ABSTRACT

Title of thesis:	A HUMAN-CENTERED APPROACH TO IMPROVING THE USER EXPERIENCE OF SOFTWARE UPDATES
	Arunesh Mathur, M.S. in Human-Computer Interaction, 2016
Thesis directed by:	Assistant Professor, Marshini Chetty College of Information Studies

Software updates are critical to the security of software systems and devices. Yet users often do not install them in a timely manner, leaving their devices open to security exploits. This research explored a re-design of automatic software updates on desktop and mobile devices to improve the uptake of updates through three studies. First using interviews, we studied users updating patterns and behaviors on desktop machines in a formative study. Second, we distilled these findings into the design of a low-fi prototype for desktops, and evaluated its efficacy for automating updates by means of a think-aloud study. Third, we investigated individual differences in update automation on Android devices using a large scale survey, and interviews. In this thesis, I present the findings of all three studies and provide evidence for how automatic updates can be better appropriated to fit users on both desktops and mobile devices. Additionally, I provide user interface design suggestions for software updates and outline recommendations for future work to improve the user experience of software updates.

A HUMAN-CENTERED APPROACH TO IMPROVING THE USER EXPERIENCE OF SOFTWARE UPDATES

by

Arunesh Mathur

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2016

Foreword

Statement of Co-Authorship

All work in this thesis was conducted under the supervision of Dr. Marshini Chetty. Dr. Michelle Mazurek, Dr. Jessica Vitak, and Dr. Tudor Dumitras provided valuable feedback throughout the design and implementation of Study Two.

I collaboratively authored the research for Study One, with Josefine Engel, Sonam Sobti, and Victoria Chang. The resulting paper was published at SOUPS 2016:

Mathur, A., Engel, J., Sobti, S., Chang, V., & Chetty, M. (2016). "They Keep Coming Back Like Zombies": Improving Software Updating Interfaces. In Twelfth Symposium on Usable Privacy and Security (SOUPS 2016).

Dedication

To my parents, Arvind Kumar Mathur and Suman Mathur, who always supported me in my endeavors.

Acknowledgments

This thesis would not have been possible without the contributions of several others. I would like to start by thanking my advisor Marshini Chetty, who left no stone unturned in making sure I always had everything I needed to be successful: the right ideas, the necessary mentoring, and the timely feedback. Over the past two years, she championed my research, and provided invaluable guidance in every stage of graduate school—I'm glad our association is set to continue at Princeton.

My thesis committee members, Jessica Vitak, Michelle Mazurek, and Tudor Dumitras offered invaluable insights in the design of Study Two. I thank them for their feedback, and for meeting with me, often at short notice.

The students of the Human-Computer Interaction Lab (HCIL) helped keep me company during my time at the lab. These two years went by swiftly largely thanks to Kotaro, Alina, Matt, Deok, Fan, Adil, Josefine, Victoria, Daniel, Sonam and others—I will miss having them as my lab mates. My roommates Saurabh and Sanjeeb ensured I was never alone and hungry, and contributed to fairly frequent discussions about life, the universe and everything.

Finally, I would like to thank my parents, Arvind and Suman, and my brother and sister-in-law, Animesh and Mary, for their unconditional love and support—despite the hardships that arise as a result of being geographically located in three distinct continents.

This research was funded by the Maryland Procurement Office under contract H98230-14-C-0137.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Motivation and Research Problem

Vulnerabilities in client-side applications that run on user devices such as desktops and mobile devices are on the rise. For instance in 2014, Symantec [1] reported that 17% of all Android applications were malware and 20% of all website vulnerabilities were critical in nature. Typically, software vendors roll out software updates or “patches” to protect users by fixing these vulnerabilities and making changes to the software—such as adding new features, enhanced performance, or bug fixes. For this reason, the United States (US) government, various security agencies, and security experts advise end-users to download and install updates in a timely fashion to keep their systems secure [1–3].

Software vendors use several different approaches to push updates to end-users: some choose to automate the update process while others choose to actively involve users. Both mechanisms have shown to affect end users in different ways. Recent studies have shown that when end-users are involved in the update process, they report delaying updates because they lack awareness on the importance of installing these security patches [4] or have had previous negative experiences with updating [5]. On the other hand, when updates are fully automated, they can lead to poor mental models of how systems work [6]. This thesis explores how the process of automating updates can be enhanced to provide a user experience without compromising on the goal of ensuring updates reach users at the earliest. It makes practical design suggestions for update interfaces on desktop and mobile devices by personalizing them based on various criteria, including individual user personality traits.

1.2 Research Questions

Concretely, this thesis investigates how automatic updates can be adapted to provide a better user experience, and puts forward the following research questions:

1. How do users navigate through the software update process on desktop and mobile devices?
2. What are the challenges users face and what factors prevent them from applying updates when using these devices?
3. How can the update experience be improved to increase update patching rates using automatic updates on both desktop and mobile devices?

1.3 Approach

To answer these questions, I conducted two studies to explore the design space of update systems on desktop and mobile. Study One consisted of three different phases. In Phase One, along with a research team, I interviewed 30 United States

(US) Internet users in a qualitative formative study of how they navigate through the software update process on desktop machines. In Phase Two, I distilled our findings into the design of a *minimally-intrusive*, *information-rich*, and *user-centric*, low-fidelity prototype of an alternative desktop software updating interface for the Mac OS X operating system. In Phase Three, I conducted a think-aloud study with 22 OS X users to evaluate our designs and put forth recommendations for automating updates on desktop software updating interfaces.

In Study Two, I explored how individual differences in users contribute to how they choose to update the applications on their Android mobile devices by means of a large scale survey and qualitative interviews with users. I further examined users' preferences towards installing updates automatically, and used the results to generate design recommendations for mobile updating interfaces.

1.4 Contributions

This thesis consists of two studies one geared at improving the automatic software updating experience on desktop and mobile. It makes the following contributions:

1. A qualitative study investigating why users fail to apply updates in a timely fashion or avoid updates altogether on desktop systems

2. The design and think-aloud evaluation of a low-fidelity prototype of an improved software updating interface for the Mac OS X

3. A quantitative study using a large-scale survey revealing correlations between users' personalities and how they choose to update on mobile devices

4. A qualitative study investigating the limits of update automation on mobile devices

1.5 Overview

This thesis is organized as follows. The second chapter summarizes related work in Usable Security and software updates. The third chapter describes Study One, including the formative qualitative study of desktop update experiences, and the design and evaluation of a low-fidelity desktop update interface for the Mac OS X operating system. The fourth chapter describes Study Two, including the large scale survey and interviews of Android mobile users highlighting differences in individual users towards automatic updating. Finally, the fifth chapter puts forth the discussion from both the studies, summarizes the key takeaways, and outlines directions for future work.

Chapter 2: Related Work

In this chapter, I highlight previous research related to software updates. First, I describe the software update process and the various capacities in which it involves end users. Next, I highlight studies and related research in the security and software engineering communities about developing and distributing software updates. Finally, I highlight related work from the Human–Computer Interaction (HCI) and Usable Security communities about how end users experience software updates, and contextualize the research questions in this thesis relative to previous work.

2.1 Software Updates and Automation

The Usable Security community has long recognized that human beings are the "weakest link" in the security chain [7, 8], attributing many security failures to human factors. Moreover, the community has recognized that most security decision making cannot be fully automated because humans often have to perform a part of the task—such as responding to security warnings (e.g., SSL [9], [10]) and identifying phishing emails [11]. Additionally, automation is often context dependent and limited by failure cases [12]. To compensate for the human element in security systems, Cranor [13] developed a framework to help designers fully consider all the factors to integrate users in the loop for security decisions in various systems. Software updates are no exception.

In fact, software updates typically involve users at various stages of the update process. An update process often varies based on the device type, operating system and application [14,15], and the degree of automation and user involvement in each step can result in significantly different update experiences. Generally, a software update involves [16,17]:

1. *Discovering the Update*: Users can either search for updates manually on websites or app stores, or set updating preferences for a specific application or the operating system to automatically notify them when updates become available.

2. *Downloading the Update*: Users can choose to either download available updates manually or set preferences for the system to automatically download them on their behalf.

3. *Installing the Update*: Users can manually install or have their system automatically install updates. Installation may involve closing applications affected by the update and often, an update is only applied after an application restart or machine reboot.

4. *Using the System Post-Update*: Once applied, updates may notify users that they have completed.

Depending on the degree to which the update system notifies and involves users, software update preferences are often referred to as [6,17]:

- *Manual*: Users initiate and complete all the steps of the updating process, e.g., software drivers for input and output computer peripherals.

- *Automatic*: The update system automates one or more steps of the updating process such as downloading, installing, and notifying users. Users may have to briefly discontinue using the application to complete the installation or perform a restart of their machine or application, e.g., MS Windows patches and MS Office updates.

- *Silent*: The update system automates the entire update process and in addition, does not notify users explicitly at any step. Typically, in a silent update, the system installs the update without interrupting the user and applies it when users restart or re-open the application. Often, users fail to notice such updates, and lack the provision to disable or prevent them [18], e.g., Google Chrome updates [17].

In terms of reaching users' machines soonest after release, recent studies suggest that silent updating mechanisms may be the most effective in patching machines after an exploit is disclosed when compared to methods requiring a user's consent to download, install, or apply an update [17,18]. Most software vendors and the US government recommend automatic updates for users to keep their systems secure instead of manual updates for this reason [1–3].

2.2 Deploying Software Updates

Numerous studies have explored ways to develop and deploy software updates, and compared the effectiveness of different mechanisms. For instance, Duebendorger and Frei studied the effectiveness of silent updates [17], Vojnovic *et al.* studied automatic patches [19], and Gkantsidis and Karagiannis studied the Windows patching system for distributing patches on a planet scale [20]. These studies comment on each patching mechanism's strengths and weaknesses and make suggestions for improving the creation and distribution of patches at scale but with no focus on how users will appropriate these updates. Another set of studies focuses on improving the deployment of patches in large organizations [16, 21]. For instance, Oberheide *et al.* help network administrators infer the impact of patches before deployment [22]. Others have investigated how to improve the deployment of patches via USB drives in regions with sporadic connectivity [23]. These studies have focused on improving the creation and deployment of the software patches and updates but not the end users who apply these patches, why they avoid patches, or suggestions to improve their update interfaces.

2.3 Users' Attitudes and Behaviors with Software Updates

There is a growing body of work focused on understanding users' general online security behaviors and on user barriers to software updates. For example, Ion *et al.* [4] compared the capability of expert and non-expert users to process security advice. They found that non-experts updated their software less frequently compared to experts, lacked awareness about the effectiveness of software updates, and avoided updates that they felt introduced software bugs. Similarly, Wash and Rader surveyed almost 2000 US Internet users and found only 24% used protective security behaviors such as downloading patches [24]. Other studies show that users often disable or only perform updates on WiFi networks when they have limited and expensive Internet data plans [25, 26]. While these studies, provide some evidence that users infrequently apply updates and touch on a few barriers in the process but they are not solely focused on users and software updates.

Several researchers have studied users and software updates in more depth. For instance, Fagan *et al.* [27] surveyed 250 users about attitudes toward software updating notifications. They found that users were reluctant to apply updates because they disliked being interrupted by notifications which were often perceived as obscure and unclear. In complementary work, Vaniea *et al.* [5] studied 37 non-expert MS Windows users and found that past negative updating experiences, such as dealing with user interface changes that required re-learning how to use an application, affect future updating behaviors. In another study of the same Windows users, Wash *et al.* [6] found that users' updating behaviors and intentions with their updating preferences are mismatched, often resulting in less secure systems. The authors conclude that there is a tension between automation and control in the updating process which may be difficult to resolve through improved usability alone. Forget *et al.* [28] compared the level of engagement, or involvement, of end users in the management of their computers security and the state of the security of their computers, and discovered that greater engagement did not always correspond to a secure and updated system. The authors suggest that security mitigations, such as software updating, need to be designed according to how much users engage with computer security, so they are better equipped to take the right action. In more recent work, Vaniea *et al.* [29] surveyed 307 users about their experiences with updating software and like the research in this thesis, outlined the process of updating software and highlighted various user experiences in each step. While these studies unearth reasons behind users' software updating behaviors and attitudes towards automatic updates, this thesis goes a step further to offer concrete suggestions toward improving software updating interfaces.

2.4 Software Updates Interfaces

Thus far, only two studies have sought to improve updating interfaces. First, Sankarpandian *et al.* [30] developed a desktop graffiti system TALC, which reminded users to install updates by painting their desktop with graffiti when their machines were left un-patched for a long amount of time. These researchers focused more on how to improve the process of gently notifying and nudging users to pay attention to install updates rather than with improving the overall experience of updating. Second, Tian *et al.* [31]—the only study thus far for software updates on mobile devices—developed a novel updating notification that used user generated reviews to help mobile users make privacy conscious decisions about which updates to apply based on what permissions were asked for by the updates. In contrast, this thesis deals with not just update notifications, but interfaces that deal with the update process as a whole both on desktop and mobile.

2.5 Individual Differences in Security and Software Updating

Only recently have researchers begun exploring differences in individuals to tailor and personalize security mitigations. Egelman and Peer developed [32] and validated [33] the Security Behavior Intentions (SeBIS) scale to measure behavior intentions across four different domains: Password Generation, Proactive Awareness, Device Updating, and Device Securement. The authors demonstrated how the various SeBIS dimensions correlated with people's personalities. In particular, they examined the following psychometric constructs:

1. **Decision Making:** The General Decision Making Scale (GDMS) [34] measures people's decision making styles across the following dimensions: Rational, Avoidant, Intuitive, Dependent, and Spontaneous.

2. **Risk Taking:** The Domain Specific Risk Taking (DoSpeRT) scale [35] measures people's risk taking propensity across the following five dimensions: Ethical, Financial/Investment, Financial/Gambling, Social, Recreational, and Health/Safety.

3. **Future Consequences:** The Consideration for Future Consequences (CFC) scale [36] measures how much people consider the long-term consequences of their actions and decisions.

4. **Cognitive Endeavors:** The Need for Cognition (NFC) scale [37] measures how much people consider and indulge in thought and curiosity provoking endeavors.

5. **Impulsiveness:** The Barratt Impulsiveness Scale (BIS) [38] measure impulsive behavior across the following dimensions: Attention, Motor, and Non-Planning.

With their data, Egelman and Peer observed the following correlations between the psychometrics and keeping software up to date:

1. People who scored high on the GDMS-Rational trait were more likely, and people who scored high on the GDMS-Avoidant and GDMS-Spontaneous trait were less likely to keep their software up to date than people who scored low on these scales

2. People who scored high on the DoSpeRT-Ethical and DoSpeRT-Health/Safety scale—or high risk takers—were less likely to keep their software up to date then people who scored low these scales

3. People who scored high on the CFC scale—or those who considered the future consequences of their actions—were more likely to keep their software up to date than people who scored low

4. People who scored high on the BIS—or those who displayed impulsive behavior—were less likely to keep their software up to date than people who scored low

5. People who scored high on NFC—or those with a greater propensity to engage in cognitive endeavors—were more likely to keep their software up to date than people who scored low

The authors suggested that these psychometrics may have the potential to infer users' security intentions, and therefore be used to personalize and tailor security mitigations. While the authors established correlations between these various personality traits and the likelihood to keep software updated, I use these to tease out differences between attitudes towards automating updates, and use these differences to make design recommendations for update interfaces on mobile devices.

2.6 Chapter Summary

In this chapter, I first explained the software update process, and how, the degree to which users are involved in this process determines how updates are delivered to them. I draw from research in the systems and software engineering domains, and highlight how the process of software updating demands attention because they directly affect and change end users' systems. I then list how the Usable Security field has contributed to our understanding of users' preferences and behaviors but has not considered ways to improve software updating interfaces thus far, both on desktop and on mobile—the primary contribution of this thesis.

Chapter 3: Study One: Automating Updates on Desktops

3.1 Overview

Study One consisted of a three-phased research process as shown in Table 3.1. Phase One examined how users navigate through the software update process on desktop machines, and the challenges they face in doing so. Phase Two and Phase Three cover the design and evaluation of a low-fidelity interactive prototype for the Mac OS X operating system.

Phase	**No of Users**	**Timeline**
1. Formative Study	30	Jun '14–Sep '14
2. Prototype Design	–	Oct '14–Feb '15
3. Prototype Evaluation	22	Feb '15–May '15

Table 3.1: Research Timeline Overview.

3.2 Phase One: Formative Interviews

In Phase One, I, and two other researchers, investigated why users avoid installing updates in the first place on desktop machines through interviews with 30 participants.

3.2.1 Procedure

In mid to late 2014, we recruited 30 participants to take part in semi-structured interviews about their overall experience with software updates, including their likes and dislikes about software updates and their current software updating behaviors. We recruited participants through advertisements on university and affiliated mailing lists around the US, and social media (Facebook, Twitter) posts. We focused on finding adult Internet users that used Internet-enabled devices such as a desktop, laptop, tablet, or smartphone since they were likely to encounter software updates frequently. All interviews were conducted over the phone or Skype, audio-taped, and lasted between 45–60 minutes each. Participants were compensated with USD 15 gift cards for their time. The study was approved by the Institutional Review Board (IRB) of the University of Maryland (UMD).

The interview guide was developed after a survey of the existing literature on software updating and Usable Security at the time and informed by Cranor's human in the loop framework [13, 17] to ensure all aspects of the software updating process were covered. Cranor's model describes the human factors that affect secure systems, namely: *Communication* (informing the human that an action is necessary), *Communication impediments* (what might prevent the human from taking the action?), *Characteristics of the human receiver* (demographics, intentions, comprehension, and knowledge retention), and finally, *Behavior*. We used the framework to tease out the various human elements in the software updating process I discussed in Chapter 2. Concretely, we walked the participants through the specifics of the update process—discovering, downloading, and installing updates—and asked them the following questions for their applications and operating systems:

1. How do users learn about updates on their machines? Do they manually seek updates or wait for notifications?
2. Do users feel updates are important to security? What motivates them to either install or avoid updates?
3. How do users navigate the update process and how do they make decisions about security vs non-security related updates?
4. Do updates interrupt users' workflow? How does this affect their behavior?
5. Do users understand software update change logs and more generally, what action updates ask of them?

We also asked participants whether they ever changed, or sought help to change, the default update preferences for their operating systems and applications. In addition, we collected the participants' demographics (age, gender, education, income), their security management practices on their Internet-enabled devices such as installing anti-virus or enabling firewalls, their online security knowledge/actions when downloading software and dealing with suspicious emails, and past experiences with security incidents. The interview guide in its entirety is available in Appendix A.

3.2.2 Analysis

Once the interviews were transcribed, three researchers independently analyzed the transcripts. We inductively looked for patterns and threads in the data, marking them with labels, and then grouped and organized these labels into themes [39]. The research team held regular meetings to discuss the initial results during this time, and arrived at the final set of themes shown in Table 3.3 after multiple rounds of discussions and consensus building. Example themes included "Updates interrupt users" and "Users need information for decision making". In the following section, I use the prefix P to indicate an interview participant.

3.2.3 Participants

Table 3.2 summarizes Phase One's demographics. Participants were predominantly between 18–34 years old, with a fairly even gender split. Most were educated, having college degrees, lived in the District of Columbia, Georgia, and Maryland, and earned a median annual income of USD 60,000. All participants owned desktops and 24/30 owned laptops as well. Two thirds of the participants used a single operating system: MS Windows (15/30), Mac OS X (3/30), and Linux (2/30). The remaining third used a combination of two operating systems: both Mac OS X and MS Windows (8/30) or Linux and MS Windows (2/30).

A large portion of the participants were aware of security breaches that were heavily publicized in the media. For instance, 18/30 were aware of the Heartbleed bug [40] and 11/30 were aware of the 2013 Target breach [41]. One-third of the participants had been victims of online breaches and malware including credit card frauds and computer viruses, and 8/30 participants stated they went above and beyond their e-mail providers' services to maintain their security. For instance, one participant reported using text-only mode for reading messages, and another reported scanning all downloaded attachments. Overall, while the sample was gender balanced, it represented a younger, more educated, and as a result, more technology savvy set of users.

Demographic	**Phase One (N = 30)**	**Phase Three (N = 22)**
Age		
18–34	66.7%	95.5%
35–54	26.7%	0%
>55	6.6%	4.5%
Gender		
Male	53.3%	36.4%
Female	46.7%	63.6%
Education		
College	6.7%	45.5%
Bachelor's	30.0%	40.9%
Master's	36.7%	9.1%
Other	26.6%	4.6%

Table 3.2: Demographic Information: Phase One and Phase Three.

3.3 Findings

The formative study showed that software updates interrupt users and their computing activities, supporting findings of previous studies [5, 27], for users of operating systems other than MS Windows. This study also illuminates new evidence of information barriers to updates namely: trust in vendors, obscure change logs, and unknown installation times. Information barriers extend beyond the wording of unclear and obscure notifications [27] to the update's purpose, possible consequences of applying an update, and information to plan when to do an update. We also noted that unlike in constrained settings [25,26], the participants were less concerned about updates using up Internet data. Finally, the participants varied on how they wanted to be notified or provide consent for updates based on the frequency of application use and the changes the update was going to perform. Additionally, they wished to manage and control all the updates on their devices centrally.

3.3.1 Interrupting Users

While participants appreciated the importance of software updates for maintaining security, enhancing performance, and adding new features to their software, they felt that updates disrupted their computing activities in two ways: Interruptive update notifications and reminders diverted their attention while unwanted reboots and context switches lowered their productivity.

Notifications and Reminders: 22/30 participants reported that inopportune update notifications caused the largest disruption because they appeared during regular computing activities such as watching a video, doing a presentation, or during work times. These notifications were also hard to dismiss completely, so the same update could interrupt a user multiple times with reminder prompts. In a typical example, P17 remarked: *"I tend to let the update notifications go away but these days it looks like people keep forcing it so it comes back and back like a zombie."* The participants prioritized dismissing update notifications and opted for reminders.

Yet, these intrusive messages led to them ignoring many software updates because frequent interruptions were annoying and required active attention.

Rebooting and Context Switch: 19/30 participants reported that they delayed updates if they thought the update would require them to reboot machines, restart applications, and save their work. P9's example captures participants' feelings: *"I absolutely put them off until later, because the update requires me to stop what I'm doing, restart the program and computer, and then completely try to reconstruct where I left off."* Even if participants went through with updates, they became frustrated at having to recreate the context of their activities from which they were interrupted. This caused a negative perception of updates as a disruptive force. In another illustrative example, P12 expressed displeasure about restarts losing the context of open tabs in a browser: *"Usually when it tells me I have to shut down my browser, that's when I'm not happy. That and restarting, especially if I know I have a lot of windows or programs or something open, having to restart."*

3.3.2 Information for Decision Making

We asked participants what information they actively sought or wanted for informing decisions about applying software updates. They reported the following factors:

Update Categories: Vendor-specified update categories influenced the participants' decision making. 24/30 participants said they prioritized performing "major updates" including operating system and security related updates over others.

P5's quote exemplifies the reasoning: *"I think if I saw the words security or something along those lines, I would be more apt to do the updates than if it said, you know, this improves usability."*

Other participants revealed that existing vendor categories for updates inadequately captured an update's purpose and often led to them ignoring an update as P20's quote highlights: *"Just being told that it is critical does not really make me feel like it is critical. I need to feel the urgency and feel like there could be a consequence if I don't update it."* Concretely, they mentioned not knowing whether the update improved performance, fixed bugs, or enhanced security upfront and that these factors helped make decisions about going forward with an update or not.

Update Change Logs: Two thirds of the participants reported they glanced through the update change logs. In one telling example, a participant elaborated: *"I read almost all update notes and if it does not have any then I am going to disregard it. I take it pretty seriously. I have to know what the update is going to do on my software." (P17)* However when asked to reflect more deeply, 6 of those admitted the change log was unlikely to influence their decision to update. Close to half of the participants on the other hand, felt logs either presented too little or too much information, remarking that change logs could use less technical language or visual cues to better interpret the update information.

Trust in Vendors: The participants' opinion of the software vendor influenced their decision to go forward with an update. 19/30 mentioned that they preferred updates from sources they trusted such as the app store or through the vendor's official website. This trust, they further explained, was amplified either through the reputation of the vendor (e.g., a large software company such as Microsoft or Apple), or through positive past experiences with applying updates from that vendor. P2's example quote captures this sentiment: *"I'm pretty good about—just when I see an update request, if it's from a source I know and trust—running it right then."* For the participants, trusting an update's creator was crucial for making a decision to go forward with a suggested update.

Even though trust was important to the participants, they often had trouble finding reputable sources for updates as P8 explains: *"Sometimes finding reputable sources for updates can be challenging and some software packages put their software out on all sorts of different sites. And, you know, it's like which one of these guys do I really trust to download from?"*. Participants tried to ensure that the updates they installed were legitimate but found it difficult to easily determine the authenticity of an update in current updating interfaces.

Compatibility Issues: 16/30 participants struggled with updates that caused unexpected consequences such as removing certain features they used or that led to compatibility issues with other software. Other participants felt that they were forced to install updates to ensure that software they used frequently would not stop working. In an example illustrating this theme, P13 said: *"Typically it's because I have no other choice. If the program that I want won't run on the version of Windows that I have, and I really want to run that program, I'll do the operating system update."* In another instance, P7 complained that their computer crashed after an update and they had to perform a system restore to get things working again. Overall, compatibility issues made participants reluctant to apply updates especially since they could not predict these interactions in advance.

User Interface Changes: 16/30 participants were dissatisfied with updates that changed the user interface because they had to re-acquaint themselves with the application. This is captured by P9's quote: *"For example, one of the updates on one of my frequently used programs switched around the confirm and cancel buttons."* This change caused him to inadvertently erase documents that he needed following the update. Participants thus became averse to updates with user interface changes but more importantly, they could not always predict which updates would have these changes.

Social Influences: Nearly a third of the participants discovered security flaws and updates through social influences such as online media blogs, the news, or through family and friends. While in some cases these social cues pushed participants to actively seek out updates, in other cases—especially with large and critical updates—they made users cautious about performing updates they had been warned against by the media or social networks. For example, P3 said: *"Usually when I hear about updates from things like PC Magazine or those people start talking about it, and then I would just read about it—just to get an idea before I even consider whether to do it."* In another instance, P6, a frequent Mac OS X user explained: *"There are some that I don't do at least until I go online and research it."* Participants therefore depended not only on vendor-specified information but also on social networks and the media to inform them about whether or not they should apply a particular update.

Results Post-Update: A little over one-third of the participants mentioned that they could not always discern the changes an update made post-update, because they received little, if any, feedback about the update's actions. This made them question the overall benefits of updates especially when they had invested considerable time and effort in applying the update (e.g., interrupting their primary activity and rebooting their machines).

Infrastructure Constraints: 8/30 participants told us they avoided updates because of infrastructure constraints such as insufficient disk space and slow Internet connections or because they believed updates slowed down their machines. In a typical example, P23 explained: *"Some software updates take a lot of additional space because it always comes with that extra storage amount that I need. So it is like all the updates that I am doing are only making my computer slow so it's an annoyance."* Participants also avoided updates to save data but to a lesser extent than suggested by findings in settings where Internet constraints are more prevalent [25, 26]. For this reason, participants told us they wanted the update size in advance to more easily weigh the costs of doing an update.

Installation Time: Because software updates disrupted the participants' workflow, they sought information to organize their activities such as the time required for updates to complete. 7/30 participants indicated a willingness to perform updates if they had access to this information in advance. P13 summed this up as: *"But if they actually tell me how long it'll take, that will make me more willing to start an update."* Knowing how long an update process would take, participants told us, would allow them to perform updates at convenient times.

3.3.3 Users in the Update Loop

We asked the participants how they wanted to be notified and provide consent for downloading and installing updates.

Frequently Used Applications Matter: 17/30 participants mentioned that the frequency of use and their perception of an application's importance determined the degree of care they expressed towards keeping software and devices updated. In one example, P15 explained: *"An Evernote plug-in was not up to date and it asked me to update it. And I just deleted it because I don't want to deal with going through an update for a program that I don't use all that much."* Participants were most concerned about applications that mattered to them in some way; either by frequency of use or if it served some crucial function for them.

Tracking Updates: Just over a third of the participants found it difficult to track update downloads and installs because update settings and notifications were spread over multiple locations for the operating system and third party applications. 11/30 talked about needing a central update manager to review updates for all the software installed on their devices. Specifically, participants found it difficult to easily tell what needed to be updated and when or how to change the update settings for applications and the operating system. Overall, participants desired a central location on their devices to track all updates.

At the end of Phase One, the research team decided to focus on addressing three main areas of concern stemming from the formative work as show in Table 3.3, namely updates interrupting users, the lack of adequate updating information, and

Phase One Themes	Participants (N = 30)
Interrupting Users	
Notifications & Reminders	22
Rebooting and Context Switch	19
Information for Decision Making	
Update Categories	24
Update Change Logs	20
Trust in Vendors	19
Compatibility Issues	16
User Interface Changes	16
Social Influences	12
Results Post-Update	12
Infrastructure Constraints	8
Installation Time	7
Users in the Update Loop	
Frequently Used Applications Matter	17
Tracking Updates	11

Table 3.3: Themes from Phase One.

finding ways to keep users in the update loop. I describe the part of the user-centered design process next.

3.4 Phase Two: Prototype Design

In Phase Two, I, and the two other researchers created a low-fidelity, interactive prototype using MS PowerPoint to improve how users receive updates on their machines and how an update system could scaffold users' decision making. The prototype contains images of the mocked up updating interfaces linked together to create the illusion of a working system. Users can navigate through the mock system using the links to explore a limited set of predefined features with a system concept for each feature. The goal of the prototype was to elicit user reactions to the different design concepts it embodies as described in this section. As such, we

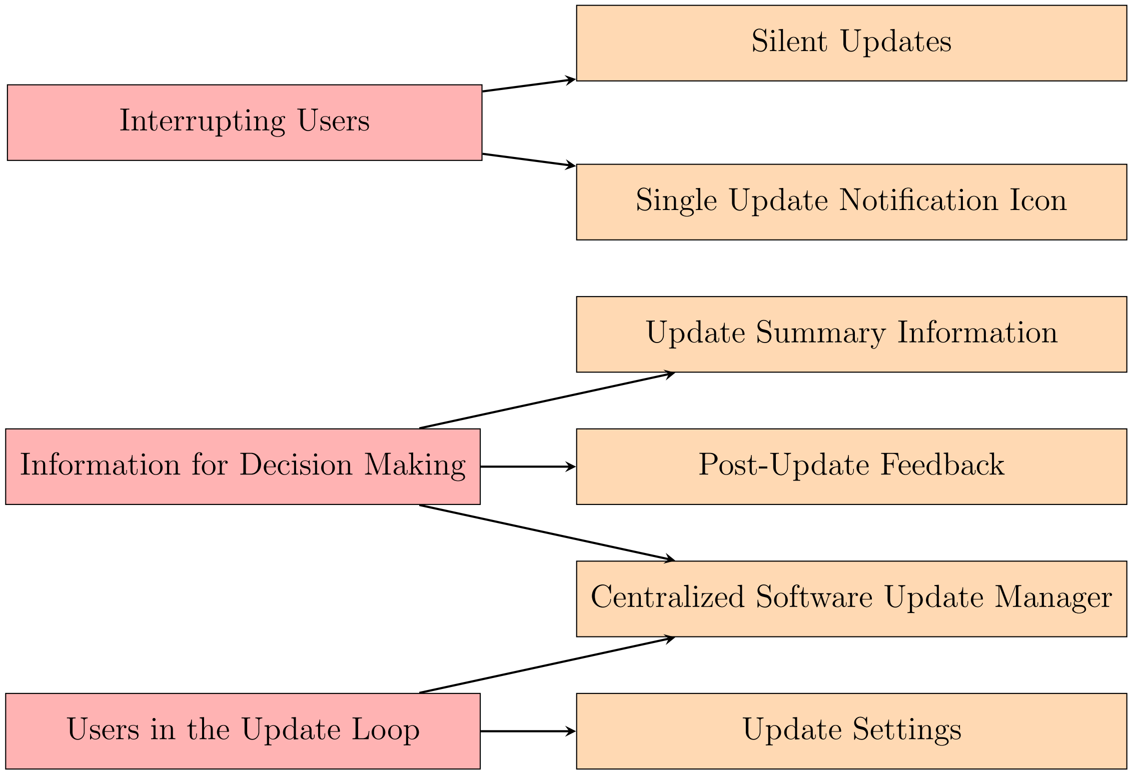

Silent Updates
Interrupting Users
Single Update Notification Icon
Update Summary Information
Information for Decision Making
Post-Update Feedback
Centralized Software Update Manager
Users in the Update Loop
Update Settings

Figure 3.1: Themes from Phase One (Left) and the Resulting Prototype Features (Right)

focused less on the implementation details such as how the information presented could be acquired ahead of time.

Given that both desktop and mobile are heading towards an app-based model of software distribution [42–44], we chose to create the prototype for Apple's Mac OS X, a major operating system player [45], that has been using this model since 2010 [46].

3.4.1 Procedure

After identifying three areas of concern to users, the research team explored the design space for improved updating interfaces. We began with a lightweight sketching, brainstorming, and ideation phase over these themes. After several iterations of sketches, mockups, and designs were refined by feedback sessions with the research team, we settled on the following issues to alter in updating interfaces. First, we modified how users are interrupted about updates and the manner in which they provide consent to updates. Second, we augmented the information users need for making decisions about updates such as providing clear and concise update logs and the type of the update. Third, we consolidated the updates across a device into a single update manager for users to keep track of updates. We then sketched multiple designs of improved interfaces, discussing and validating the decisions we took at each point, and condensed these into a prototype I describe in the following paragraphs. Figure 3.1 illustrates how the themes from Phase One resulted in the various prototype features.

3.4.2 Altering Update Interruptions

The *minimally-intrusive* low-fidelity interactive prototype alters how end-users are interrupted by either update or reboot notifications in two ways in the design: first, all update notifications are pushed through a single channel icon, and second, all update requiring restarts are piggybacked to other times when users restart their applications or devices. We designed the following features in the interface mock-ups

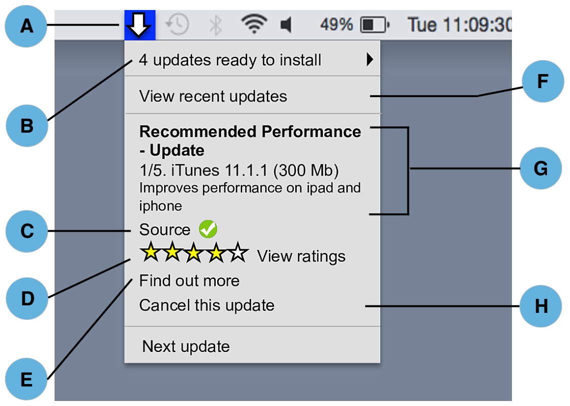

A
B
C
D
E
F
G
H
49%
Tue 11:09:30
4 updates ready to install
View recent updates
Recommended Performance - Update
1/5. iTunes 11.1.1 (300 Mb)
Improves performance on ipad and iphone
Source
View ratings
Find out more
Cancel this update
Next update

Figure 3.2: Update DropDown Menu. A: Update Icon. B: List of Remaining Updates. C: Source Verification. D: Update Ratings. E: Additional Information In Central Update Manager. F: List of Recent Updates In Central Update Manager. G: Summary Information For Current Update. H: Cancel Current Update.

to reflect these concepts:

Single Update Notification Icon: All update notifications are reduced to a minimal visual cue, the subtle animation of a single system tray icon (Figure 3.2A). This inverted arrow icon animates only when an update is being downloaded or installed. Users can click on the icon to display a list of impending updates (Figure 3.2B).

Silent Updates: Conceptually, our design forces all updates to download and install automatically by default without a user's consent. When available, we envisioned that an update lives in the list of updates for a buffer period (e.g., 24 hours) to allow users to intervene. If needed, users can cancel it via the "Cancel this update" option (Figure 3.2H). When this buffer period expires, in this design concept, the update is automatically downloaded and installed but users can continue using their applications without any reminders to restart. In cases where the restart is fundamental for an update to function, in this design concept, users' systems may remain unpatched until the next restart. This design does not eliminate disruption from unwanted update changes but shifts the onus onto the user to decide whether or not to proceed with an impending update based on additional information. We made this design decision to be provocative to evaluate if users prefer a universally "silent" update mechanism across all their operating system.

3.4.3 Addressing Lack of Information

The *information-rich* design adds to existing update information to help users accept or ignore updates via an update summary and post-update feedback.

Update Summary: Each impending update (Figure 3.2G) contains four important details we learned were lacking in the formative study and a "Find out more" (Figure 3.2E) link to more details in the centralized update manager:

1. *Source Verification:* The design displays a green "tick-mark" (Figure 3.2C) next to the name of the software vendor to indicate a verified and trusted source.

2. *Update Type:* We tag each update with one of five categories (Figure 3.2G): UI fix (user interface changes), Bug fix (fixes software bugs), Security fix (fixes a major security flaw), Performance (performance enhancements), Compatibility (could cause compatibility issues with other software) to help users learn the update's purpose at a glance.

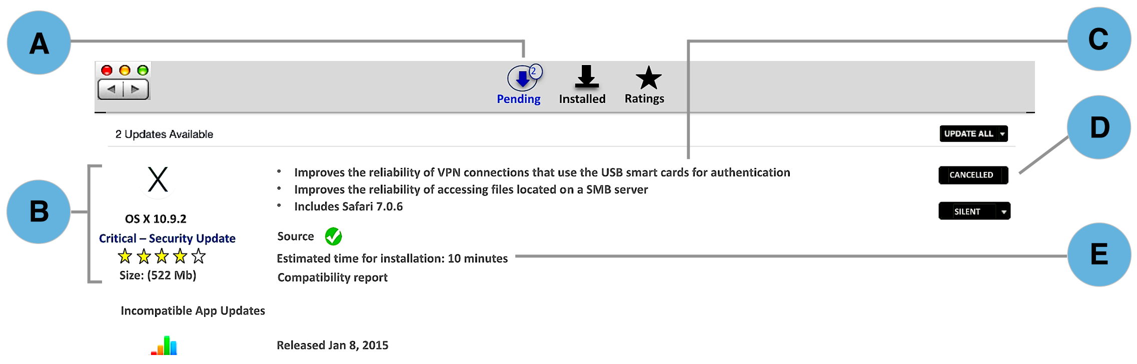
3. *Update Size:* To inform users about the data and disk space an update might consume, we display the size of the update (Figure 3.2G) in the summary.

4. *User Ratings:* Each update displays a five star rating (see Figure 3.2D) based on other users' experiences with it; with one star being poor and five stars being excellent.

Post Update Feedback: Participants in the formative study could not easily identify when an update was installed or what changes were made by the update. The design shows a pop-up message when a user closes a newly updated application or the operating system for the first time post-update as shown in Figure 3.5. This message shows the changes made to the application and the date on which the most recent update was installed. The pop up also forces the user to rate the update before they can close the application. In this design concept, update ratings are mandatory to ensure they are eventually populated with information and to force participants to comment on this feature.

3.4.4 Centralizing Update Management

In the formative study, users desired a way to track *all* the updates across their device. The *user-centric* prototype conceptually houses and controls the information and settings for all software updates through a central software update manager on the device. A "Pending" tab (see Figure 3.3) shows the updates that have been downloaded but not yet been installed, or that have been canceled by the user; the "Installed" tab shows recently installed updates to be viewed by last week, last 30



2 Updates Available

X

OS X 10.9.2

Critical – Security Update

☆☆☆☆☆

Size: (522 Mb)

- Improves the reliability of VPN connections that use the USB smart cards for authentication
- Improves the reliability of accessing files located on a SMB server
- Includes Safari 7.0.6

Source ✓

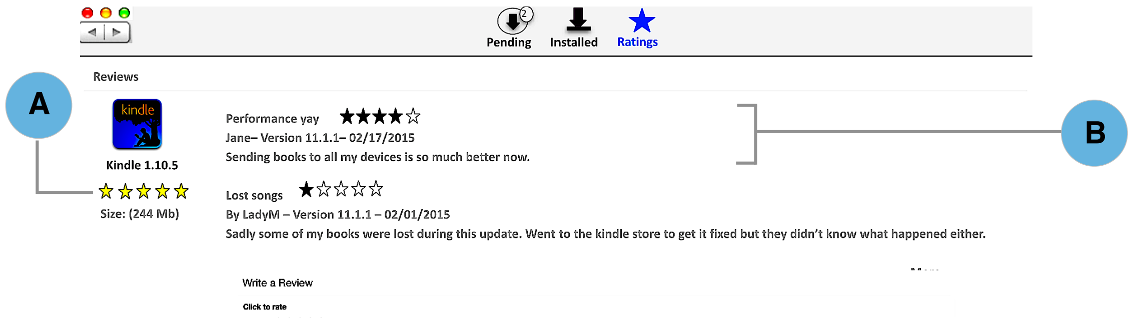
Estimated time for installation: 10 minutes

Compatibility report

Incompatible App Updates

Released Jan 8, 2015

Figure 3.3: The Central Update Manager's "Pending Updates" Tab. A: Icon Showing No Of Pending Updates. B: Summary of the Update Information. C: Change Log. D: Update Preference Settings. E: Estimated Installation Time.



Reviews

Kindle 1.10.5

☆☆☆☆☆

Size: (244 Mb)

Performance yay ★★★★☆

Jane– Version 11.1.1– 02/17/2015

Sending books to all my devices is so much better now.

Lost songs ★☆☆☆☆

By LadyM – Version 11.1.1 – 02/01/2015

Sadly some of my books were lost during this update. Went to the kindle store to get it fixed but they didn't know what happened either.

Write a Review

Click to rate

Figure 3.4: The Central Update Manager's "Ratings" Tab. A: Update Overall Rating. B: Update Rating and Reviews.

days, or all time; and finally, the "Ratings" tab shown in Figure 3.4 shows review comments, update ratings, and allow users to add their own reviews. Updates display a:

1. *Change Log:* Each update has a change log in bullet-point form clearly listing the changes the update makes as seen in Figure 3.3C.

2. *Time to Install:* We show the estimated time (Figure 3.3E) to install an update

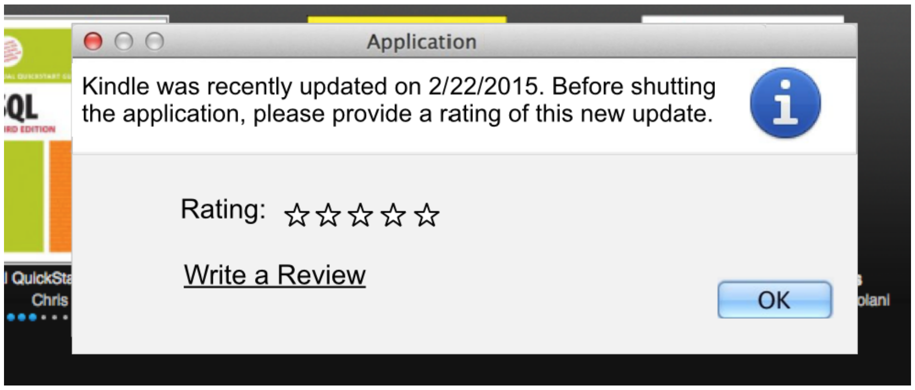
to help users plan when to do an update.

3. *Compatibility Report:* All the known possible disruptions an update might cause are listed in a report (not shown).

4. *Update Settings:* Users can reconfigure updating settings—silent, automatic, or manual—for every application or the operating system from the central update manager (Figure 3.3D).

Comparison to current Mac OS X Updating System: The current Mac OS X operating system notifies users about an incoming update by means of two notifications (a pop-up, and an red call-out on the App Store icon) when updates are requested manually. No explicit notifications are provided when updates are downloaded and installed automatically. The prototype switches all updates to silent, provides ambient notifications via an update notification icon, and does not seek users' consent to update by default.

We based the central manager design on the current Mac OS X App Store interface, which handles updates for all App Store applications and the operating system only but not third party applications. This version of the manager adds to the information the current App Store displays, with a clear and cohesive description of the change log in bullet form, and the update's type, size and ratings, along with an estimate of the installation time and compatibility report. Unlike the current App Store, this version of the update manager includes an update configuration—manual, automatic or silent—for each application.


Application

Kindle was recently updated on 2/22/2015. Before shutting the application, please provide a rating of this new update.

Figure 3.5: In-Application Post-Update Feedback Dialog.

Post-update, the current Mac OS X operating system notifies users about an installed update by means of placing a tiny blue dot next to the application icon in the app "Launcher" menu. The prototype, on the other hand, presents users with a dialog to notify them an update has taken place and to solicit a rating.

To sum up, we designed the proof of concept prototype to minimize interruptions, augment the update information available to users, and to centralize update management across a device. The design was purposefully extreme in nature—in this case, having *all* updates as silent and having *all* applications solicit feedback post update, providing users with *all* the necessary information—much like a breaching experiment [47] to elicit user reactions and feedback.

3.5 Phase Three: Prototype Evaluation

In Phase Three, I, and the two other researchers, evaluated the low fidelity prototype. Although software updating, much like other security tasks, is a secondary task, the objective with evaluating the prototype was to elicit users' reactions and feedback on the design concepts and features.

3.5.1 Procedure

To evaluate the prototype, we recruited 22 adult Mac OS X users via advertisements on the UMD's mailing lists, social media (Facebook, Twitter), and from users who participated in Phase One of the study. Each participant completed a pre-study demographic survey before participating in the think-aloud session, a technique employed regularly in usability testing to gather feedback on proof of concept designs. Specifically, we employed the "speech communication" think-aloud protocol by Boren and Ramey [48], using verbal communication only as a means to acknowledge user response and keep the thought process alive [49].

Each participant performed a series of 11 tasks with the low-fidelity interactive prototype on a laptop provided by the research team. The tasks were designed to elicit user interaction with the single update icon, the central manager, and an application post-update. The first task was to search for updates on the machine to observe whether participants could find the update notification icon on their task bar. When a participant located and clicked on the icon, they were shown a list of updates. Next, participants were asked to cancel an update and following this task, they were asked to check for other pending updates on the machine and to verify whether the source of an update was genuine.

Participants were then introduced to the central update manager through a task to find out more about a particular update which linked them to the manager. In this view, they had tasks that asked them to view the specific additional information provided about updates such as the time to install and the compatibility report. Participants were also asked to do tasks that allowed them to see an empty list of pending updates and the installed and pending tabs in the central update manager. Finally, participants were tasked with rating and reviewing an update as well as changing the update setting for the operating system. The list of think-aloud tasks available in Appendix A.

Participants were instructed at the beginning of the session how to provide feedback, how the session would be recorded, and how they were to proceed through the tasks with the guidance of two facilitators from the research team. They were then presented with a paper list of tasks and two researchers observed the participants as they interacted with the prototype, recording their thoughts and actions as they completed the tasks. The researchers used probes such as "keep talking" and "um hmm" to remind participants to verbalize their thoughts when they stopped talking. When the participants failed to speak for more than 30 seconds, one of the researchers asked a stronger probing question relating to the task at hand. For instance, when participants failed to locate cancelled updates, the researcher asked "Where would you expect to see these updates and why?". For each participant, task responses that did not require these strong probing questions were marked "successfully completed".

Once the think-aloud session was complete, the researchers conducted an semi-structured exit interview with each participant. In this interview, we first explained the silent updating mechanism to the participants, i.e., how updates would install on their devices and how restarts would function. We then asked them about both their positive and negative reactions to the prototype, how they performed each task, and the design concepts embodied by the prototype. We also sought feedback for those tasks that the participants were unable to complete noting down how they expected the system to function. Each session lasted between 45 minutes to 1 hour and both the think-aloud session and exit interview were audio and video-taped. Participants were compensated with a USD 15 gift card for the entire session. The study was approved by the IRB of the UMD.

3.5.2 Data Analysis

Once the think-aloud sessions and exit interviews were transcribed, two researchers—including one from Phase One—independently analyzed the data. We created profiles for each participant and noted their completion of the various think-aloud tasks. We also analyzed the transcripts using the same process as for Phase One—specifically seeking for differences in participants' reactions to the various design elements. In the following section, I use the prefix *T* to indicate a think-aloud participant.

3.5.3 Participants

Table 3.2 summarizes the demographics of the think-aloud session participants. Almost all of the participants were aged between 18 and 35. About 85%

of them had completed their bachelor's degree or college, and either worked at, or near, or studied at the UMD. Since many of the participants were students, the median annual income reported was <USD 25,000. The participants owned a median number of 2 Internet-enabled devices and went online more frequently on laptops and smartphones than desktops, tablets, and gaming devices.

Similar to Phase One, the participants were aware of security breaches that were heavily publicized in the media—17/22 knew about the Heartbleed bug and 16/22 about the 2013 Target breach. Again, one-third of the participants had experienced either viruses or malware on their systems. Unlike Phase One, none of the participants reported extra measures to enhance their email security but claimed to ignore email attachments if the email was sent from an unknown source (15/22) or if it looked suspicious (18/22). Overall, the participants were younger, less educated, and less technical than the participants from Phase One.

3.6 Findings

The prototype elicited varying reactions from the participants in the think-aloud session. The participants wanted to be notified and actively consent to some but not all updates, they were positive about augmented update information as input for update related decision making, and appreciated the control of centralizing software update management.

3.6.1 Interruptions Sometimes Required For Control

Participants struggled to find the update icon—half failed to notice the system tray icon entirely. However, once participants discovered the icon, they were able to complete the remaining think-aloud tasks with ease. About half preferred the design to current software updating interfaces because they believed it would interrupt them less. T20 explained: *"It prompts me the least. I don't have to worry about it, I don't have to think about it."* The other half of the participants reacted negatively to the prototype's silent update mechanism and told us they wanted to be in the update process more actively. These participants' wanted updates with notifications at download and install time, particularly for applications they used frequently or depended on in some way, to prevent undesired changes. In an example quote, T8 said: *"I want to know how frequently the updates are, how frequently they're occurring and if there's something new or there's a bug. If there any changes, I want to know when and how they happened."*

When asked for further feedback, participants revealed they were willing to adopt a silent updating mechanism for a few of their applications. For instance, participants were amenable to silent updates from trusted vendors and for applications that did not impact their workflow significantly. T9 said: *"I definitely prefer the silent so I wouldn't really have to do anything and to constantly be the best experience that I could have."* They believed a silent update process would keep their system secure and up to date since they often ignored updates otherwise.

3.6.2 Users Desire Improved Update Information

When completing tasks to read and interact with the additional update information we provided, participants were generally positive that the information would help them to make decisions about updates more so than current interfaces.

Update Type: 15/22 participants reacted positively to the update type, telling us these labels would clarify the purpose of various updates. However, about a third said the update type would not influence their decision to apply an update. Others preferred the binary classification of "critical" and "non-critical" to multiple labels, which they felt could be overwhelming and possibly confusing. Poor labels, participants told us, could backfire and cause a user to avoid an update if they obscured the update's purpose. In all, participants appreciated the concept of more informative labels for the update type to help in their updating decision making process.

Compatibility Report: 19/22 participants were able to find and interpret the compatibility report and 12/22 told us it would be helpful for singling out potentially problematic updates. In a typical example, T8 said: *"If I'm going to be using those apps, I wouldn't go ahead with the update because it will cause problems."* One suggested improvement was to only display compatible updates and only one participant worried how this report would be implemented. Overall, participants desired this predictive capability to help them prevent updates having an unwanted ripple effect.

Ratings: 15/22 reacted positively to the concept of update ratings with 12/22 saying that ratings would potentially influence their decision in going ahead with an update (especially with large updates). T9 explained: *"If there were mostly good reviews and there was nothing standing out as 'oh, this is a problem for my computer' it would help me make the decision to go ahead."* At least five participants wanted to see number of ratings for an update to better contextualize the information. Almost a third of participants felt that ratings would not add any value to their updating experience because they paid less attention to others' opinions. Most of the participants resonated with the idea of leveraging social networks for information to help them decide about performing updates.

Post-Update Feedback: All the participants were unanimous in disliking the concept of providing a mandatory rating post-update, seeing this as a nuisance in the long run. However, they were willing to provide feedback for updates that made visible changes (e.g. user interface modifications), and for applications that they frequently used or were important to them.

Time to Install: 13/22 found that having the information about how long was needed to install an update prior to beginning the process was useful for deciding when to do an update. T8 said: *"I think that's very important because if you're in a hurry or if you have some other work to do and sometimes you should know if you can finish the update by then or not."* 1 participant wanted aggregated view of the time required for all the pending updates and another wanted the time to install to be visible in the list of updates not just the central manager. For participants, having an estimate of the time involved for the update process was crucial for planning so as to minimize interruption to their activities.

Installation Size: Four participants reacted positively to the size of the update being displayed upfront. These participants desired some warning if the update would consume their remaining disk space. Two participants felt this information was less useful as they cared less about disk space on their devices.

Source Verification: Nine participants reacted positively to having the ability to easily identify an authentic update source. In an illustrative example, T17 remarked: *"If it's not from the verified source, then that will be the one thing that will stop me from installing the update."* However, several said they would probably not pay attention to this cue and a few felt that this cue was most important for third-party applications only. It was clear, however, that visual cues indicating update authenticity can build trust with users.

3.6.3 Users Prefer Centralized Update Management

Over half of the participants reacted positively to centralized software update management, especially for non-Mac applications that currently do not push updates via the app store. For example, T13 said: *"I like it: It seems more comprehensive because it has (for e.g.) the Microsoft stuff in it so you don't have to run the Microsoft updater as well as the app store updater mechanism."* Only a few participants thought that being able to control the update preference on a per-application basis in a central update manager was useful. However, it was unclear if participants reacted this way to the additional controls because they told us they generally preferred to keep default settings. Participants suggested the central update manager could also provide a history of updates so that changes could be rolled back to a state before the update occurred if something went wrong. In summary, participants felt managing updates in a single location would reduce information overload and make the updating process more consistent across a device.

3.7 Discussion

Users do not apply updates on their desktops for a variety of reasons. While on one hand, they dislike that updates interrupt them, they wish to retain control and explicitly provide consent to major updates, or applications that they use frequently. This provides an opportunity to personalize updates, where certain updates can be automated whereas for other updates users can be notified based on their preferences towards auto-updating. At the same time, when users are interrupted, it is important to provide them with information that is actionable, and addresses their beliefs about software updates. Finally, centralizing software updates across the desktop with an option to roll back updates may help users keep track of how updates are taking place across their applications, and also provide them with an option to try out updates updates with minimum consequences.

3.8 Limitations

Study One has several limitations. First, the use of self-reported data in Phase One is subject to recall bias, meaning there may exist errors and differences in the recollections recalled by the participants. Second, because of the low fidelity nature of the prototype, its evaluation in Phase Three is based on participants' opinions rather than their actual behaviors. The evaluation also required participants to work with updates as a primary task and thus, the results may not generalize to real-world settings where software updating is considered a secondary task. Third, the participant pools were dominated by a younger set of participants, with many participants in Phase One having advanced degrees, and many students in Phase Three. Therefore, the results from both the studies may not generalize to the entire population, and hold limited validity.

3.9 Chapter Summary

We used a three phase approach to improve the user experience of software updating on desktops. In Phase One, we first investigated why users avoid updates on their desktop machines. We then used this information to design a *minimally-intrusive*, *information-rich*, and *user-centric* prototype in Phase Two, and subsequently evaluated the prototype with participants in a think-aloud study in Phase Three. Overall, we found that users avoid installing updates because they feel updates interrupt them, that software updating is an information problem, i.e., users often lack enough information to go forward with an update, and that users vary in their preferences about how they wish to be notified about updates. From the evaluation of the prototype, we learned that users vary in terms of which updates they wish to auto-update, that they appreciated the improved update information, and liked the idea of centralizing updates across their machines.

Chapter 4: Study Two: Automating Updates on Mobile

4.1 Overview

In Study One, I found that desktop updating interfaces can better support update automation by interrupting users when necessary (e.g., with a large update, or for important and frequently used applications), enhancing information about the update, and by centralizing update management across a device or system. Mobile devices, such as Android or iOS smartphones, have already implemented these three design suggestions: they have reduced the interruptive nature of update-associated notifications by means of streamlining update requests and eliminating device restarts, they have centralized most available updates by means of the App Store, and they have provisioned the ability for software developers to provide information about updates in the App Store. Given these improvements, I set forth to explore why people install updates manually on mobile, and what changes, if any, can be made to the Android interface to help users update automatically. Limiting the domain to the Android Operating System, in Study Two, I conducted a large scale survey and qualitative interviews with Android users to understand why users chose to update their applications manually, and their preferences towards auto-updating their applications.

4.2 The Android Play Store Update Process

The Android Play Store allows applications to be updated either automatically or manually; the former can be restricted to times when the phone is connected to a Wi-Fi connection only. In addition, Android allows certain applications to be updated manually when in automatic mode. The operating system updates are delivered through a separate channel, often times through the cellular carrier or device manufacturer Over-the-Air (OTA).

If users choose to install updates manually, they see an update notification indicating the applications with available updates. Users also see a notification when after an application is updated indicating the updated applications, independent of the update mechanism. All of these notifications can be toggled from the Play Store.

Updates containing an addition of permissions requested by the application cannot be automated. However, this has changed since Android 6.0 permissions are requested at run-time [50].

4.3 Procedure

In the following section, I describe the method I employed, along with details of the surveys and interviews, and how I analyzed the resulting data. Both the survey and the interview transcript are available in their entirety in Appendix B. The study was approved by the IRB of the UMD.

4.3.1 Research Questions and Hypotheses

To investigate individual differences in software updating on mobile devices, I built upon previous research from Egelman and Peer [32, 51]. While these authors demonstrated that keeping software updated is associated with taking fewer risks (DoSpeRT Ethical, Health/Safety), being inherently curious and inquisitive (NFC), and thinking about long-term implications of actions (CFC), I explore how these personality traits correlate with how users install updates: manually or automatically on mobile devices. To this end, I included another trait: how averse users are to change, measured using the Resistance to Change (RTC) scale [52]. Consequently, I formulated the following hypotheses:

1. H1: Users who update their applications manually are less risk taking than those who update their applications automatically because they are concerned updates might lead to unintended consequences.

2. H2: Users who update their applications manually consider the future consequences of their actions less than those who update their applications automatically because they are concerned updates might lead to unintended consequences in the long-term.

3. H3: Users who update their applications manually are more curious and inquisitive than those who update their applications automatically because they have a tendency to keep appraised of the changes updates make.

4. H4: Users who update their applications manually are more resistant to change

than those who update their applications automatically because they seek routines in their lives and want control over the changes updates make.

5. H5: Users who update their applications manually have had a previous negative experience with updating and they, therefore, prefer to take action by themselves to avoid negative experiences.

While these hypothesis reveal differences between users who install their updates manually and automatically, I also investigated how comfortable users are automating updates across their different applications, and what factors limit them from automating the updates. To answer these questions, I conducted a large scale survey and interviews with Android users, and using those, draw design suggestions for mobile update interfaces.

4.3.2 Survey Instrument

The survey instrument contained three parts. The first part included the various psychometric scales namely, the DoSpeRT scale, the RTC scale, the CFC scale, the NFC scale, and the SeBIS as a covariate. The questions in each scale and the order of the scales were randomized to avoid order bias. Given that users may have poor mental models of how updates on their systems work [6], I chose to elicit their update settings using detailed instructions as opposed to asking them directly. Thus, the second part of the survey asked participants to report their update settings by following a set of labelled instructions leading to the auto-update setting on the Play Store. Following that, users were asked to select the applications installed on their devices from a visual grid containing the most installed Android applications of all time [53]. For each application, participants then rated their level of comfort with automating security and non-security updates on a scale of Uncomfortable (0) to Comfortable (100) using a visual slider. To limit the number of applications participants answered these questions about, I randomly drew 10 applications from the participants' selection. I designed the survey to elicit preferences in this manner—for each application, rather than an aggregate preference score towards security and non-security updates—to gather how participants varied in their responses. The third part of the survey asked participants whether they had previously had a negative experience with software updates i.e., if they had regretted updating any device or software. This part always appeared in order—right before the end of the survey—to avoid sensitizing users in their response to the question about their preferences towards automating updates in the second part. The survey ended with demographic questions relating to age, gender, income and occupation, andtook roughly 15 minutes to complete.

I piloted the survey with a 6 participant convenience sample drawn from within the Human-Computer Interaction Lab (HCIL) at the UMD to ensure the questions and instructions were clear and concise; a few questions were revised and refined consequently. The survey was hosted on SurveyGizmo[1], and advertised as an "Android Apps Update Survey" task on Amazon Mechanical Turk (AMT). Turkers were invited to participate if their primary smartphone was an Android device and if they had previously used the Play Store for at least one month to ensure familiarity with

[1]http://surveygizmo.com

the Android OS and how applications are installed/updated. To ensure response quality, I restricted the task to Turkers based in the US who had an approval rating of 95% or higher. Because I did not filter Turkers based on their productivity (i.e., number of previous tasks completed), I added three attention check questions to the survey based on the recommendation of Peer *et al.* [54], one of which specifically asked about the Android OS. I filtered all responses that failed any of the attention check questions. Turkers were compensated $2.50 for completing the survey.

4.3.3 Interview Guide

With the interviews, I aimed to further gain insight into how Android users updated their applications, and what other factors limited them from automating updates on mobile devices. The interview was semi-structured in nature, and sections of it varied depending on how participants updated the applications on their devices. First, I asked all participants for the reason behind their choice of updating: automatic or manual. I asked participants who updated applications manually on their mobile device to walk through the update process, elaborating specifically on how they reacted they to update notifications, how quickly they updated after receiving an update notification, and how they chose which applications to update. I asked participants who updated applications automatically on their mobile device for their reactions to post-update notifications, and if they had ever changed the update settings on their devices. Next, I asked participants to show me their Android device and together, we examined the list of pending application updates. I asked participants why they updated certain applications over others, and how they felt about the updates that were recently installed on their devices.

I recruited participants through advertisements on the UMD's and affiliated mailing lists around the US, and social media (Facebook, Twitter) posts. Participants were required to own an Android phone, and have previously used the Google Play Store for at least one month to ensure familiarity with the Android OS. The recruited participants were invited to the HCIL at the UMD, where I conducted and led the interviews. These interviews lasted for 35-40 minutes, and were audio-taped. The participants were paid \$20 for their time.

4.4 Survey Data Analysis

The survey received 525 responses in total from AMT, out of which 48 responses (9.14%) failed at least one attention check question and were subsequently discarded.

4.4.1 Reliability and Validity of the Psychometric Scales

I analyzed the psychometric data both in terms of reliability and validity. Reliability indicates how consistent a given scale or test is when administered repeatedly. I measured reliability using Cronbach's alpha (α) for which, a value of 0.7 of greater is considered acceptable, while a value greater than 0.8 is considered good [55]. Validity indicates the accuracy of a measurement, i.e., whether the scale or test truly measures what it was intended to measure. I measured validity through a Confirmatory Factor Analysis using Principal Component Analysis (PCA) [56] and applying the rotation method used originally by the authors of the psychometric scales.

Starting with the DoSpeRT scale, a PCA with promax rotation revealed all the six original factors had eigenvalues greater than 1, and explained 56% of the variation in the data. However, all the items in the Health/Safety subscale loaded on the Ethical subscale, similar to what the authors observed in their subsequent research with the scale [57]. For this reason, I combined both the subscales to measure one Ethical risk taking score. The final set of factors were: Ethical ($\alpha = 0.82$), Financial/Investment ($\alpha = 0.80$), Financial/Gambling ($\alpha = 0.90$), Recreational ($\alpha = 0.83$), and Social ($\alpha = 0.78$).

With the RTC scale, a PCA with promax rotation revealed three of the four original factors had eigenvalues greater than 1, and explained 63% of the variance in the data. However, all of the items in the Short-term Focus scale loaded on the Emotional Reaction scale. The high correlation between the Emotional Reaction and Short-Term Focus subscales has been explained by virtue of both dimensions being affective in nature [52]. Because the fit of the three-factor model was better than the originally described four-factor model, I combined Emotional Reaction and Short-term Focus into an Affective Factor, as explored in [58]. The final set of factors were: Affective Factor ($\alpha = 0.92$), Routine Seeking ($\alpha = 0.81$), and Cognitive Rigidity ($\alpha = 0.70$).

Next, I examined the uni-dimensional CFC and the NFC scales. Both these scales revealed high reliability: $\alpha = 0.9$ and $\alpha = 0.95$ respectively.

Finally, with the covariate, SeBIS, a PCA with varimax rotation revealed the original four factors all had eigenvalues greater than 1, and explained 59% of the variance in the data. The four factors were: Proactive Awareness ($\alpha = 0.72$), Password Generation ($\alpha = 0.78$), Device Securement ($\alpha = 0.80$), and Device Updating ($\alpha = 0.70$). I, therefore, concluded that the data was both reliable and valid, and proceeded to construct the regression model.

4.4.2 Update Type Logistic Regression Model

Next, I constructed a logistic regression using the glm() from the "stats" package in R, where I regressed the update type ($manual = 1$ vs $automatic = 0$) on the psychometrics (DoSpeRT, RTC, CFC, NFC), the covariates (SeBIS, Age, Gender, Education), and the presence of a previous negative experience with updating software (Neg. Experience). To reduce the number of levels in Education, I considered it as a continuous variable using the following scheme: No High School:1, (High School Graduate, Some College):2, (Bachelor's Degree, Associate's Degree):3, (Master's Degree, Doctoral Degree, Professional Degree):4.

4.4.3 Preferences Towards Automatic Updating

Finally, to better understand participants' preferences, I first filtered all those participants who contributed the automatic update comfort scores for fewer than 5 applications since it would be difficult to categorize participants if little is known about their preferences. This reduced the number of participants to 460. Next, I computed the median security and non-security auto-update comfort scores for each participant, and treated each participant's scores as a pair. I then clustered these preference pairs using the K-Means clustering algorithm [59] implemented in R as the kmeans() function in the "stats" package. This clustering technique partitions the participants into K groups, where each participant is assigned to the group to minimize the within-cluster sum of squares. To examine differences towards security and non-security update automation within each cluster, I conducted a one-sample two-tailed Wilcoxon Signed-Rank test, accounting for multiple comparisons using the Bonferroni Correction. This non-parametric test evaluates the null hypothesis that the true difference between the level of comfort with automating security and non-security updates is 0 (no difference), and the alternative hypothesis that the true difference is non-zero.

4.5 Interview Data Analysis

I recruited 13 participants in total. Once the interviews were transcribed, I analyzed the transcripts, inductively looked for patterns and threads in the data, marking them with labels, grouping and organizing these labels into themes [39]. During the analysis, I also specifically examined for differences between the participants who updated their applications in an automatic and manual fashion on their mobile devices. In the following sections, I use the prefix R to indicate an interview participant.

4.6 Participants

Table 4.1 summarizes the survey and interview participant demographics. The survey participants were predominantly between 18–34 years old, with more males than females (62.3% vs 37.1% respectively). Nearly 90% of the survey participants had either a college or bachelor's degree, and earned a median annual income between \$35,000 and \$49,999. 77% of the survey participants updated their applications automatically, and about half of the participants updated at least one application manually. 39% of survey the participants had regretted or had a previous negative experience with updating their software.

The interview participants were predominantly between 18–34 years old as well, with more males than females (76.9% vs 23.1% respectively). Compared to the survey participants, more interview participants had master's degrees (30.8%), and earned a median annual income between \$50,000 and \$74,999. 69% of the interview participants updated their applications automatically.

4.7 Findings

Overall, my study showed that users who update their applications manually are less risk taking compared to users who auto-update, that they are more proactively aware when it comes to their security (e.g., fixing security problems by themselves, verifying links before they visit them), and have previously had, or can recall a negative experience with updating their software. Users preferred automatically

Demographic	**Survey Participants (N = 477)**	**Interviews Participants (N = 13)**
Age		
18–34	69.2%	84.6%
35–54	28.9%	15.4%
>55	1.9%	0%
Gender		
Male	62.3%	76.9%
Female	37.1%	23.1%
Other	0.6%	0%
Education		
College	45.4%	7.7%
Bachelor's	45.6%	53.8%
Master's	8.4%	30.8%
Other	0.6%	7.7%

Table 4.1: Demographic Information: Survey and Interview Participants.

updating security updates over non-security updates, but update automation was limited by infrastructure constraints such as storage space. The following section describes these findings in further detail.

4.7.1 Differences between Auto and Manual Update Users

The first research question explored how individual differences in users contribute to how they choose to update their software on their mobile devices. The output of the logistic regression—where the dependent variable is the update method, and the independent variables are the psychometrics, demographics, and previous negative experiences with software updating—is shown in Table 4.2. A Likelihood Ratio Test (LRT) between the null and full models revealed a significant model fit (p = 0.0001) with a reduction in deviance of 45.32. The Likelihood Ratio (R^2) Effect Size of the model was 0.088.

Predictor	Estimate	Std. Error	Odds Ratio [95% C.I.]
(Intercept)	0.15	1.35	1.16 [0.08, 16.29]
CFC	−0.34	0.21	0.71 [0.47, 1.07]
NFC	0.02	0.16	1.02 [0.75, 1.39]
DoSpeRT-Ethical	−0.45	0.17	0.64 [0.46, 0.89] **
DoSpeRT-Social	−0.10	0.12	0.90 [0.72, 1.14]
DoSpeRT-Recreational	0.18	0.11	1.20 [0.97, 1.48]
DoSpeRT-Investment	−0.24	0.09	0.79 [0.66, 0.94] **
DoSpeRT-Gambling	0.08	0.11	1.08 [0.86, 1.34]
RTC-Affective Factor	0.11	0.16	1.12 [0.82, 1.54]
RTC-Routine Seeking	−0.03	0.18	0.97 [0.68. 1.38]
RTC-Cognitive Rigidity	−0.01	0.14	0.99 [0.75, 1.32]
Neg. Experience [Yes]	1.04	0.24	2.82 [1.76, 4.57] ***
SeBIS-Proactive Awareness	0.35	0.17	1.42 [1.01, 2.00] *
SeBIS-Password Generation	0.08	0.16	1.08 [0.79, 1.48]
SeBIS-Device Updating	−0.13	0.16	0.88 [0.65, 1.19]
SeBIS-Device Securement	0.10	0.11	1.10 [0.89, 1.38]
Age	−0.01	0.02	0.99 [0.96, 1.02]
Education	−0.12	0.19	0.88 [0.61, 1.28]
Gender [Male]	0.10	0.26	1.10 [0.67, 1.85]
Null Deviance: 517.57 on 476 degrees of freedom			
Residual Deviance: 472.25 on 458 degrees of freedom			
Model Fit Likelihood Ratio Test: Deviance = 45.32, p <0.0001			
Likelihood Ratio (R^2) Effect Size: 0.088			

Table 4.2: Results of the Logistic Regression Modelling the Outcome (Updating Manually) on the Various Predictors. 95% C.I. is the 95% Confidence Interval. $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

Risk Taking: The output shows that the coefficients of DoSpeRT-Ethical ($\beta = -0.45$, $e^\beta = 0.64$, $p < 0.01$) and DoSpeRT-Investment ($\beta = -0.24$, $e^\beta = 0.79$, $p < 0.01$) were both significant. This means that for every point increase on the DoSpeRT-Ethical (which measures ethical risk taking, e.g., passing off somebody elses work as your own) and DoSpeRT-Investment (which measures financial investment risk taking, e.g., investing 10% of your annual income in a moderate growth mutual fund) scales, the odds of a user updating his/her applications manually decreased multiplicatively by 0.64 and 0.79 respectively, holding all else constant.

Negative Experience	**Frequency (N = 477)**
The old version of the software functioned better	36.4%
The update introduced new bugs in the software	34.3%
The update changed the user interface of the software	27.6%
The update took a long time to install	11.3%
The update used up a lot of data	10.7%

Table 4.3: Negative Experiences Reported by the Survey Participants.

This result lends evidence to support hypothesis H1 that users who update their applications manually tend to be low risk takers.

Previous Negative Experience: The output also shows that the coefficient of Negative Experience was significant ($\beta = 1.04$, $e^{\beta} = 2.82$, $p < 0.001$). This means that having had a previous negative experience with software updating increased the odds of a user updating his/her application manually by 2.82, holding all else constant. This result lends evidence to support hypothesis H5, that users who have had a previous negative experience preferred to take action and update their applications manually to avoid future problems.

The survey participants reported a variety of reasons about their negative experiences with software updating—similar to the reasons I heard in Study One and as reported in previous studies [5, 28, 29]—such as updates caused the software to be buggy, made the user interface of the software uncomfortable to use, and took a long time to install. Table 4.3 lists the negative experiences reported by the survey participants along with the frequency of their appearance.

Proactive Awareness: The coefficient of SeBIS-Proactive Awareness was significant ($\beta = 0.35$, $e^{\beta} = 1.42$, $p < 0.05$) as well. This means that that for every point increase on the SeBIS-Proactive Awareness scale, the odds of a user updating his/her applications manually increased multiplicatively by 1.42, holding all else constant. This results suggests that users who update manually also tended to take proactive security action such as verifying links before opening them, ensuring the green HTTPS lock is visible on a website before submitting information, and fixing security problems by themselves rather than depending on others.

I could not find any evidence to support hypotheses H2, H3, and H4.

4.7.2 Users Prefer Automating Security Updates

The second research question explored how users varied in their preferences toward automating security and non-security updates on their mobile devices. After computing the median level of comfort towards automating security and non-security updates across the applications for each participant, I examined a plot of the percentage of variance explained by the number of cluster to arrive at the number of clusters. The "elbow" of the curve appeared at $K = 5$, and hinted at the presence of 5 clusters. These clusters, when extracted, explained 88.6% of the variation in the data. Table 4.4 shows the results of running the K-Means algorithm.

Figure 4.1 presents a visual representation of the clusters, where the X-axis and Y-axis represents participants' comfort with automating security and non-security updates respectively. Cluster 1 and cluster 5 contain participants at the extremes of the preference spectrum. Participants in cluster 5 ($N = 37$) are not comfortable automating either security or non-security updates ($\mu_{security} = 7.9, \mu_{non-security} = 6.4$), and prefer control over what gets updated on their devices. On the other hand,

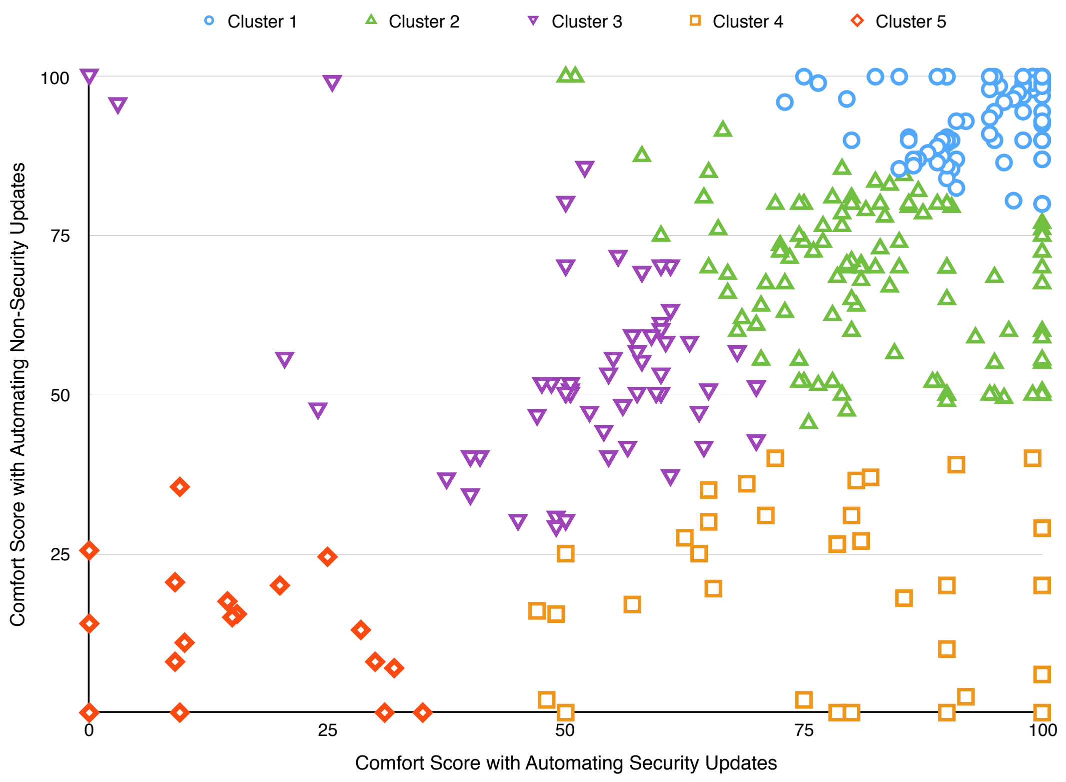


Figure 4.1: Visual Representation of the Cluster Resulting from the K-Means Clustering of the Survey Participants.

participants in cluster 1 ($N = 206$) are comfortable automating both security and non-security updates ($\mu_{security} = 97.0, \mu_{non-security} = 96.8$).

All the other clusters of participants lie in between the preference spectrum. Cluster 3 ($N = 69$) consists of participants who are indifferent towards automating both kind of updates, i.e., they do not feel strongly one way or the other ($\mu_{security} = 51.1, \mu_{non-security} = 52.9$). Clusters 2 ($N = 114$) and 4 ($N = 34$) both consist of participants who are exhibit strong a preference towards automating security updates over non-security updates. A Wilcoxon Signed-Rank Test revealed a statistically significant between security and non-security updates for both these clusters ($p < 0.001$). Participants in cluster 2 ($\mu_{security} = 82.1, \mu_{non-security} = 68.5$), however exhibited a higher preference for automating non-security updates than participants in cluster 4 ($\mu_{security} = 76.7, \mu_{non-security} = 19.5$).

Overall, participants in clusters 1, 2 and 4—consisting of 77% of the total participants—displayed a strong preference towards automating security updates.

Cluster	**Cluster Size**	**Comfort Automating [Security, Non-Security] Updates**	**Preference?**
1	44.8%	[97.0, 96.8]	Neither
2	24.8%	[82.1, 68.5]	Security ***
3	15%	[51.1, 52.9]	Neither
4	7.4%	[76.7, 19.5]	Security ***
5	8.0%	[7.9, 6.4]	Neither

Table 4.4: K-Means Clustering of the Survey Participants based on their Comfort Automating Security and Non-Security Updates using $K = 5$. The Values in [X, Y] are the Means Calculated for Security and Non-Security Update Automation for each Cluster. The “Preference” Column Indicates Which Update Type was Preferred. To Account For Repeated Testing, only p-values < 0.01 are Reported for the Wilcoxon Signed-Rank Test. $^{***}p < 0.001$, $^{**}p < 0.01$.

4.7.3 Infrastructure Limits Update Automation

I conducted interviews with 13 Android users to further investigate what limits automating updates on mobile devices. The primary theme that emerged from the analysis was that automatically updating applications on mobile devices is severely limited by infrastructure constraints such as available storage memory. About half of the participants (6/13) reported their mobile devices could no longer sustain updates automatically because the internal memory was either exhausted or close to exhaustion. As a result, these 6 participants either picked and chose which applications to update (4/6), or simply stopped updating altogether (2/6).

4 of the 6 participants who picked and chose which applications to update, updated their applications manually. Because these participants were concerned about the degrading performance of their devices, they only updated applications that they frequently used or were important to them, or applications with major updates—similar to the preferences participants stated in Study One. Participant R6 explained: *"For example, if it's Facebook, then I wouldn't just update, because Facebook is huge, the space and everything, so I just wouldn't update it."* In another example, R5 explained: *"Then the problem comes that my phone starts maybe hanging if I keep the auto update on. Because as I said, I have an old phone so that's why the newer applications don't work well. That's why I have not checked the Auto Update on Wi-Fi option."* These participants often took steps to clean up their device, removing old applications, photos and videos they no longer needed.

2 of the 6 participants who stopped updating altogether had set their applications to auto-update but because their phones had no space to accommodate new applications and updates, the Android OS prompted them to update their applications. R9 remarked: *"Actually, I face this memory problem. There are many applications in my phone. Sometimes, an app like Facebook takes somewhere around 380 MB, which is a lot of space. It prompts me to update but I don't."*

Overall, automation was limited by the available storage space on users' devices, and this meant they either updated manually—relying on their judgement about picking and choosing updates—or stopped updating altogether.

4.8 Discussion

These findings have several implications for the design of mobile update user interfaces. My study show that users who take fewer risks (Financial/Investment or Ethical), are proactively aware about their security, or have had a previous negative experience with updating tend to update their applications manually. Because these users are low risk taking, they can be nudged to auto-update by emphasizing and framing a message around the Financial or Ethical risk that might result if applications are not auto-updated. As another example, the predisposition of a negative updating experience can be leveraged to nudge users to auto-update by explaining and guaranteeing how these negative experiences will not be repeated again. Alternately, the risk taking trait can be used to set defaults to automate certain kinds of updates that are less risky—such as security updates.

Indeed, as my study demonstrates, the majority of users are comfortable automating security updates, and these updates are a low-hanging fruit developers of update systems can exploit as candidates for auto-updating. This also supports the findings from Study One that, whenever possible, security updates must be decoupled from all other updates. However, at the same time, developers of software updates for mobile devices must also be cognizant of the constraints of memory and storage on older devices.

4.9 Limitations

Study Two has two major limitations. The first limitation is that it is unclear how my results from this study generalize to the general population, i.e., to what extent are the individual differences and preferences in software updating exhibited by the AMT population also exhibited by the general population. The second limitation relates to how I collected the update settings on Android as part of the survey. While I provided detailed instructions containing screenshots so participants could accurately report their update settings, the data still remains self-reported in nature, and therefore, likely to be less reliable. Future research could identify ways to collect this information from users' devices, without having them report it.

4.10 Chapter Summary

To explore how update interfaces on mobile devices can be improved to support automation, I first conducted a survey to examine individual differences amongst Android users towards automatic updating, and to elicit their preferences towards automating security and non-security updates. To further investigate what limits automation on mobile, I conducted semi-structured interviews with Android users. I found that users who applied updates manually tend to be low risk takers, that they are more proactively aware about their security, and are able to recall previous negative experiences with software updating. Users also displayed a strong preference towards automating security updates, but often times, automation was limited by the storage space available on users' devices.

Chapter 5: Discussion and Future Work

5.1 Discussion

Wash *et al.* [6] argued in their paper that increasing usability alone may not be enough to improve the updating experience because users it may enable users who want to be less secure to execute their actions. The findings of this thesis suggest that users may want to be less secure in the first place because their preferences are misaligned with how update systems currently function. Improving usability, therefore, should still be a goal to adapt automatic updates in a manner that encourages users to apply updates. This is particularly important as the Internet of Things evolves and users are faced not only with updating their personal devices but devices in their homes, office, on their bodies, and elsewhere. The findings suggest four primary directions to improve update automation: personalizing update interfaces, enhancing update information, centralizing desktop update management, and awareness of infrastructure constraints. I also outline considerations about the socio-technical aspects of the software updating process specifically around trust, control, and consent.

5.1.1 Personalizing Software Update Interfaces

My first recommendation is to personalize update interfaces so certain updates can be automated over others. Study One showed desktop users are comfortable automating updates for applications they use less frequently or depend less on, or for security updates. Study Two suggest the same, that mobile users who update applications manually are less risk taking, that they are more proactively aware of their security, and have had previous negative experiences with updating—and yet are comfortable automating security updates.

On both the desktop and mobile, personalization has the potential to reduce the number of update requests users receive when they prefer to retain control and provide consent to software updates. Study One and Two both suggest that users are comfortable automating security updates, and therefore wherever possible, developers of applications should decouple software updates from all other updates. Of course, this may only be possible if update systems facilitate the automation of only security updates in the first place. For instance, the current Android OS fails to distinguish between security updates and all other updates, providing only an option to automate all kinds of updates.

As a further extension, and perhaps a more lofty goal, personalization of update interfaces could occur in a similar way as others have suggested [60] in the domain of requesting Android permissions. In other words, how can we design update systems that that prompt users to obtain consent to updates only when absolutely necessary? Study One highlighted several factors that can be used to determine if updates can be automated including the frequency of use of applications over time (less frequently used applications can still be sources of vulnerabilities [18]), their importance, and various characteristics of the update (e.g., purpose, size, or installation time).

Study Two demonstrated how individual differences in users can be used to frame messaging to nudge users to auto-update. These nudges can emphasize the Ethical and Financial/Investment risk as a result of leaving auto-updates off. An Ethical risk centered nudge, for instance, can state: *"Switching auto-updates on will provide you the latest security updates, and keep you and others like you protected from exploits"*, or a Financial/Investment risk centered nudge can state: *"Switching auto-updates on will protect you from financial thefts"*. The timing of these nudges may be crucial, and can optimized to maximize compliance. As a example, users can be nudged right after they install an update rather than at a random time. Future research could test these hypotheses by the means of controlled experiments.

Further, when users are prompted, update systems should empower them with a "cancel" option to delay the update. This stands in contrast to current automated updating systems that delay updates but install them anyway if users fail to take action within a certain period of time. Giving users more power over their systems and applications may make them more likely to trust the updating process.

5.1.2 Enhancing Update Information

My second recommendation to improve updating interfaces on desktop and mobile revolves around better informing users about updates and their consequences. This information can help build trust in the update process, e.g., via compatibility reports and update ratings. Further research into how to generate compatibility reports and how update ratings can be gathered and provided will help users make informed choices about which updates to apply. Similarly, while the Android OS has streamlined the information source for updates—in the form of a "What's New" notice containing information about the update—augmenting existing information with guarantees about the updates or targeting their (mis)beliefs can help users make decisions. For instance, "social proof" has already been shown to improve security feature adoption in Facebook and update ratings could similarly help users assess if they should move forward with an update [61,62]. Other visual enhancements to show that update sources are verified or vetted could also instill trust and further motivate users to perform updates.

5.1.3 Centralize Desktop Update Management

My third recommendation for desktop updating interfaces specifically is to centralize update management where possible. Changes to current interfaces that would require operating system vendors to provide better ways for updates to be pushed through a single channel or for the information about updates to be gathered for a central update information repository across a device. Examples of applications that are already making strides in this vein include Metaquark's AppFresh [63] for centralizing Mac OS X updates and SparkleProject, a framework for third party application developers to push Mac OS X updates [64]. This model is already manifest on mobile devices where updates and their notifications already propagate more centrally than on the desktop through app stores.

The central update manager could further provide ways for users to preview the effects of an update on their system to mitigate the fact that users are averse to unwanted changes. For instance, users could be given the option to try out the new version of an application or operating system via an interface overlay or by using a parallel version of an application installed on a system without committing to the update process or applying the update. Providing an easier way to roll back unwanted changes may also make users more amenable to automate updates without fear of breaking their workflow and consequently, minimize negative experiences.

5.1.4 Awareness of Infrastructure Constraints

My fourth and final recommendation is that software developers should be cognizant and aware of infrastructure constraints that may emerge from continual updating. Study One and Study Two both revealed how constraints of disk space limit update automation for desktop and mobile devices—and in certain cases on mobile, made it impossible for users to update automatically. Developers need to be considerate about the size of the updates they publish so users can continue to receive at the very least, updates that can help them stay secure and safe from exploits and vulnerabilities. By decoupling and packaging security updates separately, we can ensure that they do not occupy large amounts of space on the device. Future work could also investigate how this affects users in developing countries, where such constraints on infrastructure may be more prevalent [26].

5.1.5 Socio-Technical Updating Aspects

The study and recommendations highlight the complex socio-technical nature of updates and the stakeholders involved. Updates involve trust between users and those seeking to make changes to their systems, gathering consent from users to make those changes, and surrendering control to external parties to make those changes. This involves a complex interplay of actors such as application and operating system vendors, developers, and users.

The question of whether these stakeholders will want to make the updating interface improvements we recommend possible remains open. For example, these recommendations depend on application developers modifying the information they provide about updates, and on the desktop, how updates are deployed to users, how users are notified about updates, and the potential consequences of applying any update. To centralize update management on the desktop, operating system vendors will have to build supporting frameworks to enable developers to push updates centrally and to have a vetting process similar to mobile systems for checking all updates and whether they comply with established guidelines for best informing users of upcoming changes.

Vendors might have large incentives to improve updating interfaces or otherwise risk alienating and losing their user base. For instance, a recent study of Tinder and Tesla updates [65], showed the backlash of unhappy users when unwanted changes were made without their consent to these apps. Yet, it is unclear who should have ultimate control over software changes, and whether there needs to be some governmental oversight for security purposes and consumer protection, particularly as some user bases grow as large as the population of several countries.

Furthermore, if update interfaces are indeed personalized, the question of transparency and accountability for selectively applying automatic updates is also open i.e., how would such systems explain their decision making to users in a comprehensible way and provide meaningful controls to users so that they can still provide consent for changes being made to their applications and systems. We will only know more about whether a personalized interface would make or break users' mental models of updates by testing these recommendations out in the wild.

In all, these socio-technical issues around software updating will require the Usable Security community to closely examine the practice of software updating from all viewpoints. We can certainly empower users through improved software updating interfaces but we need to better understand all the nuances of control, consent, and trust in updates. We should also be asking ourselves the question of who should be allowed to make changes to systems to properly address the issue of improving security through updates.

5.2 Limitations and Future Work

While both Study One and Study Two reveal insights into users' preferences about automatic updating, and offer design suggestions to appropriate automatic updating interfaces, these pieces of information and design still need to be validated in real-world contexts. These contexts are particularly important to prove the external validity of the design suggestions in the real-world, where software updating is a secondary and computer maintenance task. Future work for desktop updating systems could repeat Study One with higher fidelity versions of the prototype, and validate the user interface suggestions against actual users behavior. Similarly, future work in designing mobile interfaces could test out the various hypotheses for nudges relating to risk taking and previous negative experiences. Furthermore, since Study One consisted of advanced users, and Study Two consisted of AMT users, future work could replicate the results to the more general population. Finally, given that updating involves an ecology of stakeholders, future work could examine updating practices from other perspectives such as how network administrators manage updates for large groups of users or how developers create updates in the first place.

5.3 Conclusion

Different users have different needs and expectations with their update systems. Some users prefer control over how and when updates take place, while others are comfortable automating all their updates. However, both categories of users need to apply updates at the earliest, particularly when dealing with security updates. This thesis presented two studies—one on the desktop and the other on mobile—investigating how auto-update user interfaces can be designed to incorporate and align with these preferences. Study One showed that desktop interfaces can be improved by minimizing update interruptions, improving update information, and centralizing updates across the device. Study Two showed that mobile users can be nudged to auto-update by leveraging differences in their personalities and preferences. Both studies provided suggestions to personalize automatic updates based on various criteria including characteristics of the update, the application, and the user. Crucially, they suggest that automatic updating can be improved if users have all the necessary information about the updates, and a path to test and rollback from updates when necessary. Finally, these studies suggest that the socio-technological aspects of updating are complex and need to be explored in more depth for future work. Ultimately, improving update interfaces and addressing these open questions will enhance the security overall.

Appendix A: Study One

A.1 Phase One: Interview guide

Thank you for participating in today's study. As you read in the consent form, we will be recording the session so we can review it to make sure that we don't miss any part of our conversation. Your name will not be associated with the recording or with any data I collect. Your comments and opinions will only be used in combination with the feedback gathered from other people participating in this study. Your individual comments will remain confidential. Do you have any questions regarding the consent form? Do I have your permission to start the recording?

A.1.1 Session Introduction

Today, I'm going to be talking with you about security issues and discussing your software updating habits. The interview should last around 30 to 45 minutes. Before we begin, there are a few things I would like to mention:

1. During our discussion, I will ask you to share with me your thoughts and opinions on the different topics that we cover. I would like you to be honest and

straightforward about your knowledge and habits regarding software security. I might ask you to expand on anything you mention if the information could be useful to the study. Please keep in mind that there are no right or wrong answers.

2. We are not here to test you or your knowledge about security issues or software technology. We want to get a better understanding of how people really think and act when it comes to these issues.

3. Please feel free to comment on any thoughts or ideas you have as we talk. Your feedback is important in that it helps us get a better picture of real user behavior. Do you have any questions before we begin? Okay, let's get started.

A.1.2 Security Awareness

1. Are you aware of any major breaches to personal online security?

2. Have you heard about the following security breaches: Heartbleed, Adobe 2013, Target 2013

3. How do you find about online security issues? From which sources?

4. How do you determine integrity of the source of software update? Have you ever avoided installing an update because it was not from an authentic source?

5. Have you ever been a victim of an online security breach?

A.1.3 Security Habits

1. Do you secure your physical electronics (like laptop, tablet, phone)? How?
2. Are you concerned about email security? Do you take any measures to protect your security on email?
3. Are you concerned about software security? Do you take any measures to protect your security in regards to software?

A.1.4 Software Updates

1. What comes to mind when you think of software updates? How do you feel about them? Do you usually have a positive or negative feeling? What is your motivation for acting on software updates? How do you decide whether to go through the update or not?
2. How comfortable are you in installing software updates? If not, what would make you more comfortable?
3. Do you treat all software updates the same? Or do you think or act differently depending on the type, say whether they are app updates, or operating system updates?
4. Do you feel that it's necessary to update software every time an update is available? Are there some updates that you always do and some that you usually ignore?

5. Does the process of software updating feel like a necessity or more of an interruption? Why?

6. Do you typically understand what the update is going to do to the software before you download and install it? Do you read the information presented in the update notice? Do you understand the information? Does this information have an impact on your decision to go through with the update or not?

A.1.5 Software Update Process

1. Discovering the update

   (a) How do you find out about a software update? Do you usually wait to be notified or if you hear about an update from an external source, do you seek it out?

   (b) Are there any barriers that get in the way of you discovering updates?

   (c) What makes it easy to discover updates?

2. Downloading the update

   (a) How do you download updates? Where do you go or what do you do? Do you seek them out?

   (b) Are there any barriers to downloading updates? For example, connectivity issues like data usage, Wi-Fi availability?

   (c) What makes it easy to download updates?

3. Installing the update

    (a) How do you typically install updates? Automatically or manually?

    (b) When do you typically install updates? After download or later (why if later)?

    (c) Are there any barriers to installing updates (restarting, downloading an installer, interrupting current activity)?

    (d) What makes it easy to install updates?

4. Applying the Update

    (a) What do you typically expect to have happen after you install an update and begin using the software again?

    (b) Does your interaction with the software typically match your expectations after

    (c) Do you usually have a positive or negative experience after installing an update?

    (d) Do any of these factors have an effect on how you feel about future updates?

A.1.6 Software Updating Preferences

1. Do you configure any of your systems to do updates in any of silent, automatic, manual ways? Do you prefer any of these mechanisms? Why or why not? Do

you know where to change these settings in your operating system? And the settings for each piece of software.

2. What are the reasons you go through with software updates? Does it depend on the device? Piece of software? Operating system? How do you determine if an update is critical or not? How do you feel about whether an update might address a security issue vs. a feature change?

3. What are the reasons why you might avoid software updates? Frequency of patches (Do you avoid if they're released more frequently?), cost of updates, connection speed, incompatibility with other software, fear of breaking existing software/changing interface.

A.1.7 Current Software Updating Interfaces

1. How frequently do you update your software?

2. Which device do you most frequently update and why? (Does the update behavior depend on the device being used?)

3. What software do you update more often?

4. Have you installed updates both on Windows PC and Mac OS? Which one did you prefer?

5. What browsers do you use to access Internet?

6. Which one do you prefer most in terms of update?

7. Did you ever avoid installing an update to avoid incurring additional charges when your Internet was not free? (wifi/3G/broadband)

8. How would you want to improve the updating process?

A.2 Phase Three: Think-aloud tasks

A.2.1 Locating updates

1. Find out if there are software updates for your machine

2. Find the list of available updates

3. Cancel the OS X update from taking place

A.2.2 Authenticating the Source

1. Verify that the source of iTunes 11.1.1. update is authentic

2. How would you get additional information about what the iTunes 11.1.1 update does?

A.2.3 Installed Updates

1. Close the update manager and find out if there are any more updates

2. Find out which updates were recently installed

A.2.4 Information About the Update

1. How much time will the next update will take to download and install?

2. Change the update preferences for iTunes updates

3. Check for applications the OS X 10.9.2 update is incompatible with

4. Read update ratings and comments from users

Appendix B: Study Two

B.1 Survey Instrument

1. Is your primary mobile phone an Android device?
   (a) Yes
   (b) No
2. Have you used the Google Play Store for at least a month?
   (a) Yes
   (b) No
3. Which of the following statements are true about the Play Store? Please check all that apply:
   (a) The Play Store is the official app store for the Android Operating System
   (b) The Play Store is available as an app on Android smartphones
   (c) You cannot rate apps on the Play Store
   (d) You can write reviews for apps on the Play Store
   (e) The Play Store is the official app store for the iOS Operating System

(f) All Android apps on the Play Store are free

(g) None of the above statements are true

B.1.1 Part One

1. Domain Specific Risk Taking (DoSpeRT) scale [35]

2. Consideration for Future Consequences (CFC) scale [36]

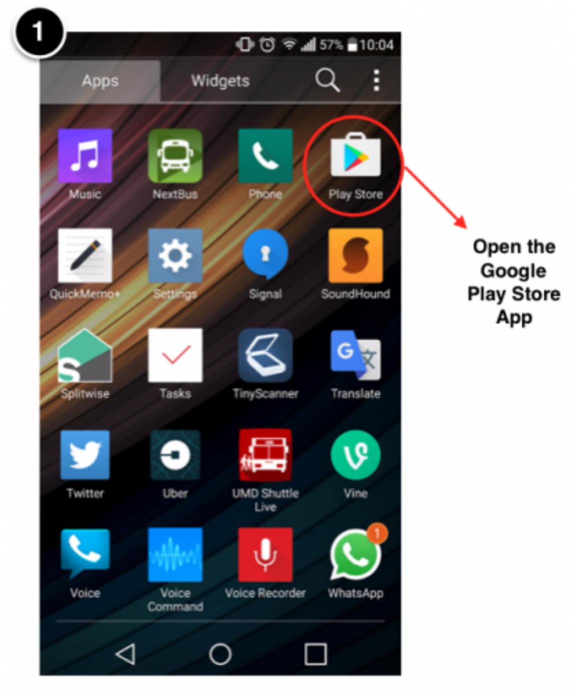
3. Need for Cognition (NFC) scale [37]

4. Resistance to Change (RTC) scale [52]

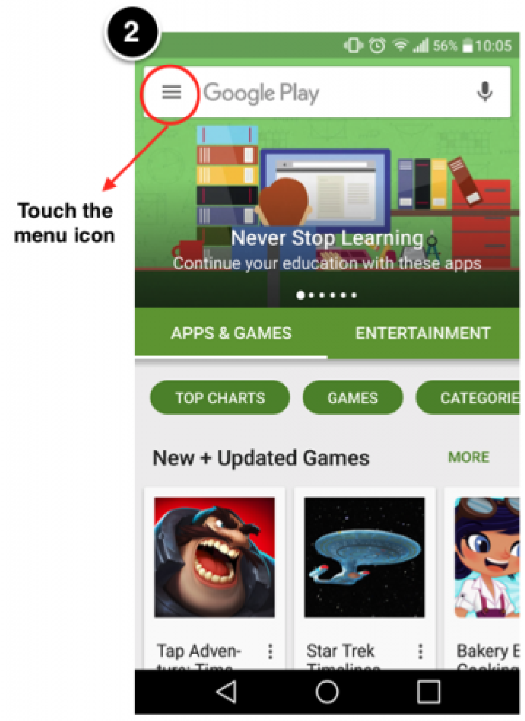
5. Security Behavior Intentions (SeBIS) scale [32]

B.1.2 Part Two

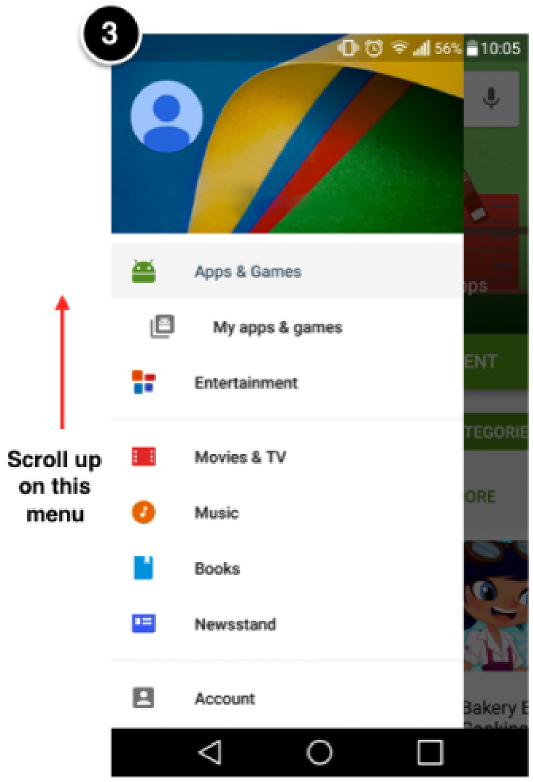
1. Please report the following update settings for your Android device by following the instructions in the images below.


1
57% 10:04
Apps
Widgets
Music
NextBus
Phone
Play Store
QuickMemo+
Settings
Signal
SoundHound
Splitwise
Tasks
TinyScanner
Translate
Twitter
Uber
UMD Shuttle Live
Vine
Voice
Voice Command
Voice Recorder
WhatsApp
Open the Google Play Store App

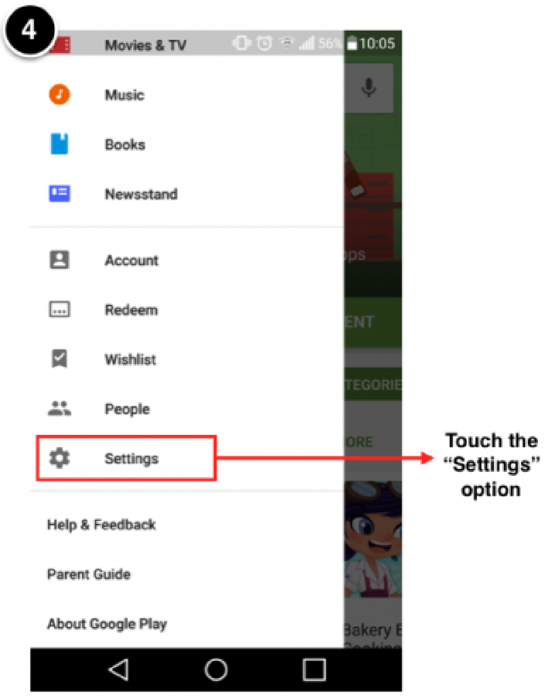
(a)


2
56% 10:05
Google Play
Touch the menu icon
Never Stop Learning
Continue your education with these apps
APPS & GAMES
ENTERTAINMENT
TOP CHARTS
GAMES
CATEGORIE
New + Updated Games
MORE
Tap Adven-
Star Trek
Bakery E

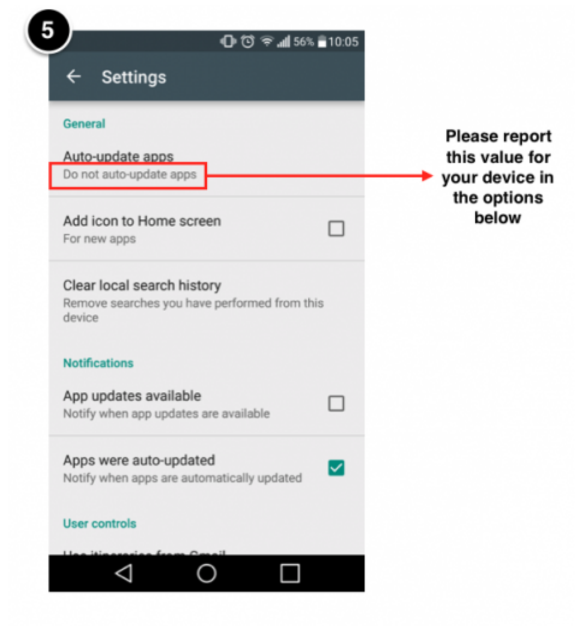
(b)


3
10:05
Apps & Games
My apps & games
Entertainment
Movies & TV
Music
Books
Newsstand
Account
Scroll up on this menu

(c)


4
Movies & TV
10:05
Music
Books
Newsstand
Account
Redeem
Wishlist
People
Settings
Help & Feedback
Parent Guide
About Google Play
Touch the "Settings" option

(d)



5

Settings

General

Auto-update apps
Do not auto-update apps

Add icon to Home screen
For new apps

Clear local search history
Remove searches you have performed from this device

Notifications

App updates available
Notify when app updates are available

Apps were auto-updated
Notify when apps are automatically updated

User controls

Please report this value for your device in the options below

(e)

Please report the text in image (5) for your device:

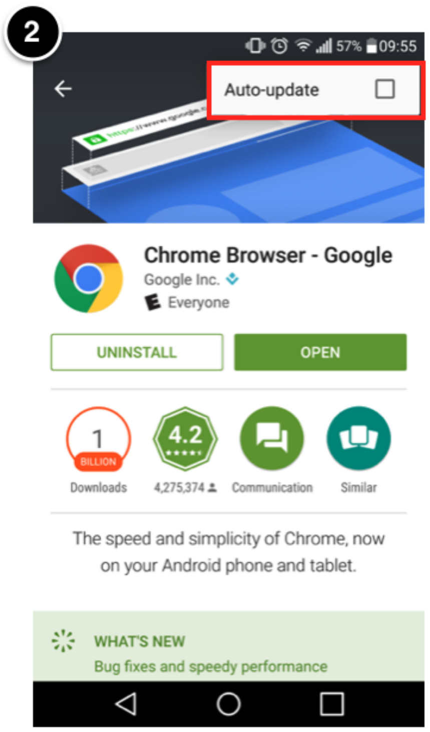
(a) Do not auto-update apps.

(b) Auto-update apps at any time. Data charges may apply.

(c) Auto-update apps over Wi-Fi only.

(d) I don't know

2. The Google Play Store allows certain apps to be updated manually. For example, the following images describe how by deselecting the Auto-update checkbox, the Google Chrome app will no longer be auto-updated. (Only shown if the answer to the previous questions is (b) or (c))



Do you manually update certain apps in the manner shown above?

(a) I do not manually update any of my apps in this manner

(b) I manually update some of my apps in this manner

(c) I manually update most of my apps in this manner

(d) I don't know

3. The following is a list of the most downloaded Android apps from the Google Play Store. From this list, please select ALL the ones you have installed on your Android phone. List taken from [53].

4. For a maximum of 10 randomly selected applications from the previous question:

(a) Assuming no data charges apply, how comfortable are you setting security updates to automatically download and install for the following apps? [0 - 100]

(b) Assuming no data charges apply, how comfortable are you setting NON security updates to automatically download and install for the following apps? [0 - 100]

B.1.3 Part Three

1. Have you ever regretted or had a negative experience updating any software?

   (a) Yes

   (b) No

   (c) I don't remember

2. The following are some reasons why people regret installing updates. Please check all the reasons that have caused you to regret updating your software. (Only shown if the answer to the previous questions is "Yes")

   (a) The update introduced new bugs in the software.

   (b) The update changed the user interface.

   (c) The update used up a lot of data.

   (d) The update took more time to install than I expected it to take.

   (e) The old version of the software worked better than the updated one.

   (f) Other - Write In

B.1.4 Demographics

1. What is your age? [Write In]
2. What is your annual household income?
   (a) Less than $25,000
   (b) $25,000 to $34,999
   (c) $35,000 to $49,999
   (d) $50,000 to $74,999
   (e) $75,000 to $99,999
   (f) $100,000 to $124,999
   (g) $125,000 to $149,999
   (h) $150,000 or more
   (i) Prefer not to answer
3. What is the highest education level you have completed?
   (a) No High School
   (b) High School Graduate
   (c) Some College
   (d) Bachelor's Degree
   (e) Associate's Degree
   (f) Master's Degree

   (g) Doctoral Degree

   (h) Professional Degree (e.g., MBA, J.D.)

   (i) Prefer not to answer

4. What gender do you most closely identify with?

   (a) Male

   (b) Female

   (c) Other

   (d) Prefer not to answer

B.2 Interview Guide

Thank you for participating in today's study. As you read in the consent form, we will be recording the session so we can review it to make sure that we don't miss any part of our conversation. Your name will not be associated with the recording or with any data I collect. Your comments and opinions will only be used in combination with the feedback gathered from other people participating in this study. Your individual comments will remain confidential. Do you have any questions regarding the consent form? Do I have your permission to start the recording?

B.2.1 Session Introduction

Today, I'm going to be talking with you about Android updating. The interview should last around 30 to 40 minutes. Before we begin, there are a few things I would like to mention:

1. During our discussion, I will ask you to share with me your thoughts and opinions on the different topics that we cover. I would like you to be honest and straightforward about your knowledge and habits regarding how you update your Android applications. I might ask you to expand on anything you mention if the information could be useful to the study. Please keep in mind that there are no right or wrong answers. We are not here to test you or your knowledge about any software technology.

2. Please feel free to comment on any thoughts or ideas you have as we talk. Your feedback is important in that it helps us get a better picture of real user behavior. Do you have any questions before we begin? Okay, let's get started.

B.2.2 Android Updating Process

1. Have you ever updated the applications on your Android device?

2. Could you please tell me more about how you update your apps on your smartphone?

   (a) For those who said automatic:

i. Why do you update your apps in this manner?

ii. Do you receive post-update notifications? Why? Why not?

iii. Do you click them to see what has changed? Why? Why not?

iv. Have you ever changed the update settings of your phone? Why or why not did you change the settings?

(b) For those who said manual:

i. Why do you update your apps in this manner?

ii. Whats your reaction when first receive the update notification?

iii. Do you click on it immediately or do you wait? Why or why not?

iv. Do you examine what has changed in each app or do you click "Update All"? Why or why not?

(c) For part-auto, part-manual:

i. Which are the apps that you update manually? Why do you do so?

ii. (Repeat both auto and manual questions)

B.2.3 Examine Android Device

1. Could you please show me your device? How long have you had this device for?

2. Does anyone else beside you use it?

3. Please show me where you can configure the update settings of your device (If changed settings before)

4. (If discrepancy between stated and observed settings) Why is there a difference? Could you please clarify?

B.2.4 Examine Android applications

1. For manual/part-auto, part manual update users:

   (a) Which applications are updated and which ones are not updated?

   (b) Tell me more about the apps you update and the ones you havent updated. Why, or why not?

   (c) Which are apps you more frequently use/trust/satisfied with/care about? Why or why not?

2. For auto-update users, look at the last five updated applications:

   (a) Are there any that you don't want to keep updated? Why or why not?

   (b) What had changed in each of those applications updates (looking at the change log)? Are you comfortable

Bibliography

[1] Symantec. 2015 Internet Security Threat Report, Volume 20. https://know.elq.symantec.com/LP=1542, 2015.

[2] Microsoft. Microsoft Security Intelligence Report Volume 18: July through December 2014. http://download.microsoft.com/download/7/1/A/71ABB4EC-E255-4DAF-9496-A46D67D875CD/Microsoft_Security_Intelligence_Report_Volume_18_English.pdf, 2015.

[3] National Cyber Security Alliance. Stay Safe Online. https://www.staysafeonline.org/, 2015.

[4] Iulia Ion, Rob Reeder, and Sunny Consolvo. "...No one Can Hack My Mind": Comparing Expert and Non-Expert Security Practices. In *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*, pages 327–346, Ottawa, July 2015. USENIX Association.

[5] Kami E. Vaniea, Emilee Rader, and Rick Wash. Betrayed by Updates: How Negative Experiences Affect Future Security. In *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems*, CHI '14, pages 2671–2674, New York, NY, USA, 2014. ACM.

[6] Rick Wash, Emilee Rader, Kami Vaniea, and Michelle Rizor. Out of the Loop: How Automated Software Updates Cause Unintended Security Consequences. pages 89–104. USENIX Association, 2014.

[7] Paul Dourish, E. Grinter, Jessica Delgado de la Flor, and Melissa Joseph. Security in the Wild: User Strategies for Managing Security As an Everyday, Practical Problem. *Personal Ubiquitous Comput.*, 8(6):391–401, November 2004.

[8] Martina Angela Sasse, Sacha Brostoff, and Dirk Weirich. Transforming the "Weakest Link"–a Human/Computer Interaction Approach to Usable and Effective Security. *BT Technology Journal*, 19(3):122–131, July 2001.

[9] Joshua Sunshine, Serge Egelman, Hazim Almuhimedi, Neha Atri, and Lorrie Faith Cranor. Crying Wolf: An Empirical Study of SSL Warning Effectiveness. In *Proceedings of the 18th Conference on USENIX Security Symposium*, SSYM'09, pages 399–416, Berkeley, CA, USA, 2009. USENIX Association.

[10] Devdatta Akhawe and Adrienne Porter Felt. Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness. In *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*, pages 257–272, Washington, D.C., 2013. USENIX.

[11] Min Wu, Robert C. Miller, and Simson L. Garfinkel. Do Security Toolbars Actually Prevent Phishing Attacks? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 601–610, New York, NY, USA, 2006. ACM.

[12] W. Keith Edwards, Erika Shehan Poole, and Jennifer Stoll. Security Automation Considered Harmful? In *Proceedings of the 2007 Workshop on New Security Paradigms*, NSPW '07, pages 33–42, New York, NY, USA, 2008. ACM.

[13] Lorrie Faith Cranor. A Framework for Reasoning About the Human in the Loop. In *Proceedings of the 1st Conference on Usability, Psychology, and Security*, UPSEC'08, pages 1:1–1:15, Berkeley, CA, USA, 2008. USENIX Association.

[14] Microsoft. Install Windows Updates. http://windows.microsoft.com/en-us/windows-vista/install-windows-updates, 2015.

[15] Apple. Get Software Updates for Your Mac. https://support.apple.com/en-us/HT201541, 2015.

[16] Joseph Dadzie. Understanding Software Patching. *Queue*, 3(2):24–30, March 2005.

[17] Thomas Duebendorfer and Stefan Frei. Why Silent Updates Boost Security. *TIK, ETH Zurich, Tech. Rep*, 302, 2009.

[18] Antonio Nappa, Richard Johnson, Leyla Bilge, Juan Caballero, and Tudor Dumitras. The Attack of the Clones: A Study of the Impact of Shared Code on Vulnerability Patching. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 692–708, May 2015.

[19] Milan Vojnovic and Ayalvadi Ganesh. On the Effectiveness of Automatic Patching. In *Proceedings of the 2005 ACM Workshop on Rapid Malcode*, WORM '05, pages 41–50, New York, NY, USA, 2005. ACM.

[20] Christos Gkantsidis, Thomas Karagiannis, and Milan VojnoviC. Planet Scale Software Updates. In *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '06, pages 423–434, New York, NY, USA, 2006. ACM.

[21] Thomas Gerace and Huseyin Cavusoglu. The Critical Elements of the Patch Management Process. *Commun. ACM*, 52(8):117–121, August 2009.

[22] Jon Oberheide, Evan Cooke, and Farnam Jahanian. If It Ain't Broke, Don't Fix It: Challenges and New Directions for Inferring the Impact of Software Patches. In *Proceedings of the 12th Conference on Hot Topics in Operating Systems*, HotOS'09, pages 17–17, Berkeley, CA, USA, 2009. USENIX Association.

[23] Henry Corrigan-Gibbs and Jay Chen. Flashpatch: Spreading Software Updates over Flash Drives in Under-connected Regions. In *Proceedings of the Fifth ACM Symposium on Computing for Development*, ACM DEV-5 '14, pages 1–10, New York, NY, USA, 2014. ACM.

[24] Rick Wash and Emilee Rader. Too Much Knowledge? Security Beliefs and Protective Behaviors Among United States Internet Users. In *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*, pages 309–325, Ottawa, July 2015. USENIX Association.

[25] Marshini Chetty, Richard Banks, A.J. Brush, Jonathan Donner, and Rebecca Grinter. You're Capped: Understanding the Effects of Bandwidth Caps on Broadband Use in the Home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 3021–3030, New York, NY, USA, 2012. ACM.

[26] Arunesh Mathur, Brent Schlotfeldt, and Marshini Chetty. A Mixed-methods Study of Mobile Users' Data Usage Practices in South Africa. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '15, pages 1209–1220, New York, NY, USA, 2015. ACM.

[27] Michael Fagan, Mohammad Maifi Hasan Khan, and Ross Buck. A Study of Users' Experiences and Beliefs about Software Update Messages. *Computers in Human Behavior*, 51, Part A:504 – 519, 2015.

[28] Alain Forget, Sarah Pearman, Jeremy Thomas, Alessandro Acquisti, Nicolas Christin, Lorrie Faith Cranor, Serge Egelman, Marian Harbach, and Rahul Telang. Do or Do Not, There Is No Try: User Engagement May Not Improve Security Outcomes. In *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*, pages 97–111, Denver, CO, June 2016. USENIX Association.

[29] Kami Vaniea and Yasmeen Rashidi. Tales of Software Updates: The Process of Updating Software. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 3215–3226, New York, NY, USA, 2016. ACM.

[30] Kandha Sankarpandian, Travis Little, and W. Keith Edwards. Talc: Using Desktop Graffiti to Fight Software Vulnerability. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1055–1064, New York, NY, USA, 2008. ACM.

[31] Yuan Tian, Bin Liu, Weisi Dai, Blase Ur, Patrick Tague, and Lorrie Faith Cranor. Supporting Privacy-Conscious App Update Decisions with User Reviews. In *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*, SPSM '15, pages 51–61, New York, NY, USA, 2015. ACM.

[32] Serge Egelman and Eyal Peer. Scaling the Security Wall: Developing a Security Behavior Intentions Scale (SeBIS). In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 2873–2882, New York, NY, USA, 2015. ACM.

[33] Serge Egelman, Marian Harbach, and Eyal Peer. Behavior Ever Follows Intention?: A Validation of the Security Behavior Intentions Scale (sebis). In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 5257–5261, New York, NY, USA, 2016. ACM.

[34] Susanne G Scott and Reginald A Bruce. Decision-making style: The development and assessment of a new measure. *Educational and psychological measurement*, 55(5):818–831, 1995.

[35] Ann-Renée Blais and Elke U Weber. A domain-specific risk-taking (DOSPERT) scale for adult populations. *Judgment and Decision Making*, 1(1), 2006.

[36] Jeff Joireman, Monte J Shaffer, Daniel Balliet, and Alan Strathman. Promotion orientation explains why future-oriented people exercise and eat healthy evidence from the two-factor consideration of future consequences-14 scale. *Personality and Social Psychology Bulletin*, 38(10):1272–1287, 2012.

[37] Richard E Petty, John T Cacioppo, and Chuan Feng Kao. The efficient assessment of need for cognition. *Journal of Personality Assessment*, 48(3):306–307, 1984.

[38] Jim H Patton, Matthew S Stanford, et al. Factor structure of the barratt impulsiveness scale. *Journal of clinical psychology*, 51(6):768–774, 1995.

[39] Irving Seidman. *Interviewing As Qualitative Research: A Guide for Researchers in Education and the Social Sciences*. Teachers college press, 2013.

[40] Codenomicon. The Heartbleed Bug. http://heartbleed.com, April 2014.

[41] Target Corporation. Data Breach FAQ. https://corporate.target.com/about/shopping-experience/payment-card-issue-faq, April 2014.

[42] David Barrera and Paul Van Oorschot. Secure Software Installation on Smartphones. *IEEE Security & Privacy*, 9(3):42–48, May 2011.

[43] Dion Hinchcliffe. The App Store: The New "Must-Have" Digital Business Model. http://www.zdnet.com/article/the-app-store-the-new-must-have-digital-business-model, January 2010.

[44] Jared Newman. How Mobile Apps are Changing Desktop Software. http://www.pcworld.com/article/259110/app_invasion_coming_soon_to_your_pc.html, July 2012.

[45] Statista. Global market share held by operating systems Desktop PCs from January 2012 to June 2015. http://www.statista.com/statistics/218089/global-market-share-of-windows-7, 2015.

[46] Darren Murph. Apple mac app store: open for business starting January 6th. http://www.engadget.com/2010/12/16/apple-mac-app-store-open-for-business-starting-january-6th/, 2010.

[47] Andy Crabtree. Design in the Absence of Practice: Breaching Experiments. In *Proceedings of the 5th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques*, DIS '04, pages 59–68, New York, NY, USA, 2004. ACM.

[48] T. Boren and J. Ramey. Thinking Aloud: Reconciling Theory And Practice. *Professional Communication, IEEE Transactions on*, 43(3):261–278, Sep 2000.

[49] Erica L. Olmsted-Hawala, Elizabeth D. Murphy, Sam Hawala, and Kathleen T. Ashenfelter. Think-aloud Protocols: A Comparison of Three Think-aloud Protocols for Use in Testing Data-dissemination Web Sites for Usability. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 2381–2390, New York, NY, USA, 2010. ACM.

[50] Working with System Permissions. https://developer.android.com/training/permissions/index.html.

[51] Serge Egelman and Eyal Peer. The Myth of the Average User: Improving Privacy and Security Systems Through Individualization. In *Proceedings of the 2015 New Security Paradigms Workshop*, NSPW '15, pages 16–28, New York, NY, USA, 2015. ACM.

[52] Shaul Oreg. Resistance to change: developing an individual differences measure. *Journal of applied psychology*, 88(4):680, 2003.

[53] List of Most Downloaded Android Applications. https://en.wikipedia.org/wiki/List_of_most_downloaded_Android_applications.

[54] Eyal Peer, Joachim Vosgerau, and Alessandro Acquisti. Reputation as a sufficient condition for data quality on amazon mechanical turk. *Behavior Research Methods*, 46(4):1023–1031, 2014.

[55] Rosemary R Gliem and Joseph A Gliem. Calculating, interpreting, and reporting Cronbachs alpha reliability coefficient for Likert-type scales. Midwest Research-to-Practice Conference in Adult, Continuing, and Community Education, 2003.

[56] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[57] Ann-Renee Blais and Elke U Weber. The domain-specific risk taking scale for adult populations: Item selection and preliminary psychometric properties. Technical report, DTIC Document, 2009.

[58] Shaul Oreg, Mahmut Bayazit, Maria Vakola, Luis Arciniega, Achilles Armenakis, Rasa Barkauskiene, Nikos Bozionelos, Yuka Fujimoto, Luis González, Jian Han, et al. Dispositional resistance to change: measurement equivalence and the link to personal values across 17 nations. *Journal of Applied Psychology*, 93(4):935, 2008.

[59] S. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, 28(2):129–137, September 2006.

[60] Adrienne Porter Felt, Elizabeth Ha, Serge Egelman, Ariel Haney, Erika Chin, and David Wagner. Android Permissions: User Attention, Comprehension, and Behavior. In *Proceedings of the Eighth Symposium on Usable Privacy and Security*, SOUPS '12, pages 3:1–3:14, New York, NY, USA, 2012. ACM.

[61] Sauvik Das, Adam D.I. Kramer, Laura A. Dabbish, and Jason I. Hong. The Role of Social Influence in Security Feature Adoption. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '15, pages 1416–1426, New York, NY, USA, 2015. ACM.

[62] Sauvik Das, Adam D.I. Kramer, Laura A. Dabbish, and Jason I. Hong. Increasing Security Sensitivity With Social Proof: A Large-Scale Experimental Confirmation. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 739–749, New York, NY, USA, 2014. ACM.

[63] Metaquark. Appfresh. http://metaquark.de/appfresh/mac, 2015.

[64] Sparkle Project. http://sparkle-project.org, 2015.

[65] A. Acker and B. Beaton. Software Update Unrest: The Recent Happenings Around Tinder and Tesla. In *Proceedings of the 49 Hawaii International Conference System Sciences*, HICSS '16. IEEE, 2016.