Use of Ratings from Personalized Communities for Trustworthy Application Installation

Pern Hui Chia[1], Andreas P. Heiner[2], and N. Asokan[2]

[1] Q2S* NTNU, Trondheim, Norway
[2] Nokia Research Centre, Helsinki, Finland
chia@q2s.ntnu.no, {andreas.heiner,n.asokan}@nokia.com

Abstract. The problem of identifying inappropriate software is a daunting one for ordinary users. The two currently prevalent methods are intrinsically centralized: certification of "good" software by platform vendors and flagging of "bad" software by antivirus vendors or other global entities. However, because appropriateness has cultural and social dimensions, centralized means of signaling appropriateness is ineffective and can lead to habituation (user clicking-through warnings) or disputes (users discovering that certified software is inappropriate).

In this work, we look at the possibility of relying on inputs from personalized communities (consisting of friends and experts whom individual users trust) to avoid installing inappropriate software. Drawing from theories, we developed a set of design guidelines for a trustworthy application installation process. We had an initial validation of the guidelines through an online survey; we verified the high relevance of information from a personalized community and found strong user motivation to protect friends and family members when know of digital risks. We designed and implemented a prototype system on the Nokia N810 tablet. In addition to showing risk signals from personalized community prominently, our prototype installer deters unsafe actions by slowing the user down with habituation-breaking mechanisms. We conducted also a hands-on evaluation and verified the strength of opinion communicated through friends over opinion by online community members.

Keywords: Usable security, User-centered design, Risk signaling.

1 Introduction

The versatility of mobile devices paves the way for a large array of novel applications; mobile devices today contain ever more sensitive information such as medical data, user location and financial credentials. As device manufacturers open up the mobile

[*] Centre of Quantifiable Quality of Service in Communication Systems (Q2S), Centre of Excellence appointed by the Research Council of Norway, is funded by the Research Council, Norwegian Uni. of Science and Technology (NTNU) and UNINETT.
http://www.q2s.ntnu.no

T. Aura, K. Järvinen, and K. Nyberg (Eds.): NordSec 2010, LNCS 7127, pp. 71–88, 2012.

platforms to encourage third party software development, applications from different sources are becoming available. Some of these applications, although not malicious, are inappropriate in the sense that they can cause harm (e.g., loss of privacy) or offense (e.g., culturally or religiously-insensitive content) to some users. The appropriateness of FlexiSpy – one of several commercial applications intended to spy on the activities of the user of a mobile phone – has been contentious. Mobile applications with potentially inappropriate content are becoming publicly available[1].

The bar for developing "applications" is also being lowered drastically. One can now develop simple applications for mobile devices by using only scripting languages (e.g., using JavaScript+HTML+CSS for Palm webOS [27]), or even without much programming experience using online tools (e.g., OviAppWizard [28] and AppWizard [29]). These applications are unlikely to be malicious (as they don't do too much) but we can expect a flood of applications from a larger variety of originators which increases the chance of a given application offending a certain group of users.

1.1 What Is Inappropriate Software?

StopBadware.org [30] defines badware as software that fundamentally disregards a user's choice about how his or her computer or network connection will be used. In addition to software with malicious intent, the definition covers bad practices, such as installing additional unexpected software, hiding details from users, and incomprehensible End User License Agreement (EULA) that hinder an informed consent. Our understanding of "inappropriate software" is close to this notion of badware. In addition to maliciousness and disregard of user-choice, we consider software appropriateness to cover also the cultural and social dimensions.

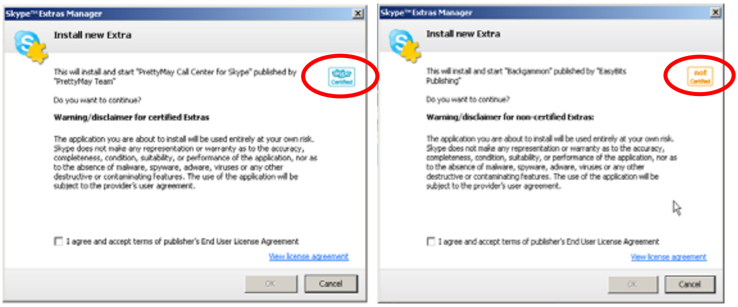
1.2 Software Certification and Its Limitations

A dominant approach for reducing the risk of malicious software on mobile platforms (e.g., Symbian, BlackBerry, J2ME and Android) is to rely on software certification and platform security. *Software certification* (e.g., Java Verified Program [31] and Symbian Signed [32]) is usually subject to software testing conducted by an authorized third party using publicly available criteria. But testing typically focuses only on technical compliance such as proper usage of system resources, proper application start/stop behavior and support for complete un-installation. *Platform security* (e.g., Symbian OS Platform Security [10] and Java Security Architecture [8]) refers to the isolation and access control features of the operating system or runtime. Ideally, software certification and platform security are used in tandem: an application is granted the privileges it requires if it is signed by a party trusted by the device platform. However, certification does not guarantee software security. It also does not consider the social and cultural aspects of software appropriateness.

Uncertified Software: The Risk of Habituation. Many application installers (in mobile or desktop environment) resort to displaying warning and disclaimer notices to

[1] A search using the keyword 'entertainment' in the iTunes Appstore returns a number of applications with potentially mature content.

signal risks when software to be installed is not certified. Visual difference when installing certified and non-certified software is often low; the text is also typically uninformative (see Figure 1). Providing system-generated notifications to which user attends to maintain security is the practice of "security by admonition" [26]. Besides degrading user experience, such notices lead to a high rate of false-positives causing many users to habitually click-through them. Click-through behavior is further entrenched when warnings equating "uncertified software" as possibly "harmful" may contradict other signals a user receives. An example of this is the installation of Gmail application (Figure 2a); the installer warns that it is 'untrusted' and 'maybe harmful' since it is not certified. A user, who trusts Google and who has just downloaded the application from Google's website will ignore and click-through the warning.



This will install and start "PrettyMay Call Center for Skype" published by "PrettyMay Team"

Do you want to continue?

Warning/disclaimer for certified Extras

The application you are about to install will be used entirely at your own risk. Skype does not make any representation or warranty as to the accuracy, completeness, condition, suitability, or performance of the application, nor as to the absence of malware, spyware, adware, viruses or any other destructive or contaminating features. The use of the application will be subject to the provider's user agreement.

□ I agree and accept terms of publisher's End User License Agreement

View license agreement

This will install and start "Backgammon" published by "EasyBits Publishing"

Do you want to continue?

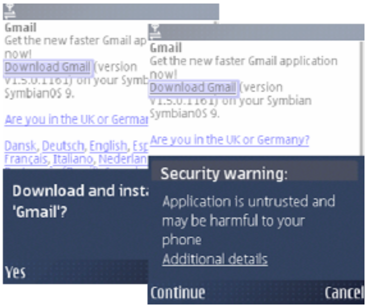
Warning/disclaimer for non-certified Extras:

The application you are about to install will be used entirely at your own risk. Skype does not make any representation or warranty as to the accuracy, completeness, condition, suitability, or performance of the application, nor as to the absence of malware, spyware, adware, viruses or any other destructive or contaminating features. The use of the application will be subject to the provider's user agreement.

□ I agree and accept terms of publisher's End User License Agreement

View license agreement

Fig. 1. The Skype PC version has a list of 'featured extras' that include both Skype-certified and non-certified plugins. The visual difference when installing the two types is only the color of certification label (light-blue vs. soft-yellow).



Gmail
Get the new faster Gmail ap
now!
Download Gmail (version
V1.5.0.1161) on your Symb
SymbianOS 9.

Are you in the UK or Germar

Dansk, Deutsch, English, Esp
Français, Italiano, Nederlan

Download and inst
'Gmail'?

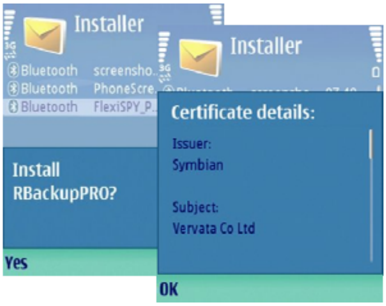
Gmail
Get the new faster Gmail application now!
Download Gmail (version V1.5.0.1161) on your Symbian SymbianOS 9.

Are you in the UK or Germany?

Security warning:

Application is untrusted and may be harmful to your phone

Additional details



Install RBackupPRO?

Certificate details:

Issuer:
Symbian

Subject:
Vervata Co Ltd

Fig. 2a. Gmail is not certified

Fig. 2b. FlexiSpy is certified [33]

Certified Software: The Risk of Centralized Judgment. On the other hand, software certified by a central authority may be perceived as inappropriate by some communities. An example of this is FlexiSpy – advertised as a tool to monitor the work force and protect the children and is available on most mobile platforms. The application has a number of characteristics that can be construed inappropriate: it spies on user activities (call, SMS, email, location), is invisible in the application list, uses a deceptive name (RBackupPro) and allows the device to be controlled remotely. F-Secure flagged it as spyware that may be used for malicious purposes illegally [33] but as FlexiSpy fulfills the certification criteria, it is Symbian certified. In other words, a user is given a warning (Figure 2a) when he tries to install Gmail although he may likely trust it, whereas FlexiSpy can be installed without warnings (Figure 2b) even though he may belong to the group of people who consider it inappropriate.

On iPhone, Apple decides which 3rd party applications can be distributed through the iTunes Appstore; we regard this as a scheme of implicit certification. Apple has also the means to activate a "kill-switch" to disable applications that may have been "inadvertently" distributed and later deemed "inappropriate by Apple". Apple's review criteria are, however, not publicly available. This has resulted in outcomes that are contested by developers and the Electronic Frontiers Foundation [34]. South Park, Eucalyptus and the Stern.de reader were among applications that were deemed "inappropriate by Apple" but later approved after protests [34]. Such contentions exemplify that centralized judgment can hardly cater for the value systems of different users.

1.3 Our Contribution

- We derived a set of design guidelines, grounded in cognitive and information flow theories, for a trustworthy software installation process (Section 2). Although we focus on mobile devices here in this paper, the guidelines are applicable to other platforms (e.g., desktop, Facebook) where installation by end-users can take place.
- We surveyed for the behaviors during installation, and we found high relevance of information from friends/family and user motivation to protect them. (Section 3)
- We built and evaluated a prototype system (Section 4 & 5). Although we could not test the efficacy of our prototype against habituation, we verified that opinion by friend is of higher impact than that of by online community through the user study.

2 Designing a Trustworthy Installation Process

We consider that *a trustworthy installation process* to be one that helps users to avoid installing inappropriate application. Besides providing risk signals that are perceived reliable and relevant, the installer should take into account of the risk of habituation, which undermines the efficacy of many security mechanisms involving end-users.

2.1 Cognition during Application Installation

In the conventional installation task flow, as a user defines his expectation or desired software functionality (for a task at hand), he starts by searching for an application in the application market or on the web that meets his requirements. When such an application is found (and downloaded), the user will have to perform some "post-selection" actions such as accepting security-related conditions and configuration options before he is able to use it (objective attained). These "post-selection" steps are nearly always made without the user paying attention to what is asked. Habituation to click-through this "post-selection" phase could be attributed to current design of installation that lacks understanding for user's cognition.

To develop guidelines that take into account of user's cognition, we draw on the *dual processing theory* [12] in cognitive science, which identifies two main types of cognitive processes: *controlled* and *automated* processes.

Controlled processes are goal-directed; a user defines an objective and plans a path that (in his opinion) will lead to the objective. At certain points, the user will make an appreciation of the current context in order to decide on the next best-move in achieving his end goal. This process is highly dynamic and requires logical thinking. For these reasons, one can execute only one controlled process at a time. Appreciation of the current context and decision for a course of action, over time, can be based on superficial comparison of contexts. This leads to faster decision making [7,12]. Despite a potential high degree of *automation in decision making*, it remains a controlled process as one will always have to compare between multiple contexts.

Automated processes such as habits, on the other hand, pose little to no cognitive load. *Habits* develop from deliberate into thoughtless actions towards a goal. If the context for an action is nearly identical over a series of performances, the action becomes mentally associated with the context; observing the context is enough to trigger the action [1,17]. The simpler a task, the more frequently it is executed and the higher similarity in context, the stronger a habit can become. New information that invalidates the initial conditions (which led to an action or habit) will go unnoticed.

The difference between *habits* (automated) and *automation in decision making* (controlled) lies in the constancy of the context. Habits are developed if the context is (nearly) always the same. With the latter, context varies between a number of states with reasonable likelihood, thus requiring a controlled process of context comparison.

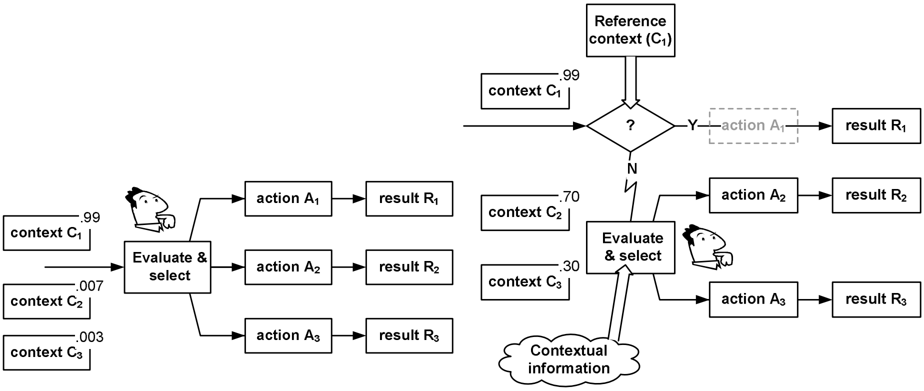


Fig. 3. (Left) A constant context results in habitual behavior. (Right) Using the attention capture process with the dominant context as reference prevents this.

The constant context (lack of context-sensitive information) during installation makes the action of confirmation a habit. This is exemplified in Figure 3 (left); the context of a normal installation flow (C1) demands the decision of action A1 (install) that results in R1. An abnormal context (C2) should lead to R2 (installation aborted). But as context C1 occurs much more often (denoted with probability .99) than context C2, user will over time expect context C1 and habitually selects action A1. This is more likely if there is no clear visual difference between the contexts (e.g., Figure 1). Furthermore, from a user perspective, the choice (install or abort) is asked after the last conscious step of *having decided* to download and install a particular application. Users also rarely face immediate consequence for installing inappropriate software.

We argue that habituation can be avoided by eliminating the need for user action in the normal and frequent context (an easy target of habituation) altogether. Depicted in Figure 3 (right), context C1 can be taken as reference context with an implied action A1. User can then be made aware of the deviation from this reference context through *attention capture* – the process of making a user aware of a change in environment that (may) require the user to attend to a new task [15]. A predominant view is that attention capture is an automated, stimulus-driven process modulated by the current controlled task [18]. The cognitive load required for the current task, as well as the strength and the relevance of the stimulus to the current task, affect the likelihood that a person will act on the stimulus. Thus, in addition to *visual salience*, the *relevance* and *strength of a warning (risk signal)* are paramount to ensure that users will take note of and evaluate the warning, during the installation process.

2.2 Information Flow and Risk Signaling

Software warnings (risk signals) have conventionally been communicated to users in a hypodermic-needle manner by expert entities (e.g., antivirus vendors). These risk signals are designed against malware and do not cover for aspects such as the respect for user choice and the social/cultural factors of software appropriateness.

In search of risk signals that are relevant and of high impact, we refer to the *two-step flow theory* [13] – the founding work of innovation diffusion theory – which describes how communication can be more effective through a network of people (rather than through the hypodermic-needle fashion). Central to the theory are the information brokers (originally known as opinion leaders in [13]) who are not necessarily the most knowledgeable but are nevertheless skillful in interconnecting people [3]. Information brokers guide the information flow from sources into separate groups (first step) given incentives such as early information access and social capital [3]. When information gets into a particular group, competition among group members can serve to encourage each other to improve own knowledge and exchange opinions, which constitutes the second step of information flow [3]. Social media such as Twitter and Facebook are successful examples that have harnessed the power of social networks for effective communication. Use of social networks for provisioning or relaying of risk signals is, however, still an early concept.

PhishTank [35] and Web of Trust (WOT) [35] are systems that employ “wisdom of crowds” (using a global community, *not* personalized network) to improve web security. PhishTank solicits reports and votes against phish-sites, while WOT collects public opinions on the trustworthiness, vendor-reliability, child-safety and privacy-handling of websites. Both systems aggregate user ratings into global (rather than personalized) values. Such global values can, however, be susceptible to exploitation. Moore and Clayton [16] argued that as participation in PhishTank follows a power-law distribution, its results can be easily influenced by the few highly active users[2].

Prior work has pointed to the advantages of using inputs from personalized networks instead of the global community. Against phishing, Camp advocated for the use of social networks to generate risk signals that are trustworthy as the incentive to cheat is low among members who share (long-term) social ties [4]. Inputs from social networks can also be verified through offline relationship, allowing incompetent or dishonest sources to be removed [4]. Personified risks are also perceived greater than anonymous risks [22]; this may help to mitigate the psychological bias (known as valence effect) in which people overestimate favorable events for themselves. Inputs from social networks are also socially and culturally relevant.

2.3 Design Guidelines

To sum up, we consider that a trustworthy installation process should:

- **Avoid requiring user actions that can be easily habituated.** User actions in a normal and frequent context could be made implicit and complemented with an attention capture mechanism to signal any deviation from this context.
- **Employ signals that are visually salient, relevant and of high impact.** Signals should cover both the objective and subjective factors of software appropriateness.
- **Incorporate mechanisms to gather and utilize feedbacks from user's personalized community.** In this work, we refer a *personalized community* to *friends* and *experts* whom individual users trust in providing valuable inputs about software appropriateness. *Experts* could be vendors or gurus who are knowledgeable in the technical evaluation of software. A list of reputable experts can be set for all users by default. Meanwhile, *Friends* refer to ones whom users have personal contacts with and whom could help by sharing personal experience about applications or relaying information. Here, we hypothesize that risk signals from the personalized community can be more effective (due to their relevance and trustworthiness) than that of from global community. We verified the relevance and strength of inputs from friends in our survey (Section 3) and user study (Section 5).

3 Web-Based Survey

We conducted an online survey to identify the installation behaviors and to evaluate the potentials of a personalized community in providing relevant and helpful signals.

[2] We note that this may be not too serious as determining whether a website is a phishing site (similar to whether an application is malicious) is usually objective. But judging if a website is trustworthy (with WOT, similar to evaluating the subjective factors of software appropriateness) can be contentious and prone to dishonest behavior (e.g., Sybil attack [5]).

Recruitment and Demographics. We recruited our participants mainly from universities. We put up posters around popular campus areas. Emails were also sent to colleagues in other universities with the request to take part and to the forward the invitation to their contacts. Throughout the recruitment and responding process, we referred our survey as a study on user behaviors during installation using the title: "A Survey on Software Installation". Considerations were taken to avoid priming of secure behaviors. The reward for participation was to receive a cinema ticket on a lucky draw basis. Winners who do not reside in the Nordic region were rewarded with a souvenir-book. The lucky draw was made a few weeks after the data collection.

The survey was open for participation for 3 weeks. In total, 120 participants took part in the survey. Participants who did not complete all questions, or whose total response time was unrealistically low (<10 minutes) were excluded. The final population consists of 106 subjects (36% females). 12% have a PhD degree, 42% have a Master degree while 28% have a Bachelor degree. 61% have a background in IT or engineering (power, electrical, mechanical, etc.) while 39% have a non-technical background (see Table 1). Subjects took 15 minutes on average to complete the survey, which was structured into 12 questions with 105 items in total. We mostly used a 4-point Likert scale on the perceived importance of an element and the likelihood or frequency of performing an action.

Table 1. Demographics of survey participants

Education/work background	
IT or Engineering	61%
Business / Finance	12%
Science / Math	8%
Arts & Social Science	10%
Others	9%

Age group	
18-24	15%
25-29	41%
30-39	32%
40-49	11%
50+	1%

Table 2. When know of digital risks

User would always / often inform	
friends or family	62 %
members of online community	15 %
expert individuals	14 %
expert organizations	8 %
antivirus software company	6 %

Results. We present a few interesting findings that we obtained. Finding-1 concerns the behaviors during installation while the others demonstrate the potentials of ratings from a personalized community. The percentage values were computed after reducing the responses from 4-point Likert scales into nominal levels of important/not, likely/not, or usually/seldom.

i. **Information during installation is mostly ignored.** 83%, 90% and 75% of the subjects reported that they seldom read the EULA, privacy policy and disclaimer notices respectively during the installation process. Similarly, 78% of the subjects seldom check for digital signatures (or software certificates), nor abort installation when they are absent. Only 30% usually abort installation given warnings from the installer. However, 69% usually abort installation if unnecessary personal questions were asked. 76% usually abort installation if warned by antivirus software, while 53% usually abort installation in the presence of advertisement pop-ups.

ii. **Security vendors, experts and friends are important sources for information on digital risks.** About 90% of the subjects reported that antivirus software is an important source of information about digital risks (e.g., harmful or inappropriate software/services). Expert organizations and individuals also scored high (75%). Undeniably, security vendors and experts are the most important sources of information on digital risks. The survey gave further interesting results. 65% of the subjects regarded the first-hand experience by friends and family members as important. In comparison, fewer subjects (50%) considered the experience from members of an online community to be important. This difference was statistically significant (p<.01, Chi-square). This suggests that users regard inputs from friends and family members to be more relevant than that of from an online community.

iii. **When users know about digital risks, they are motivated to inform friends or family rather than the online community.** 60% reported that they could usually find security-related information by themselves. However, only 34% have been asked by friends or family members on whether software is trustworthy or appropriate. This could be due to the lack of existing system to share their opinions about software with his friends or family members. Indeed, we find that motivation to inform friends or family members about digital risks is high. 62% of the subjects would inform them about digital risks. Comparatively, only 15% were motivated to inform the online community (see Table 2). The difference was statistically significant (p<.0001). This suggests that users have more motivation to protect his friends than members of online community. This supports the feasibility of a rating system based on personalized communities over the global-community compatriot.

iv. **Users consider reviews from trusted sources to be helpful.** With considerations to the limited screen size of mobile devices, 80% regarded reviews from trusted sources to be important/helpful information during software installation.

Limitation and Discussion. We note that the education level of the participants was high, and 61% of the subjects have a background in IT or engineering. Yet, even though we might expect the subjects to be more aware of digital risks, there is an evident 'click-through' behavior. Excluding those with an IT/Engineering background, slightly fewer subjects (51%) could usually find security-related information themselves. However, the key results remain unchanged: 66% regarded friends as important source of risk information; 60% would inform friends or family when know about digital risks (compared to only 12% would inform such risks to an online community); 72% perceived reviews and ratings from trusted sources to be important/helpful information during software installation.

4 System Architecture and Prototype

Two important components in our architecture are: (i) *software repository*, which maintains a list of applications available for installation and a software catalog (containing metadata such as price, author, description and keywords); (ii) *rendezvous server*, which issues identity certificates and manages the user database, social graph and application reviews. To use the prototype installer (developed on the Nokia N810 tablet), a user must first register and obtain his credentials at the rendezvous server. Thereafter, the user can add friends and experts whom he trusts into his personalized community, and share software reviews with them, using the prototype. Sharing is done through the rendezvous server, Bluetooth or email. Software reviews are digitally signed and verified on the prototype to ensure authenticity and integrity.

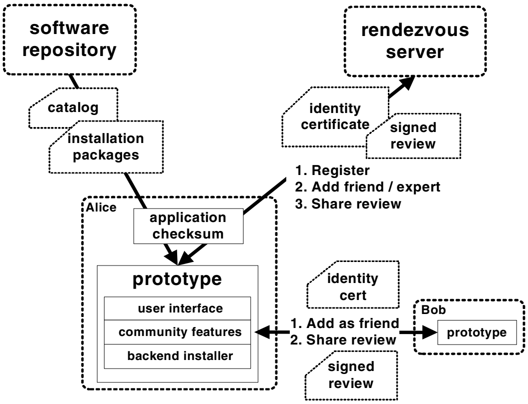
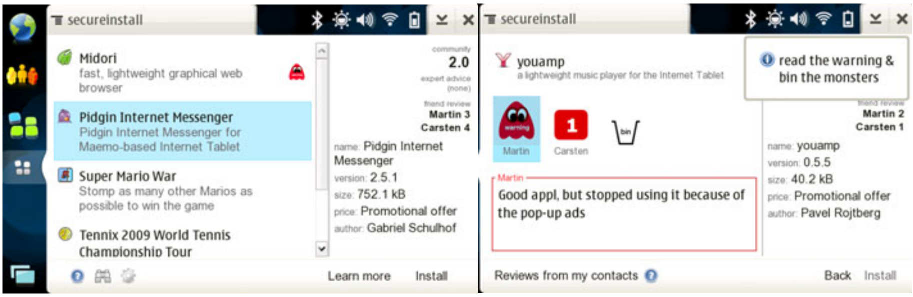


Fig. 4. System Architecture. The prototype was implemented on the Nokia N810 tablet, while a rendezvous server was setup on an Ubuntu desktop. The prototype interacts with conventional software repositories to obtain application catalog and installation packages.


secureinstall

Midori
fast, lightweight graphical web browser

Pidgin Internet Messenger
Pidgin Internet Messenger for Maemo-based Internet Tablet

Super Mario War
Stomp as many other Marios as possible to win the game

Tennix 2009 World Tennis Championship Tour

community
2.0
expert advice
(none)
friend review
Martin 3
Carsten 4

name: Pidgin Internet Messenger
version: 2.5.1
size: 752.1 kB
price: Promotional offer
author: Gabriel Schulhof

Learn more Install

secureinstall

youamp
a lightweight music player for the Internet Tablet

read the warning & bin the monsters

warning
Martin
1
Carsten
bin

friend review
Martin 2
Carsten 1

Martin
Good appl, but stopped using it because of the pop-up ads

name: youamp
version: 0.5.5
size: 40.2 kB
price: Promotional offer
author: Pavel Rojtberg

Reviews from my contacts Back Install

Fig. 5. Prototype. (Left) The front-page shows an application list with basic description on the right panel. (Right) The experimental 'bin-the-monster' mechanism: user clicks on a monster to read the negative review; he has to drag it into the bin if he chooses to disregard the review. (Note that the reviews and ratings were artificially generated for evaluation purposes only)

The installation task flow was redesigned. When a user defines his requirements and searches for suitable applications (using some keywords), our prototype displays a list of related software (Figure 5, left). The right panel shows basic information of a selected application, while detailed reviews from user's personalized community can be accessed by clicking on the "learn more" button. The "install" button will install an application without further prompting (if it has not been 'flagged' as potentially inappropriate by the user's personalized community). This removes user actions (in the post-selection phase of conventional task flow) that are prone to habituation.

For an application that has received *negative reviews* (i.e. flagged by the personalized community), a risk signal is shown prominently to catch the user's attention. To reflect the personal/social dimension of the warning, we chose a non-conventional risk symbol: a Pacman-like monster. Warning triangles and stop signs may signal that it is an "objective" opinion by some authorities.

The symbol is shown for flagged applications only; salience is increased by not showing positive cues. It is placed at the same level as the application name, and is enlarged when the application is highlighted. If the user decides to install an application that has been flagged, he is redirected to the review-page (Figure 5, right) where he has to read the detailed reviews. Textual review improves the relevance of a risk signal as user can appreciate what is said better than numerical values [20]. Negative reviews are framed in red (bottom-up salience). To mitigate a potential click-through when attending to the negative reviews, we experimented with two habituation-breaking tasks (to improve the efficacy of attention-capture):

- Delay: User has to read every negative review by clicking on each of the monsters with some time interval. When clicked, a monster will disappear into an icon with numerical rating only after a few seconds, before the next review can be read.
- Bin-the-monster: As before, but the monster only disappears when it is dragged into the bin. User cannot install until all monsters have been binned (Figure 5).

5 User Evaluation

We conducted a hands-on evaluation and investigated the strength of opinion given by friends compared to opinion given by online community members.

Recruitment and Demographics. Participants were mainly recruited from universities. We distributed recruitment notes around popular campus areas especially in the social science and science/math faculties. A web-form was also created to allow subjects to sign-up online. Participants of our survey were especially encouraged to take part if they reside in the Nordics; they were directed to the signup form upon completing the survey. Each participant was rewarded with two cinema tickets. There were in total 20 participants (7 females) consisting of students, researchers and a few working adults. 6 participants came from an IT background. The remaining subjects comprised of 6 mechanical, electrical or power engineering students, 4 science/math graduates, 3 art/design graduates, and 1 psychology undergraduate.

Experimental Setting. We specified 4 testing days and arranged with the participants a suitable session of an hour each. Individual participants were invited to our premises where the study took place. Each session was preceded with a brief interview. The main task was structured into four evaluation scenarios. In the end, we asked for the overall experience with our prototype before a final debrief.

In the brief interview, we asked if a subject has encountered situations where he had difficulties or doubts in determining the appropriateness of certain software; all subjects responded that they had been in such situations before. We then requested the subject to write down the names of two friends whose opinions could be useful in these situations. We then keyed in these two names into our prototype system.

We gave the subject a script containing the description of the initial setting and the four evaluation scenarios (denoted as S1, S2, S3 and S4). The initial setting depicts a situation where there was a special offer on 4 applications which the subject would have to decide if he would like to buy and install. The special offer was meant to provide motivation to buy/install the applications in the evaluation scenarios. Two games, a browser and a media player (denoted as A1, A2, A3 and A4) were selected such that the likelihood of subjects having prior experience with them was low.

Having understood the initial setting, the subject was required to decide if he would buy/install a specific application in each evaluation scenario based on some basic description (application name, file size, name of developer, a short text provided by developer) and software reviews provided by online community members as well as the two friends mentioned during the brief interview.

Four negative reviews were scripted to signal a mild level of inappropriateness. They concerned advertisement pop-ups, pornographic content, program crashes (data loss) and suspicious elements. A set of positive reviews were also scripted. Each application was associated with a fixed pair of negative and positive reviews.

The evaluation scenarios were designed to present to the subject, positive and negative reviews from either friends or online community, as described in Table 3 (left). We assigned the applications (A1, A2, A3 and A4) to the four scenarios in a rotating manner. Specifically, subject-1 would decide whether to buy/install applications A1, A2, A3 and A4, while subject-2 go through applications A2, A3, A4 and A1 in the fixed order of scenarios (from S1 to S2, S3 and finally S4). Rotating the applications in this manner avoided the potential bias due to the characteristics of individual applications and their fixed pair of positive/negative reviews.

Table 3. (Left) The 4 evaluation scenarios. (Right) Installation ratio in each scenario

		Install	Didn't Install
S1	No reviews from online community nor friends were provided	13	7
S2	Negative reviews were given by online community but friends gave positive reviews	10	10
S3	Positive reviews were given by online community but friends gave negative reviews	4	16
S4	Same as S3; the "bin-the-monster" mechanism was activated. After noting down the installation decision, subject was required to try installing the application (regardless of his decision) to experience the habituation-breaking interaction.	7	13

The subject was required to write down his decision to whether buy/install in each scenario and the reason on the evaluation script. In scenario S3, we asked for feedbacks on the use a Pacman-like monster as risk symbol. In scenario S4, we asked for experience with the "bin-the-monster" habituation-breaking mechanism. We used a 5-point Likert scale in both tasks.

Upon completing the four evaluation scenarios, we asked the subject his overall experience in using our prototype system in the form of descriptive feedback and a 5-point Likert scale (from terrible to great-idea). In the debrief, we informed the subject that all applications used were in reality good software available for the N810 tablet; all ratings and reviews had been scripted for experimental purposes only.

Results. Installation count in each evaluation scenario is shown in Table 3 (right). In S1, without any software reviews, 65% of the subjects went ahead to buy/install an application. The installation ratio decreased slightly (from 65% to 50%) in scenario S2 but dropped drastically (to 20%) in S3. Using the installation ratios, we evaluated the T1, T2 and T3 tests with the respective null hypothesis NH3, NH4 and NH5:

(T1) NH1: Negative community review does not overrule positive review by friend

(T2) NH2: Negative review by friend does not overrule positive community review

(T3) NH3: Overall strength of review by friend is not stronger than that of community review

Installation ratio in S1 served as the baseline of T1 and T2 tests (i.e. T1 compared the ratio in S2 to S1, while T2 compared the ratio in S3 to S1). Meanwhile, T3 was performed by comparing the ratio in S3 to S2. The hypothesis tests were evaluated using (one-tailed) Chi-square (good-of-fit) and binomial exact test. We favor results from binomial test as Chi-square statistics works better with a larger sample size.

Table 4. Results of hypothesis testing

	Chi-square	Binomial	Result
T1	p = .080	p = .122	NH1 cannot be rejected
T2	p < .001	p < .001	NH2 is strongly rejected
T3	p = .004	p = .006	NH3 is strongly rejected

We could not reject NH1 in T1. Although users reacted to negative reviews from online community members (resulting in a slightly smaller installation ratio in S2), the effect was not statistically significant. While we believe that users tend to react more towards negative reviews; warnings by online community members do not overrule positive feedbacks given by friends.

With T2, it was evident that negative reviews provided by friends overruled positive reviews by online community members. This was significant at 0.1% level.

The large ratio difference (30%) between S3 and S2 suggested the higher impact of information from friends. We evaluated this in T3. The overall strength of reviews by friends is stronger than reviews by online community members (significant at 1% level). The strength of (risk) signals communicated via friends should be exploited to mitigate click-through and careless behaviors during software installation.

We observed that the installation ratio in S4 (35%) was higher than in S3 (20%). We tested if the "bin-the-monster" mechanism had inadvertently reduced the effectiveness of risk signaling, and found that the effect was significant at 10% level. With our experimental "bin-the-monster" mechanism, a bin was shown after some delay when user clicked on a monster. However, the sudden appearance of the bin might have that caused subjects to prioritize binning the monster over reading the review. As it might not be obvious that the monster could be binned, we tried to assist the users by showing a hint (Figure 5, bottom). The short hint ("read the review and bin the monsters") might have been also construed as an instruction (or suggestive that it was ok to install) rather than to encourage a conscious decision. Our experimental 'bin-the-monster' mechanism was not a very successful one. An improved design could be to display the bin constantly to avoid a sudden appearance. The hint would need to be rephrased. A more direct association between the monster and review may also be helpful. For example, when user drags a monster into the bin, the corresponding review should be dragged together to signal that he is disregarding a review from his personalized community.

Table 5. On using Pacman-like monster as risk symbol

	μ	σ^2
Monster draws attention	4.3	.69
Monster gives clear message	2.8	1.3
Monster gives warning	3.8	1.1
Prefer monster over ⚠	3.2	1.1
Prefer monster over STOP	3.4	1.1

1=strongly disagree, 5=strongly agree

Table 6. Overall user experience

	μ	σ^2
Experience with habituation-breaking	3.5	1.5
Experience with social rating integrated with software installation	4.4	.61

1=terrible, 5=great idea

The reactions to the use of the monster as risk symbol were mixed (Table 5). While most subjects agreed that it drew attention (salient), a few noted that they did not get a clear message of risk/warning. Subjects remained neutral on preferring the monster over the conventional "stop" and "exclamation-mark" symbols. We interpret these as using a new risk symbol would demand extra effort in educating the users.

Experience with the experimental "bin-the-monster" habituation-breaking mechanism was diverse (Table 6). Some liked it and found it interesting, while a few found such mechanism unnecessary. We note that habituation-breaking mechanisms are designed to trade off some level of convenience for safer user actions, and may be hard to satisfy all users. Feedback on social rating (for software appropriateness) integrated with the installation process was, on the other hand, very positive. This suggests that it could be a useful feature on mobile devices (or other computing environments that involve installation of third party applications by ordinary users).

Limitation and Discussion. There are two weaknesses with regard to our user study. We note that the T3 test might have an order-bias as subjects were always required to complete scenario S2 before proceeding to S3. We should have mitigated this by randomizing the order of test scenarios.

We note that also the initial setting of "software offer" to provide subjects with motivation might not be very realistic. An alternative setting is to have the subjects to decide whether to buy/install an application on behalf of someone whom they care. However, we think that both settings have limitations that are hard to avoid in a laboratory testing. We could create a sense of realistic risks, for example by informing the subjects that they would be required to login to his email/bank account using the test device after the study. Yet, we thought that this was not too relevant as we did not require the subjects to evaluate whether to install software that are potentially harmful; our study concerned only applications that may be mildly inappropriate.

Summary of Findings

- Opinions by friends are stronger than that of by online community; warnings by friends overruled positive feedbacks by online community, but not vice-versa
- The experimental “bin-the-monster” mechanism needs to be improved; designing and evaluating an effective habituation-breaking mechanism remain as interesting research problems
- The response towards habituation-breaking mechanisms and a new risk symbol was mixed; yet, majority was very positive with the idea of integrated social rating

6 Related Work

It is well-known by now that improving only the visual salience of risk signals is not enough to ensure secure user behaviors. Studies [23,24] have shown the inefficacy of security toolbars and site-authentication images, which mainly rely on an improved risk salience. Brustoloni and Villamarin-Salomon [2] suggested using polymorphic dialogs (that will vary the order of decision options) to capture user attention and break habituation. They advocated also the use of audited dialogs that would keep track of user decisions to hold them accountable for irresponsible actions. However, subjects regarded audit dialogs as intrusive; audited dialogs also did not assist users to make better decisions. In addition to improving the visual salience (through a better interface design), our work here increased the relevance of risk signals by employing inputs from user’s personalized community.

Compared to FireFox’s approach of making potentially unsafe actions (e.g., browsing a site with invalid certificate) more difficult to slow-down the users, our experimental habituation-breaking mechanisms (albeit need further improvements) are complemented with context-relevant information from personalized communities, that is absent in FireFox.

Related to software installation is the study by Good et.al. [9] which found that displaying a short summary (especially right-after the normal EULA notice) can effectively reduce the installation of unwanted applications. Yan et.al. concluded that visualizing the reputation and a personalized trust value for applications can be a helpful feature on mobile devices [25]. These studies highlighted the importance of timely signals. Our work integrated risk signals from personalized communities with the installation process. This integration was very well received in our user study.

Our idea of the personalized community is similar to NetTrust's [4] which employs personalized rating against the threat of phishing. NetTrust employs implicit inputs of browsing and bookmarking history of friends, as well as, explicit recommendations from third parties like banks and Google. Continuing from the initial work in [11,5], in this paper, we have provided supports for the use of inputs from personalized communities, based on theories, a survey and a hands-on study on a prototype system.

7 Discussion and Future Work

Use of inputs from personalized communities is not without several shortcomings. We outline several challenges along with the potential mitigation strategies worth of future investigation.

Reliability. Inputs provided by user's personalized community may not be always correct. Information from technical sources may also be misinterpreted when guided through ordinary users. These issues can be mitigated by making the evaluation process more structured. For example, an evaluation can be divided into several aspects of software appropriateness rather than a single overall rating.

Coverage. Although users are likely to encounter similar applications with (some of) his friends in practice, undeniably ordinary users will have limited exposure and resources to identity all possible inappropriate applications. This is why we have included the notion of *expert users* (whom individual users trust) into the structure of a personalized community. A list of experts can be set by default (for all users) to deliver critical risk information. We could also extend our work to compute or infer recommendations for specific applications when there is no direct input from the personalized community. We note that there is much to learn from the field of recommender systems. However, this should be done with care so that the high relevance and strength of risk signals, as perceived by users, do not diminish.

Scalability. Software features such as usable contact and review sharing, re-usability of reviews (across mobile platforms) as well as robust handling of software versions would be helpful to scale our implementation. Rather than building a system of social networks from scratch, we plan to merge the prototype with existing services (such as Facebook) that are now seamlessly integrated with smart phones.

Incentives. Like any community-based systems, there are challenges in initiating and sustaining user efforts. An important future work is thus to design an incentive scheme that would encourage active user participation. Here, we note that in contrast to a "crowds" system (i.e. one that employs a global community, such as PhishTank and WOT) where the success of the system is a public good, our work can benefit from unselfish behaviors among members in the personalized community. Indeed, we have seen strong motivation to protect friends and family members in our survey.

8 Conclusions

We developed a set of design guidelines grounded on theories for a trustworthy software installation process. Through a survey, we verified the high relevance of inputs from a personalized community and user motivation to protect friends and family. We implemented a prototype system with contact management and reviews sharing capabilities as well as a redesigned installation task-flow. Our user evaluation confirmed the strength of information communicated through friends, while the idea of integrated ratings from a personalized community during application installation was very well-received.

There may be some challenges that need to be addressed in future work; given the high relevance and strength of inputs from known sources, we show in this paper, the potentials of relying on personalized communities to evaluate software appropriateness and to mitigate the problem of click-through habituation during installation.

References

1. Aarts, H., Dijksterhuis, A.: Habits as Knowledge structures: Automaticity in goal directed behavior. Journal of Personality and Social Psychology 78(1), 53–63 (2000)
2. Brustoloni, J.C., Villamarin-Salomon, R.: Improving security decisions with polymorphic and audited dialogs. In: Proc. SOUPS 2007 (2007)
3. Burt, R.S.: The social capital of opinion leaders. Annals of the American Academy of Political and Social Science: The Social Diffusion of Ideas and Things 566, 37–54 (1999)
4. Camp, J.L.: Reliable, usable signaling to defeat masquerade attacks. In: Proc. WEIS 2006 (2006)
5. Chia, P.H.: Secure software installation via social rating, Masters Thesis, Helsinki University of Technology (TKK) and Royal Institute of Technology (KTH)
6. Douceur, J.R.: The sybil attack. In: Proc. IPTPS 2001(2001)
7. Frederick, S.: Automated Choice Heuristics. In: Gilovich, T., Griffin, D., Kahneman, D. (eds.) Heuristics and Biases. Cambridge University Press (2002)
8. Gong, L., Ellison, G., Dageforde, M.: Inside Java 2 Platform Security: Architecture, API Design, and Implementation. Addison Wesley (2003)
9. Good, N.S., Grossklags, J., Mulligan, D.K., Konstan, J.A.: Noticing notice: a large-scale experiment on the timing of software license agreements. In: Proc. CHI 2007 (2007)
10. Heath, C.: Symbian OS Platform Security. John Wiley & Sons (2006)
11. Heiner, A.P., Asokan, N.: Secure software installation in a mobile environment (poster). In: Proc. SOUPS 2007 (2007)
12. Kahneman, D.: Maps of Bounded Rationality: Psychology for Behavioral Economics. The American Economic Review 93(5), 1449–1475 (2003)
13. Lazarsfeld, P., Berelson, B., Gaudet, H.: The people's choice (1944)
14. Lyn Bartram, L., Ware, C., Calvert, T.: Moving Icons: Moving icons: detection, distraction and task. In: Hirose, M. (ed.) Proc. INTERACT 2001 (2001)
15. María Ruz, M., Lupiáñez, J.: A review of attentional capture: On its automaticity and sensitivity to endogenous control. Psicológica 23, 283–309 (2002)
16. Moore, T., Clayton, R.C.: Evaluating the Wisdom of Crowds in Assessing Phishing Websites. In: Tsudik, G. (ed.) FC 2008. LNCS, vol. 5143, pp. 16–30. Springer, Heidelberg (2008)

17. Neal, D.T., Wood, W., Quinn, J.M.: Habits: A repeat performance. Current Directions in Psychological Science 15, 198–202 (2006)
18. Peters, R.J., Itti, L.: Beyond bottom-up: Incorporating task-dependent influences into a computational model of spatial attention. In: Proc. CVPR 2007 (2007)
19. Rogers, E.: Diffusion of innovation, 5th edn. Free Press (2003) ISBN: 978-0743222099
20. Rubinstein, J.S., Meyer, D.E., Evans, J.E.: Executive Control of Cognitive Processes in Task Switching. Journal of Experimental Psychology: Human Perception and Performance 27(4), 763–797 (2001)
21. Schneider, W., Chein, J.M.: Controlled and automatic processing: behavior, theory, and biological mechanisms. Cognitive Science 27, 525–559 (2003)
22. Schneier, B.: The psychology of security (2008), http://www.schneier.com/essay-155.html
23. Schechter, S.E., Dhamija, R., Ozment, A., Fischer, I.: The emperor's new security indicators. In: Proc. S&P 2007 (2007)
24. Wu, M., Miller, R.C., Garfinkel, S.L.: Do security toolbars actually prevent phishing attacks? In: Proc. CHI 2006 (2006)
25. Yan, Z., Liu, C., Niemi, V., Yu, G.: Trust Indication's Influence on Mobile Application Usage, NRC Technical Report (2009), http://research.nokia.com/files/NRCTR2009004.pdf
26. Yee, K.-P.: Aligning security and usability. IEEE Security and Privacy 2(5), 48–55 (2004)
27. Developing applications for Palm webOS using HTML, CSS and JavaScript, http://developer.palm.com/index.php?option=com_content&view=article&id=1603&Itemid=43
28. OviAppWizard for Symbian, http://oviappwizard.com
29. AppWizard for iPhone, http://www.appwizard.com/
30. StopBadware, http://www.stopbadware.org/
31. Java Verified Program, http://javaverified.com/
32. Symbian Signed, https://www.symbiansigned.com/app/page
33. F-Secure identified FlexiSpy as a spyware, http://www.f-secure.com/sw-desc/spyware_symbos_flexispy_f.shtml
34. Objections towards iTunes Appstore approval process, http://news.cnet.com/8301-13506_3-10317057-17.html, http://www.eff.org/deeplinks/2009/06/oh-come-apple-reject, http://www.eff.org/deeplinks/2009/05/apple-says-public-do, http://www.eff.org/deeplinks/2009/02/south-park-iphone-app-denied, http://www.thelocal.de/society/20091125-23501.html
35. PhishTank, http://www.phishtank.com
36. Web of Trust, http://www.mywot.com