Studying the Effectiveness of Android Application Permissions Requests

Kevin Benton, L. Jean Camp, Vaibhav Garg
School of Informatics and Computing, Indiana University, Bloomington, IN
{*KTBenton,LJCamp,GargV*}*@Indiana.edu*

Abstract

Popular platforms including *Android* and *Facebook* have adopted a permissions-based model. Under this model applications (apps) are required to declare specific access to user information required for functionality. We conducted two user studies on Amazon's Mechanical Turk to test the efficacy of these permissions requests on the Android platform. We found permissions were ineffective, even with the addition of an additional text warning. Conversely, we found that an app's download count had a strong effect on app installations. In order to determine if it was a failure of our text-based warning, we ran a second experiment with a previously proven visual indicator.

I. Introduction

Due to the prevalence of privacy-invasive spyware and malware in traditional computing environments, newer platforms have shifted towards a permissions-based model (e.g. *Facebook* and *Android*). Under this, applications must explicitly request access to classes of sensitive information such as location, contacts, etc. These requests should clearly indicate the types of sensitive information accessible to an application.

We measured the effectiveness of the Android permissions requests as privacy indicators to end-users. We investigated the correlation of changes in permissions, permissions notification, and the number of advertised downloads with changes in installations.

We investigated five questions. 1) *Do the majority of users understand what common Android permissions allow an application to do?* 2) *Does additional text-based explanation impinge users' understanding of application permissions?* 3) *Are visual warnings more effective than a text-based explanation.* 4) *Do excessive permissions requested by applications inform the corresponding decision to install?* 5) *Or is the decision to accept risk determined by the application's download count?*

We posited that permissions warnings are ineffective in their current state. Furthermore, we introduced short notices to explain the access allowed to an app in combination with its permissions. We measured the effectiveness of these notices by comparing installation rates and post installation regret to the group that only received the standard request. We also compared the effectiveness of text based notices, with previously successful visual cues about privacy risks [25]. Finally, we determined how well users understood what some of the commonly requested permissions allow an app to do.

II. Background and Related Work

The traditional PC threat model has led some platform developers to shift to a new permissions-based execution environment in which applications must explicitly request permissions at install-time. If a user declines the request, the installation is aborted; and, if the user accepts the request, the application is allowed access to only the requested items until the user uninstalls the application. Two popular platforms that have adopted this paradigm are Facebook's Application Platform and Google's Android mobile operating system. This study focuses specifically on the Android platform and aims to evaluate the effectiveness of its permissions request dialog as a privacy signal to the end-user.

Many traditional software vendors would only describe information collection in the end-user license agreements (EULA) [26]. Additionally, more malicious vendors would simply omit this activity from the EULA, so even a full analysis of the EULA would not protect the user from data theft [3]. Like end-user license agreements, access requests are often presented to a user during an application's installation process. EULAs contain limitations on where/how the software can be used, who can use the software, redistribution of the software, and provisions to limit the liability of the vendor in the event the software causes damage or data loss. Additionally, software that transmits data over the Internet may have statements about how the vendor handles user data.

Ideally, every user would read and understand entire privacy policies and make informed decisions about continuing with the software installation process. However, the research in the following sections shows that privacy policies (in and not in EULAs) are an ineffective privacy signaling mechanism.

A. The Broken State of EULAs and Privacy Notices

Jensen, et al. performed a usability analysis of website privacy notices [18]. They used Flesch Reading Ease Score, to determine the average education level required to understand the policy. They found an average reading level of 14.21 (just beyond an associate's degree), with 13% only readable by people with a post-graduate education, and just 6% at high-school level. Additionally, 69% of the policies specified that users would not be notified of changes. So, not only would it be very difficult for a large portion of the population to make informed consent to one of these policies, the terms of the policy could change at any given time without notice.

Grossklags, et al. conducted a similar study on the EULAs of the top 50 most downloaded programs from Download.com for a week in 2006 [15]. They found that only

978-1-4673-5077-8/13/$31.00 ©2013 IEEE

one of the EULAs scored in the ideal range for writing to the general population. The average time required just to read all of the text was 12 minutes, with some requiring more than 45 minutes. They concluded that EULAs in their current legal document form can lead to impaired judgement decisions, increased stress, and helplessness.

Böhme et al. tested user response times to various dialog boxes on approximately 80,000 users [2]. The dialog boxes requested permission from the users to collect anonymous browsing information and indicated that users would not lose any functionality by declining. Results showed that users responded significantly faster and were more likely to participate when they set the button texts to "I Accept" and "I Decline". The authors concluded that EULAs have trained users to reflexively accept prompts that use similar wording to EULA dialogs.

The issues with EULAs and privacy notices have attracted research from both the security and HCI communities.

Kay, et al. [19] proposed a design for agreements where the text is augmented with typographic manipulation, pull-quotes and factoids, vignettes, and iconic symbols. They evaluated their modifications by measuring the time users spent reading the EULAs with and without the modifications and then quizzing each group on portions of the EULA content. The average time a user spent on the standard agreement was only 7 seconds, and the time spent on the textured agreements increased to 40 seconds. Not surprisingly, the quiz scores had a strong positive correlation with the time spent reading the EULA.

Good et al. [13], [14] conducted two experiments on measuring the effectiveness of providing additional short notices before or after the EULA. The notices summarize the contents of the EULA in short sentences without legal jargon. Users chose whether to install three different programs. In a post-experiment survey, users would be shown the short notice and ask if they would install the software again in order to measure regret. They found that the short notice shown before the EULA reduced the number of software installations and significantly reduced the regret for those that chose to install the software. With respect to measuring regret, Wang et al. measured the regret that users expressed after making posts on Facebook [31].

The P3P standard [30] was introduced for websites to present privacy policies in a standardized machine-readable format. Browsers or plugins could then present policies in a short consistent format to the end-user. However, P3P has suffered from low adoption rates, a fact acknowledged in the critiques section of the P3P site. Additionally, Hochheiser and others have criticized P3P as being a political standard intended to prevent legislation rather than a solution to the privacy paradox [16].

Kelley et al. used an iterative design process to create a standard privacy label that represents the privacy policy of a website similar to the nutrition labels on food products [20]. Data is extracted from the P3P XML file and organized into a tabular format where the rows indicate the types of data columns associate use. An evaluation of the interface with an online survey of over 700 users using Amazon's Mechanical Turk [21] found that users given a privacy label significantly outperformed the users limited to the full-text format.

Vila et al. modeled privacy policies as a lemons market and found that proposals like P3P are suffering from low adoption because the cost for a website that intends to honor its privacy statement is essentially the same as one that doesn't [29]. Therefore, without a global privacy policy enforcement, it's not worth investing in high-privacy policies for legitimate sites.

Boldt, et al. [3] proposed a privacy-based software classification scheme. They create a two-dimensional matrix where software is placed based on the consent required by the user, and the negative consequences from installing the software. For example, legitimate software would have high consent and negligible negative consequences, where trojans and parasites would have low consent and severe negative consequences. Users then set privacy preferences on their computer, which are enforced via a machine-readable deed.

Bonneau et al. conducted a study on the market for privacy in social networking websites [4]. They analyzed 45 large-scale social networking websites by comparing the following aspects: the data the sites collected, the control the users have over their privacy, the accessibility of the privacy policy, an overall privacy score, and a site functionality score. They then used game theory to model the results and showed that improving privacy accessibility doesn't help a site's growth. They also find that, as the site expands, privacy can then be advertised to capture the privacy conscious segment of the market.

Simultaneously, users often make security decisions based on prior peer behaviors [12] Arguably, then an application's perceived popularity should impact installation decisions more than EULAs and privacy policies.

In summary, the free-form EULAs and privacy policies in use today are ineffective at signaling the privacy implications of installing a piece of software (or using a website) to the majority of users.

B. Permission Based Applications

Despite the failure of privacy policies and EULAs for traditional software, the application permissions model offers a promising new approach. The default permissions completely sandbox the application so it cannot read any of the user's data, access device peripherals (e.g. GPS), or send information over the network. In order to perform any of these tasks, the application declares upon installation its intent to access these items. The platform then prompts the user to grant or deny the permissions.

The approach makes vendor misrepresentation more difficult. Applications will be blocked from accessing data without stated request (barring vulnerabilities in the operating system). For example, a user can be confident that if an application does not request access to his/her GPS location, it will not have access to that information.

An important thing to note is that these permissions requests are not a replacement for a privacy policy. A

user can only know to what information an application has access, not its use or reuse. The Facebook permissions request offers a link to the vendor's privacy policy and terms of service (with the inherent limits of privacy policies). The Android interface does not offer link to a privacy policies.

Enck et al. created an Android application decompiler and analyzed the code of 1,100 apps to determine how they handled data [8]. They found that many of the applications relayed sensitive information back to advertisement servers or the vendors' servers. For example, many would use the *phone state* permission (which allows access to the phone serial numbers and identifiers) in conjunction with the *Internet* permission to create a unique profile on a remote server for that phone. Most of these leaks offered no gain in functionality for the user.

Tam et al. analyzed various design decisions for permissions requests on Facebook [27]. Participants were shown varying permissions requests and quizzed about the application's permissions. They found that grouping permissions by action (e.g. can read X, Y, and Z; can write Y and Z) appeared to be the most significant change to improve understanding and retention. Interestingly, they found that including pictographic representations instead of just text didn't appear to have any helpful effect.

King et al. ran a survey on Facebook about the users' privacy perception of 3rd-party apps [22]. They found that Facebook users were unaware of an application's data access even after viewing the permissions screen. Surprisingly, they found that the only significant demographic predictor of privacy awareness was whether or not the user had previously experienced an adverse privacy event on Facebook before. They concluded that Facebook needs a better warning system to indicate that the Apps are not part of Facebook. The integration with the website suggests that Facebook approved of the applications.

Felt et al. and Vidas et al. both published research on another problem that arises with the permissions model: developers requesting permissions they don't need [9], [28]. They found that over a third of the apps on the market at the time of their analysis were over-privileged. This is a major concern, not only immediately, but also because this "permissions creep" may train users to routinely and constantly allow the requests to excessive access.

Felt et al. conducted an analysis of the current permissions models present in both Google Chrome and Android [10]. They conclude that the permission model can significantly reduce risks to the end-user. Such an outcome depends on developers not over-requesting permissions and user care with permissions requests. One concern is that permissions are uniformly presented with no risk information or framing. Similarly, Barrera et al. [1] mapped out the permissions used by 1,100 apps on the market. They determined that the current granularity for permissions was too coarse to indicate many privacy-relevant distinctions. For example, Internet would just be used to retrieve ads, not to transmit contact data.

Doty et al. surveyed the methods that various computing platforms use to request permission from the user to reveal location information [6]. They suggest that variations in the interfaces and the lack of information about the purpose of data access (e.g. location) results in uninformed decision making. For example, if a user is accustomed to a platform that prompts every time location data is needed, he/she may mistakenly grant full-time location access to an application on another platform when prompted. The authors suggest a fix similar to the P3P protocol where users configure their location privacy preferences in a machine-readable format that all of the location aware platforms (e.g. browsers, cameras, phones) will interpret and honor. Their research illustrates the problems that arise from a lack of standards in privacy controls.

Previous research has shown that users are willing to install full applications on their computer for very little money (less than a U.S. dollar), even when aware of the privacy risks [5]. Some projects have tried to address this by allowing users to install the app and have their privacy too. Enck et al. introduced *TaintDroid* [7], which dynamically tracks references to variables that contain sensitive information. Taint Droid determines what applications do with sensitive information once it is referenced (e.g. is it sent over the network?). This tool is helpful for analysis; however, it currently doesn't allow users to block data transmission. However, reporting can be an effective mode of control in itself [23].

Nauman et al. proposed a modification to the permissions framework, *Apex* [24], where a user can tap on any permission that an application requests and deny the application access to that information without having to abort the installation. The user can also selectively grant permissions based on the time of day or limit the number of times an app can use a specific permission.

Hornyack et al. introduced a similar solution called *AppFence* [17], where any requests to sensitive information by the application are fulfilled by false or empty data. They tested their solution on many applications and found that in the majority of cases no functionality was removed, only behavioral advertising was affected. However, apps that legitimately used user location data for functionality (e.g. restaurant locators, etc.) were impacted.

Gilbert et al. proposed an automated security scanning process for all apps submitted to the market called AppInspector [11]. It would analyze the app to determine data use as well as access and automatically generate a security report containing all of the potential privacy and security risks. Users would then be able to check these reports to make better informed decisions.

III. Experimental Design

We measured the effectiveness of the permissions requests using a method similar to the one used by Good et al. [13], [14] when they tested custom notices for EULAs. Half of the users experienced the normal install and the other half viewed an additional custom notice explaining the specific permissions requested by the application. After completing

the experiment, both groups were shown the same custom notice and then were asked if they would like to uninstall the app. If the normal permissions were effective, neither group would opt to uninstall an app because they would already be aware of what the permissions allowed. When participants ignore both the normal notice and our custom notice, both groups would have members that opt to uninstall apps (i.e. express regret).

Our study consisted of an experimental portion followed by a survey. The survey had two parts; one based on decisions made during the experiment, and one that stayed the same for all participants.

A. Testing Individual Hypothesis

Hypothesis 1: The majority of users do not know the function of common Android permissions.

A short quiz in the survey presented the participant with a permission. Then a multiple-choice selection of possible behaviors was included along with an option to explicit *I don't know* answer.

Hypothesis 2: An additional text explanation of the permissions will improve users' understanding of application permissions.

Half of the participants were shown an extended permissions warning while the other half only saw the normal interface. We then compared the install rates, quiz results, and the rates of regret between the two groups. Regret was measured by informing the participants what the applications could do with the requested permissions and then asking if they would then like to uninstall the application.

Hypothesis 3: Excessive permissions requested by applications has no impact on its install rates.

We included two applications that offered the same functionality, yet vastly different requested permissions. One requested the minimum required for the described functionality, while the other requested many it would never need for its advertised functionality.

Hypothesis 4: An application's perceived popularity has a impact on its install rates.

We divided participants into two groups with different download counts for the same apps. One group received apps with a mix of high and low download counts. The other group of participants received the same app groups, but with the download counts swapped. For example, in group 1, *App1* has 100 downloads and *App2* has 100,000 downloads; however, in group 2, the reverse is true.

B. Survey Instrument

The experimental portion of the survey was designed to mimic the process of installing Android applications in a regular browser. We constructed a JavaScript/HTML application to simulate a portion of the Android Market, which allowed us to modify aspects of the install process to test our modifications. Participants were presented with a list of nine applications and were instructed to install the applications they would choose for an Android phone that contained no other applications. The participants had to at least view the description before being allowed to continue through the survey.

To avoid the bias observed in [13] caused by participants recognizing familiar applications they had previous experience with, all nine applications were generic fake applications (e.g. 'Weather Info', 'News', and 'Sports Scores Tracker').

Upon starting the experiment, the participants were placed into one of two permissions 'notice' groups. One received the regular permissions prompt and the other received an additional permissions explanation afterwards.[1]

To measure the effect of the download count, participants were also divided into two download count groups. With one group, half of the apps had a download count of "less than 100" and the other half had a high download count of "500,000+". The other group had the opposite counts. This allowed us to measure the impact of the download count on an individual application basis.

After completing the experimental portion, the participants proceeded to a two-part survey. The first part was based on the installation decisions. For each application the participant installed, the extended permissions explanation was shown with two questions asking if the participant was aware the app had those capabilities and if the he/she would uninstall the application at that point. For each application not installed, participants were asked for a reason via a multiple choice selection with an optional free-form "other" field. These questions allowed us to measure the impact of changed environmental variables on participants' installation decisions.

The second part of the survey consisted of some basic Android usage questions and four multiple-choice quiz questions. Each quiz question tested participant knowledge of permission functionality.

C. Participants

The experiment was run on Amazon's Mechanical Turk with 200 participants. The advertisement asked participants to take part in a study about the Android Market while avoiding any indications that it was a privacy-related survey to avoid biases caused by priming the participants and attracting privacy-interested people. It also stated that it was only intended for participants with Android phones. Since the participants could not be observed, they were required to visit a web page with their Android device to get an access code required to start the survey. They were also required to have a 95% job approval rate to prevent known spammers from participating.

IV. Results

The survey started with 200 participants. Nine participants' reposes were rejected because they did not correctly read the directions and chose not to install apps because their existing phones "had similar apps already". Six more participants were removed because they had difficulty getting the simulator to work, leaving a total of 185 participants.

[1]Example of extended warning shown in figure 1.

A two-tailed Fisher's exact test was used for testing the significance of variables between categories.

A. Impact of Application Download Count

To measure the impact of the download count on install decision, we compared the counts of the apps not installed between the two download groups. To get the number of applications not installed because of the download count, we counted the number of *The app didn't seem popular* responses, as there were no other 'popularity' signals such as reviews, stars, or brand names.

We compared the installation cancellations caused by lack of perceived popularity between the groups that received a high vs. low 'download counter' for the same app. The result was statistically significant ($P<0.05$) on three of the apps, indicating that users consider download counts when deciding to install an app. Only two of the apps had an alternative offering the same functionality, suggesting that users may forgo functionality due to a perceived unpopularity.

B. Normal Prompt vs. Extended Warning

We measured the impact of the extended permissions warning from two perspectives: the effect it had on the number apps not installed due to permissions concerns, and the effect it had on the number of requests to uninstall apps.

Post-experiment, all participants were shown the extended permissions warning for each app installed. They were then asked if they were aware of the conditions and if they would uninstall the app at that point. When analyzing from this perspective of uninstalls caused by regret, the extended permissions did not have a statistically significant effect for any of the applications.

When analyzing from the perspective of refusals to install due to permissions concerns, the extended permissions did not have a statistically significant effect for any of the applications.

This is interesting because the users that received the extended notice during the install received the exact same notice after the install. They then indicated that they were unaware of its contents beforehand. This appears to corroborate the findings from previous research which showed that users blindly click through text dialogs during an install process [2].

C. Impact of Permissions Requested

In order to analyze the impact of varying the requested permissions, we checked the reasons the two weather applications were not installed, which had the same description/functionality but required different permissions. One needed just Internet access and location data, while the other additionally asked for access to SMS messages, contact data, phone calls, and SD card. However, the differences were not statistically significant.

D. Participants' Knowledge of Permissions

The four quiz questions asked participants what a given permission would allow an application to do. In addition to two or three regular answers, participants were also allowed to select "I don't know" or "None of the above".

Question	Correct	Incorrect	Don't Know
Read Phone State and Identity	48.6%	28.2%	23.2%
Modify/delete SD card contents	56.8%	37.3%	5.9%
Fine (GPS) Location	64.9%	27.0%	8.1%
Read Contact Data and *Full Internet Access*	61.1%	25.9%	13.0%
Total	**57.8%**	**28.4%**	**13.8%**

Table I: Results for Permissions Quiz

Overall, the participants correctly identified what a permission could do 57.8% of the time. Table I shows the results for each question. The effect of the extended permissions warning on quiz scores was statistically insignificant ($P>0.24$). However, by the time the quiz was taken, both groups had seen the extended warnings at least once, so this result is not surprising.

E. Viewing Permission Details

The simulator also monitored user clicks on permissions during the installation process to reveal their details. Only 14 participants (7.5%) clicked on a permission at any point during the simulation to get details, and only 8 participants (4.3%) viewed more than one permission's details. The extended permissions warning had no statistically significant influence on whether or not a participant viewed a permission's details ($P>0.2764$).

V. Discussions

The additional warning did not correlate with a higher awareness of the app data access capabilities. There could be several factors contributing to this, especially the placement of the notice. Additionally, our extended warning was text-only. As a result of its failure, we introduced the following hypothesis and tested it with a second experiment.[2]

Hypothesis 5: An additional visual warning is more effective than a text explanation alone.

This was measured in the second experiment by dividing the participants into four groups defined as follows. *CORRECT* received visual warnings on apps with risky permissions; *INVERSE* received visual warnings on apps without risky permissions; *ALL* received them on all apps; and *NONE* did not get any visual warnings. For the visual warning, we used pictures of cartoon eyes[3] that were shown to be effective at indicating privacy exposure information to users in previous research on location sharing [25]. The instructions for the participants stated that the appearance of eyes indicated that the permissions allow an app to access potentially sensitive personal information.

We compared the *ALL* group against the *NONE* group and the ratio of applications not installed due to permissions increased significantly ($P < 0.02$) in the *ALL* group. To verify that the eyes did not just increase risk aversion throughout the simulation, we compared the not-installed

[2]The second experiment was also run on Mechanical Turk and resulted in 177 usable responses.

[3]Figure 1 shows an example of a permissions screen that shows the visual warning.

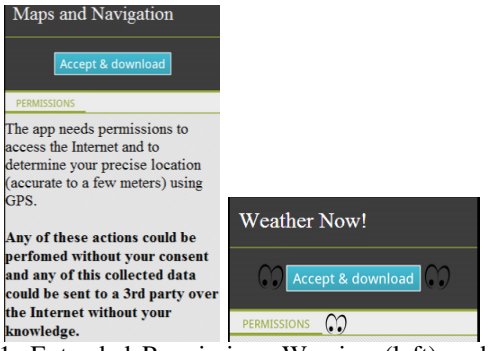

Figure 1: Extended Permissions Warning (left) and Visual Warning on Permissions Screen (right)

ratios of the apps between the *NONE* and *INVERSE* groups. There was an impact on the safe apps ($P < 0.02$) and no difference on the risky apps, indicating that the eyes appearing on some apps did not impact the installations of apps without eyes. Unexpectedly, the ratios for the apps that had eyes in the *CORRECT* group were the same as the *NONE* group, indicating no apparent impact of the eyes on that group.

VI. Conclusions

In this paper, we examined the efficacy of privacy signaling provided by Android permissions requests. Based on the high rate of participants admitting they were unaware of the implications of the requested permissions and their expressed intent to uninstall apps after learning about the permissions, permissions requests appear to be ineffective. The download count from an app had a much higher impact on user install decisions than any changes to the permissions.

To improve the permissions prompt, we also introduced a short notice, shown after the standard permissions prompt which explained the combination of permissions. Initially, it did not have a statistically significant effect on the installation count or the post-installation awareness/regret. However, we introduced an additional aggregating, non-technical visual warning for risky permissions in a second experiment that proved to be much more effective than the text-warning alone.

In light of these results, the Android Market interface may need to be modified to stress permissions or make them easier to understand for users. In their current state, they appear to be limited as a tool for only privacy conscious users. However, the results of the *CORRECT* group when considered with the *INVERSE*, *ALL* and *NONE* prevents strong advocacy for any marketplace without additional, repeated experimental validation.

Acknowledgments

This material is based upon work supported, in part, by the National Science Foundation under Grant CNS 1250367 and DHS under Contract BAA 11-02-TTA 03-0107. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF, DHS, or Indiana University.

References

[1] D. Barrera, H. G. u. c. Kayacik, P. C. van Oorschot, and A. Somayaji. A methodology for empirical analysis of permission-based security models and its application to Android. In *Computer and Communications Security*, pages 73–84, 2010.

[2] R. Böhme and S. Köpsell. Trained to accept?: A field experiment on consent dialogs. In *CHI*, pages 2403–2406, 2010.

[3] M. Boldt and B. Carlsson. Privacy-invasive software and preventive mechanisms. In *PROC of ICSNC*, 2006.

[4] J. Bonneau and S. Preibusch. The privacy jungle: On the market for data protection in social networks. In *WEIS*, 2009.

[5] N. Christin, S. Egelman, T. Vidas, and J. Grossklags. It's all about the benjamins: An empirical study on incentivizing users to ignore security advice. In *Financial Crypto*, 2011.

[6] N. Doty and E. Wilde. Geolocation privacy and application platforms. In *PROC of ACM SIGSPATIAL Workshop on Security and Privacy in GIS and LBS*, pages 65–69, 2010.

[7] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones. In *PROC of the USENIX conference on OSDI*, pages 1–6, 2010.

[8] W. Enck, D. Octeau, P. McDaniel, and S. Chaudhuri. A study of android application security. In *PROC of the USENIX conference on Security*, pages 21–21, 2011.

[9] A. P. Felt, E. Chin, S. Hanna, D. Song, and D. Wagner. Android permissions demystified. In *Computer and Communications Security*, pages 627–638, 2011.

[10] A. P. Felt, K. Greenwood, and D. Wagner. The effectiveness of application permissions. In *PROC of the 2nd USENIX conference on Web application development*, WebApps'11, 2011.

[11] P. Gilbert, B.-G. Chun, L. P. Cox, and J. Jung. Vision: automated security validation of mobile apps at app markets. In *PROC of MCS*, pages 21–26, 2011.

[12] J. Goecks, W. K. Edwards, and E. D. Mynatt. Challenges in supporting end-user privacy and security management with social navigation. In *PROC of SOUPS*, pages 5:1–5:12, 2009.

[13] N. Good, R. Dhamija, J. Grossklags, D. Thaw, S. Aronowitz, D. Mulligan, and J. Konstan. Stopping spyware at the gate: a user study of privacy, notice and spyware. In *PROC of SOUPS*, SOUPS '05, 2005.

[14] N. S. Good, J. Grossklags, D. K. Mulligan, and J. A. Konstan. Noticing notice: a large-scale experiment on the timing of software license agreements. In *PROC of CHI*, CHI '07, pages 607–616, 2007.

[15] J. Grossklags and N. Good. Empirical studies on software notices to inform policy makers and usability designers. In *PROC of Financial cryptography and Usable Security*, pages 341–355, 2007.

[16] H. Hochheiser. The platform for privacy preference as a social protocol: An examination within the u.s. policy context. *ACM Trans. Internet Technol.*, 2(4):276–306, 2002.

[17] 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 *Computer and Communications Security*, pages 639–652, 2011.

[18] C. Jensen and C. Potts. Privacy policies as decision-making tools: an evaluation of online privacy notices. In *PROC of CHI*, pages 471–478, 2004.

[19] M. Kay and M. Terry. Textured agreements: re-envisioning electronic consent. In *PROC of SOUPS*, pages 13:1–13:13, 2010.

[20] P. G. Kelley, J. Bresee, L. F. Cranor, and R. W. Reeder. A "nutrition label" for privacy. In *PROC of the 5th Symposium on Usable Privacy and Security*, SOUPS '09, pages 4:1–4:12, 2009.

[21] P. G. Kelley, L. Cesca, J. Bresee, and L. F. Cranor. Standardizing privacy notices: an online study of the nutrition label approach. In *PROC of CHI*, pages 1573–1582, 2010.

[22] J. King, A. Lampinen, and A. Smolen. Privacy: Is there an app for that? November 2011.

[23] D. Mulligan. Information disclosure as a light-weight regulatory mechanism. In *DIMACS workshop on information security economics*, pages 18–19, 2007.

[24] M. Nauman, S. Khan, and X. Zhang. Apex: extending android permission model and enforcement with user-defined runtime constraints. In *Asia Computer and Communications Security*, pages 328–332, 2010.

[25] R. Schlegel, A. Kapadia, and A. J. Lee. Eyeing your exposure: Quantifying and controlling information sharing for improved privacy. In *SOUPS*, 2011.

[26] J. C. Sipior, B. T. Ward, and G. R. Roselli. A united states perspective on the ethical and legal issues of spyware. In *PROC of ICEC*, ICEC '05, pages 738–743, 2005.

[27] J. Tam, R. W. Reeder, and S. Schechter. I'm allowing what? disclosing the authority applications demand of users as a condition of installation. May 2010.

[28] T. Vidas, N. Christin, and L. Cranor. Curbing Android permission creep. In *PROC of W2SP*, 2011.

[29] T. Vila, R. Greenstadt, and D. Molnar. Why we can't be bothered to read privacy policies models of privacy economics as a lemons market. In *PROC of ICEC*, pages 403–407, 2003.

[30] W3C. Platform for privacy preferences (p3p) project, Dec. 2007.

[31] Y. Wang, G. Norcie, S. Komanduri, A. Acquisti, P. Leon, and L. Cranor. I regretted the minute i pressed share: A qualitative study of regrets on facebook. In *Proceedings of the Seventh Symposium on Usable Privacy and Security*, page 10. ACM, 2011.