reality	0.52
Face recognition	0.13
Localization/navigation	0.52
QR/barcode scanner	0.84
Tourism information/bookings	0.48
Weather forecast	0.63
Flashlight	0.74
Photos and videos editor	0.53
Merged photos/videos functionalities	0.25
	0.55

Using our ontology, we could also determine the corresponding permissions behind our functionalities, resulting in a logically even higher agreement rate of 0.62 which is considered as substantial.

Table 6. Iota coefficients on functionalities attribution

Functionality	*Iota coefficient*
p_camera	0.59
p_localisation	0.64
	0.62

6 Discussion

Assessing the ability of smartphone users to correctly assign functionalities to applications, showed that (1) our functionalities are unequivocal for most users and (2) users are able to derive functionalities from application descriptions. The first result is a confirmation of the quality of the extraction process used in order to create our ontology. Each functionality is specific enough to avoid misattributions but is also comprehensible enough that random individuals are able to understand its meaning. With this validation, we can now extend our ontology to whole application stores. The second result is a confirmation of our hypothesis that users are able to correctly identify functionalities of applications in contrast to their poor understanding of the permissions framework [11]. This is an important contribution as it could potentially establish a new mindset on how to design permissions frameworks in order to increase the understanding of users. It is particularly important in a time where applications are included everywhere, and stores are booming. Applications are now included in smartphones, computers and ebook readers as well as in televisions, kitchen appliances, cars, cameras, sport devices and more to come. They are part of our lives and it will be more and more difficult to ignore them. While it is possible to shut down a smartphone in order to protect our privacy by not registering our location in some circumstances, it is difficult to prevent our car registering and transmitting our location while in use. Therefore this research laid the foundation for a shift from technical to human centered design of permissions frameworks.

Building on these results, we now have to find a way to assign these functionalities to a corpus of more than a million applications. In fact, the very large number of applications precludes a fully manual treatment. To solve this task, we consider a solution using semi-supervised learning algorithms. Semi-supervised learning algorithms are able to learn a classification task using an already classified sample, thereafter applying the learned algorithm to solve new cases. Furthermore, SVM algorithms (support vector machines) usually give good results for classification tasks and are especially well adapted to the specific textual data, including their very high dimensionality [19]. We propose to explore the use of semi-supervised algorithms to apply functionalities to very large corpuses of applications. By carefully leveraging crowdsourcing to prepare sets of labelled applications, we should be able to reliably attribute functionalities to applications across whole stores. Once done, this would allow us to determine the privacy threatening potential (Table 1) of each application in the store based on the users' sensibility to their privacy.

7 Conclusion

This research presents a paradigm shift in how to design permission frameworks that bridge the gap between easily understandable functionalities that are provided by applications and the less understandable technical permissions that are required in order to run the application. As editors are trying to increase their benefits in monetizing their users' profiles through the use of various ads platforms the danger is also increasing for users to see their data going in the wild [26]. Therefore, a comprehensible and efficient

permissions framework is required in order to protect the privacy of the users of these applications.

After designing the first ontology of a permissions framework, we designed an experiment to ask subjects to assign functionalities based on their understanding of the applications descriptions. With an Iota coefficient of 0.62, the subjects had a substantial agreement on the attribution of permissions according to Landis and Koch [21]. This result validated our hypothesis that presenting functionalities instead of permissions to the users could achieve a high level of agreement among subjects regarding permissions requested by applications.

Acknowledgments

This paper is partly based upon work supported by the RCSO ISnet under Award No. 37075.

References

1. Adams, W.A.: A transdisciplinary ontology of innovation governance. Artif. Intell. Law. 16, 2, 147–174 (2007).
2. Beresford, A.R. et al.: MockDroid. Proceedings of the 12th Workshop on Mobile Computing Systems and Applications – HotMobile '11. pp. 49–54 ACM Press, Phoenix, AZ, USA (2011).
3. Bläsing, T. et al.: An android application sandbox system for suspicious software detection. Proc. 5th IEEE Int. Conf. Malicious Unwanted Software, Malware 2010. 55–62 (2010).
4. Chakradeo, S. et al.: MAST: Triage for Market-scale Mobile Malware Analysis. ACM Conf. Secur. Priv. Wirel. Mob. Networks. 13–24 (2013).
5. Chin, E. et al.: Measuring user confidence in smartphone security and privacy. Proceedings of the Eighth Symposium on Usable Privacy and Security – SOUPS '12. p. 1 , Washington, DC, USA (2012).
6. Christensen, C.M. et al.: Finding the Right Job for Your Product. MIT Sloan Manag. Rev. 48, 3, 38–47 (2007).
7. Egele, M., Kruegel, C., Kirda, E., Vigna, G.: PiOS Detecting privacy leaks in iOS applications. Proc. 18th Annu. Netw. Distrib. Syst. Secur. Symp. NDSS 2011. 11 (2011).
8. Egelman, S. et al.: Choice Architecture and Smartphone Privacy: There's a Price for That. The Economics of Information Security and Privacy. pp. 211–236 Springer Berlin Heidelberg, Berlin, Heidelberg (2013).
9. Enck, W. et al.: TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. Osdi '10. 49, 1–6 (2010).
10. Felt, A.P. et al.: Android permissions demystified. Proceedings of the 18th ACM conference on Computer and communications security - CCS '11. pp. 627–637 ACM Press, Chicago, Illinois, USA (2011).
11. Felt, A.P. et al.: I've got 99 problems, but vibration ain't one. Proceedings of the second ACM workshop on Security and privacy in smartphones and mobile devices - SPSM '12. p. 33 ACM Press, Raleigh, North Carolina, USA (2012).
12. Felt, A.P. et al.: The effectiveness of application permissions. Proceedings of the 2nd USENIX conference on Web application development. p. 12 USENIX Association, Berkeley, CA, USA (2011).
13. Gibler, C. et al.: AndroidLeaks: Automatically detecting potential privacy leaks in Android applications on a large scale. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 7344 LNCS, 291–307 (2012).

14. Grace, M. et al.: RiskRanker : Scalable and Accurate Zero-day Android Malware Detection Categories and Subject Descriptors. Proc. 10th Int. Conf. Mob. Syst. Appl. Serv. 281–293 (2011).
15. Guarino, N.: Understanding, building and using ontologies. Int. J. Hum. Comput. Stud. 46, 2-3, 293–310 (1997).
16. Hornyack, P. et al.: These Aren't the Droids You're Looking for: Retrofitting Android to Protect Data from Imperious Applications. Proc. 18th ACM Conf. Comput. Commun. Secur. 639–652 (2011).
17. Janson, H.: Calculating and Reporting Rorschach Intercoder Agreement. March, (2008).
18. Janson, H., Olsson, U.: A Measure of Agreement for Interval or Nominal Multivariate Observations by Different Sets of Judges. Educ. Psychol. Meas. 64, 1, 62–70 (2004).
19. Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Proc. 10th Eur. Conf. Mach. Learn. ECML '98. 137–142 (1998).
20. Kelley, P.G. et al.: Privacy as part of the app decision-making process. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems – CHI '13. pp. 3393–3402 ACM Press, Paris, France (2013).
21. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. Biometrics. 33, 1, 159–174 (1977).
22. McDaniel, P., Smith, S.W.: Not so great expectations: Why Application Markets Haven't Failed Security. Secur. Privacy, IEEE. 8, 5, 76 – 78 (2010).
23. Pandita, R. et al.: WHYPER : Towards Automating Risk Assessment of Mobile Applications W HYPER : Towards Automating Risk Assessment of Mobile Applications. USENIX Secur. Symp. 527–542 (2013).
24. De Santo, A., Gaspoz, C.: Influence of Users' Privacy Risks Literacy on the Intention to Install a Mobile Application. In: Rocha, A. et al. (eds.) New Contributions in Information Systems and Technologies. pp. 329–341 Springer International Publishing, Cham (2015).
25. Sikorski, M., Honig, A.: Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software. (2012).
26. Stevens, R. et al.: Investigating User Privacy in Android Ad Libraries. Workshop on Mobile Security Technologies (MoST). p. 10 , San Francisco, California, USA (2012).
27. Uschold, M., Gruninger, M.: Ontologies: principles, methods and applications. Knowl. Eng. Rev. 11, 2, 93–136 (1996).
28. Yan, L., Yin, H.: Droidscope: seamlessly reconstructing the os and dalvik semantic views for dynamic android malware analysis. Proc. 21st USENIX Secur. Symp. 29 (2012).
29. Zhou, Y. et al.: Hey, You, Get Off of My Market: Detecting Malicious Apps in Official and Alternative Android Markets. Proc. 19th Annu. Netw. Distrib. Syst. Secur. Symp. 2, 5–8 (2012).
30. Zhou, Y., Jiang, X.: Dissecting Android Malware: Characterization and Evolution. 2012 IEEE Symp. Secur. Priv. 4, 95–109 (2012).