CONFIGURATION OF APPLICATION PERMISSIONS WITH CONTEXTUAL ACCESS CONTROL

by

Andrew Besmer

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Information Technology

Charlotte

2013

Approved by:

Dr. Heather Richter Lipford

Dr. Mohamed Shehab

Dr. Celine Latulipe

Dr. Richard Lambert

Dr. Lorrie Cranor

ABSTRACT

ANDREW BESMER. Configuration of application permissions with contextual access control. (Under the direction of DR. HEATHER RICHTER LIPFORD)

Users are burdened with the task of configuring access control policies on many different application platforms used by mobile devices and social network sites. Many of these platforms employ access control mechanisms to configure application permissions before the application is first used and provide an all or nothing decision for the user. When application platforms provide fine grained control over decision making, many users exhibit behavior that indicates they desire more control over their application permissions. However, users who desire control over application permissions still struggle to properly configure them because they lack the context in which to make better decisions. In this dissertation, I attempt to address these problems by exploring decision making during the context of using mobile and social network applications. I hypothesize that users are able to better configure access control permissions as they interact with applications by supplying more contextual information than is available when the application is being installed. I also explore how logged access data generated by the application platform can provide users with more understanding of when their data is accessed. Finally, I examine the effects that this contextually improved application platform has on user decision making.

DEDICATION

I dedicate this dissertation to my family and friends, thank you for everything. None of this was possible without you.

ACKNOWLEDGMENTS

Despite the fact that this is the first section you will read, it is the last I will write. As I reflect on my journey and how I got here, there are so many people I owe a debt of gratitude to. While it is me who this document is attributed to, my success is not and should not be considered a singular effort. So while you will not find any results in this section of the dissertation, I still encourage you to read it as well.

First, the work in this dissertation would not be possible without funding from various sources including the fellowship I received: Graduate Assistance in Areas of National Need or GAANN. Additionally, I was fortunate enough to receive a small Google grant and some of my collaborators were funded through the National Science Foundation. Finally, UNC Charlotte has a program called GASP or Graduate Assistant Support Plan. These funding sources have made the cost of carrying out my work much more manageable.

Next, I would like to thank my dissertation committee which consisted of Dr. Heather Richter Lipford, Dr. Celine Latulipe, Dr. Mohamed Shehab, Dr. Richard Lambert, and Dr. Lorrie Cranor. I have great esteem for these scholars and it has been my honor to work with them. Dr. Lipford, in addition to serving as the chair for my committee, was also my advisor more than half of my college career. I can not envision a better advisor and I am so grateful for everything she has done. She has celebrated both my academic successes and failures, and the advice she has given is second to none. I was also very fortunate to have a committee that was willing to give me enough rope to hang myself with and then provide me with the guidance,

feedback, and direction I needed to ensure that did not happen. I have enjoyed the privilege of being able to set my own direction and thank the committee for allowing me to do that.

I have had several collaborators beyond just those mentioned in this dissertation. Some of these collaborators in no particular order are: Tyler Thomas, Gorrell Cheek, Ashley Anderson, Katie Froiland, Vanessa Hernandez, Josh Harvel, Crystal Knox, Lauren Hamilton, Charisse Cotton, Jason Watson, Kate Strater, and Erik Northrop. It has been my pleasure to learn from and work with all of them. Jason Watson has become my very good friend and has helped me more than I could ask of any person. I am honored to have got to know/worked with such a kind, caring, and giving individual. Professionally, there is not a study I have been involved in that I have not discussed with him. Personally, there is not a thing I would hesitate to debate or ask him about.

I would like to thank the many friends I have gotten to know. Day in and day out I have enjoyed the company of many of my lab mates. Among them, again in no particular order, you will find Berto Gonzalez who is in competition with Alexander Adams to be the world's most interesting man. I will leave it to others to judge the outcome. Erik Northrop and Vikash Singh who share my enthusiasm for PHP and Javascript. Okan Pala who gave my wife and I our first official dance lesson. Felesia Stukes who takes a genuine interest in both her work and the work of others around her. Jason Watson who has been an amazing friend. Michael Whitney, he still holds the title of the worlds most interesting man that Berto and Alex are currently competing for. Jun Zhu who is a very kind and hard working student. Matt Campbell

who I wish I had more time to get to know. Mick Smythwood who very well might be the fastest learner I have ever met. Pam Wisniewski, her research motivated some of my own and ultimately got me more interested in statistical methods. Erin Carroll who is without a doubt one of the most intelligent and hard working women in computing. And finally, Jing Xie, my coast to coast friend. By sheer coincidence, fate brought us both from adjoining desks to the west coast at the same time, mere miles from one another. Jing introduced my wife and I to traditional Chinese restaurants and food. She is the reason that to this very day my wife uses chopsticks to eat rice. Besides my lab mates I have one other good friend I would like to acknowledge. Jamal Burgess who is one of the smartest people I know. Nearly everyday he and I will talk and I always enjoy the conversation.

My parents were always an advocate of higher education and it is fair to say I would not have gone down this path if not for them. So Mom, Dad, thank you. Someone once asked, "Does the world need another Dr. Besmer?" I do not know the answer to that question, but I can say that there are very few brother/sister PhDs that are at the same institution, in the same building, at the same time. While we were in very different PhD programs, my sister has shared the experiences (good and bad) of graduate life with me. While she managed to graduate a few months earlier than I did, my dissertation is longer. Just saying. Seriously though, thanks for being one of the few people who understood everything happening in graduate school. I would like to thank my brother Timothy, he is fun to be around and has no problems expressing his opinions. Keep up the good work. I would also like to thank my wife Britney. She has been supportive of me, my studies, and kept me laughing with comments like,

"I'm not a participant in your user studies mister!" Without her, instead of being a balding PhD, I would probably resemble some sort of completely bald mad scientist. I would be walking around talking about sharks with frickin' lasers beams attached to their heads.

Last but not least, UNC Charlotte itself has been an amazing institution. I came to UNC Charlotte because of the reputation it has as a national center of academic excellence in information assurance education. As luck would have it UNC Charlotte also has stellar faculty, staff and students. I have been consistently amazed at how receptive the faculty are towards students needs and desires. Additionally, I have not encountered a single staff member at UNC Charlotte who was not exceptionally kind and caring. All have gone above and beyond what is expected of them. I thank them all and the institution itself for continuing to find such amazing students, people, and talent.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

People use a variety of applications on different platforms everyday for entertainment and socializing. On social network sites, users can add 3rd party applications to enhance their profiles, interact with friends, play games, share photos, and even read their news feed. Similarly, we see the same kind of 3rd party applications on mobile device application platforms like Android and iOS (Apple's iPhone OS).

Everyday, people increasingly install and use these applications. Facebook's application platform reports that users install applications more than 20 million times a day [21]. On Android's application platform, it is estimated that each Android user will download 9 applications a month and use applications on their devices at least 79 minutes a day [1]. These numbers continue to grow as smartphone usage becomes more popular and ubiquitous.

In order to provide services and functionality to users, 3rd party applications consume user data and/or device resources. On social network sites, 3rd party applications consume user profile data such as: a user's name, list of friends, interests, etc. In exchange for trading user data, the application may provide social enhancements like a list of friends with similar interests to the users. Mobile application platforms operate similarly and require both user data and access to phone features. For example, a photo tagging application might require access to the device camera, network

connectivity, GPS location information, and user contact data.

There are good reasons to allow applications to consume profile data and resources, such as sending and receiving SMS messages. For example, there are several applications in the marketplace that replace the default SMS messaging application. This is the value of the Android ecosystem, if the default application is not sufficient for your needs, you can build or find a new one. The same is true of almost all parts of Android up to and including the home screens of the mobile phone.

Users are empowered to decide which applications they install, be it on their social network profiles or mobile devices, and therefore which applications have access to their data and resources. Yet, despite this level of provided control, we see numerous problems occurring. For example, users suffer from unintended disclosure of personal information to pay-per-text services that can cost users money. In December 2011, a set of malicious Android applications had to be removed from user phones because they sent and received unexpected SMS messages on behalf of the user [11].

Thus, as a result of 3rd party application usage, user data and resources are at a potential risk of unintended exposure. For example, BBC News developed an application that had the ability to harvest large amounts of user profile data in just three hours [4]. In March 2011, 21 malicious applications were pulled from the Android Market. These applications, which required user consent to install, harvested personal mobile phone data as well as identifying device information. It was estimated to have affected 50,000 devices before being removed from the market.

Fortunately, users are in control of the applications they install and use. Before using the applications for the first time, users are asked to configure appropriate

permissions and select what data and resources can be accessed. The method of configuration varies across different platforms, but all platforms share the common purpose of using user data to enhance functionality. The majority of access permissions are readily understood by users. For example, mobile phone users understand that an application may need access to contacts, the network, the phone vibrator, or a list of family members to provide the advertised features.

However, when users are asked to consider the permissions the application is requesting, they are forced to do so out of context. That is to say that users are aware of the type of application, the name of the application, the vendor name, a brief description with a picture, and the list of permissions. These are the only pieces of information available to the user aside from their own previous knowledge or assumptions about what the application actually does. Users can become habituated to the screen [18, 60], too committed to the primary task of installing the application to be interrupted [44, 28], or left to reason with the incomplete information provided [5] to make good decisions.

A survey of nearly 1,000 Android applications shows that almost all applications request half of the permissions Google has categorized as dangerous on the device [25]. Additionally, developers include many permissions they do not even access with their applications. More than 10% of the applications surveyed showed that they requested permissions they did not use and some even requested permissions that effectively could not be granted [25]! Users accept these permissions because they have become habituated to the installation screen, which could result in privacy violations. Similarly, they put other people's information at risk because their devices or

applications may contain personal information about their friends and acquaintances. For example, pictures and telephone numbers of other people are commonly stored on friends' phones and can be at risk of unintended disclosure to installed applications. This Android "permission creep" is a good example of how even developers can be confused about what permissions are and are not required for their applications to implement all of the desired features. Researchers in the area of mobile security are trying to reduce this "permission creep" by providing developers with additional tools to assess their permission needs [62].

It appears that users do care about their privacy and the privacy of others. The Facebook developers documentation on authentication states [23]:

> "There is a strong inverse correlation between the number of permissions your app requests and the number of users that will allow those permissions. The greater the number of permissions you ask for, the lower the number of users that will grant them; so we recommend that you only request the permissions you absolutely need for your app."

In Section 4.1.4, I present findings that suggest users are unwilling to take the time to configure policies. However, like Facebook, I also see motivated users who do take time to configure security settings. Unfortunately, even when these users are presented with security choices outside of the sharing context, they may not make well-informed decisions.

When users are left to authorize permissions out of context it is a source of frustration to them. In some cases, the application is denied installation or upgrading

and is even subject to removal when users are made aware by others. Even well established companies are not immune to this frustration and its repercussions. For example, a very popular online music service, Pandora, provides an application to the Android marketplace enabling users to listen to music on their phones. However, this application requests access to the users' contacts and the ability to create and modify calendar events for the user. The reviews for the Pandora application are littered with examples of users removing the application, calling it malicious, refusing to update to the version with the permission, and switching to competing online radio applications. I even found one review where the user recommended others use AppShield, an application on the market, to repackage the application removing the undesired permissions.

Pandora has good reason to request the permissions and includes an explanation on their website. In order to share a station with friends they ask you to select a contact to send it to. Similarly if you heard an ad for an event and want to attend, they allow you to initiate a new calendar entry. Yet this also happens to be out of context for users performing installation, and apparently misunderstood by many.

I believe that when users are asked to make these decisions in context with more relevant information available, they will be able to make informed decisions that more accurately reflect their desired disclosure preferences. Therefore, I explore methods for providing control over sharing data in context, and the design implications as well as potential impacts.

1.1 Dissertation Statement

User configuration of application permissions is currently failing because users lack the context in which to make decisions. By presenting access control decisions in a contextual way and providing users with concrete information about data access, users are able to make more informed decisions about access to their data without unreasonable user burden.

1.2 Contributions

In this dissertation I make several contributions to existing research.

- I improve the understanding of how users currently use application platforms
- I illustrate why poor mental models developed for social network sites
- I formalize an existing access control model and improve on it formalizing those changes
- I test those changes and show how when given a context users attempt to make decisions appropriate for that context
- I show why poor mental models lead to undesirable policy configurations
- I increase context by pushing access control decisions into runtime
- I show how runtime contextual access control benefits memorability
- I further improve runtime contextual access control and evaluate decision making that occurs with it

- I examine strengths and weakness of runtime contextual access control and highlight paradoxes that occur
- I test a large set of techniques for notifying users about information flows within the sharing context

My overall hypothesis is that providing contextual configuration methods to users can be an effective way to ease the burden of configuration that users encounter. Additionally, providing users with acceptable notifications can improve their understanding of data usage to enable this decision making. By improving the user's ability to configure and understand data access, users are able to take better control of their data on mobile and social network application platforms. This helps to reduce inadvertent user disclosure of personal information. It also ensures that application use better matches the user's desired policy.

Existing research shows promise in using context to improve user understanding of information access. This dissertation expands this research through a comprehensive analysis of methods for displaying contextual information. This research provides an understanding of effective ways to both prompt users and inform them when information is accessed.

1.3 Organization

This dissertation is divided into several chapters each containing a contributing body of work. Each chapter examines different aspects of users' interactions with applications and application platforms. In Chapter 2, I present background and related research to usable privacy and security and contextual information. In the

remaining chapters, I present research in the areas of user understanding, contextual configuration and information flow.

In Chapter 3, I examine how users use and perceive application platforms with a qualitative study assessing how people use Facebook's application platform. By understanding how and why users interact with applications on the platform, I learn how users are making decisions about which applications are granted access to their profile data. Additionally, I gain insight into issues users have while making access control decisions with Facebook applications.

In Chapter 4, I begin to explore the contextual configuration of application permissions by examining it at install time. I contribute findings on how users react to such a system and the different types of policies they configure. In Chapter 5, I push access control decisions into runtime to facilitate giving users additional context. This chapter is split into two major sections. In Section 5.1, I begin by examining runtime contextual configuration on mobile devices using the Android operating system. I also compare the memorability of requested permissions between install time and runtime contextual access control. In Section 5.2, I examine runtime contextual access control on the social network site Facebook. I examine decision making, strengths, and weakness in greater detail.

In Chapter 6, I examine user understanding of information access and test the boundaries of acceptability for different modes of notifications. Providing configuration in context at runtime requires additional notifications to user. I therefore, compare several different methods for informing users of information access and examine the strengths and weaknesses of notification styles that can inform the design

of configuration systems. Finally, I explore the limits of user acceptability by determining the frequency of notifications that result in users no longer tolerating that style of notification. Finally in Chapter 7 I bring all the studies together to discuss the implications of the work I performed as part of this dissertation. I highlight important findings, considerations, and future directions for the research in this area.

In the next chapter, I examine the background work associated with the study of privacy, access control, and information flow in these domains.

CHAPTER 2: BACKGROUND

The management of 3rd party applications presents both security and privacy problems. In this section, I discuss the background research related to these fields.

2.1 Security and Access Control

Information security is defined by three principles: integrity, confidentiality and availability. Each has an important role in maintaining the security of data. The first of the three, integrity, is charged with preventing data from being modified or destroyed. The second, confidentiality, seeks to allow access to information only if it is authorized. Finally, availability aims to provide access to the data in a timely manner [39].

In a networked world, users rely on information security to protect their privacy via confidentiality and protect their identity via integrity. Information security mechanisms are put in place to achieve these goals. These mechanisms—no matter how sophisticated—should not violate the principle of availability. For example, consider a social networking company that seeks to protect users' private information by putting a mechanism in place that ensures servers are always powered down. The company can ensure that no profiles or sensitive information will be released and likewise, they can claim that no one will be able to modify another user's profile. Clearly, this solution does not work and is not usable for a system designed for sharing information.

A type of mechanism used to provide information security is access control. Access control mechanisms provide authorization for subjects to take actions on certain objects. Subjects of an access control systems may include software, computers, users, etc. Objects may include personal data or resources such as the camera or vibrator on a mobile phone. Actions or permissions can include, but are not limited to, reading, modifying or executing resources. To provide authorization, the access control mechanism identifies and authenticates subjects to use available resources. The actual privilege model should follow the principle of least privilege—subjects should only have access privileges to objects necessary to accomplish the assigned task [52].

Access control and information security are necessary to prevent the malicious actions of subjects towards protected objects. Originally, the intention was to prevent a user from inappropriately modifying another user's objects. As early as 1971, researchers recognized the capability system applications have to cause the same malicious actions as humans and therefore also sought to protect objects from applications [35]. Today, applications are widely used and many access control systems treat applications as subjects.

Access control mechanisms rely on being able to both identify and authenticate subjects in order to provide the authorization previously described. Identification is making a claim of who you are while authentication is proving that you are indeed who you claim you are. For example, I can claim to be Douglas ("Dougie") Powers but when asked to produce a drivers license to authenticate this claim, I could not. Authentication is typically based on something you know, something you have, or something you are.

A commonly used example of something you know are passwords. Passwords are widely used to authenticate to computer operating systems, websites and even to check voicemail. Something you have are often tokens such as a smart card, license or credit card. Finally, something you are includes physical characteristics like a fingerprint or voice pattern. Once the subject is identified and authenticated, that subject can then be authorized to perform actions on certain objects. This forms the basis of access control which provides both confidentiality and integrity of access to available resources.

Access control models are generally grouped into one of two categories. They can be discretionary or non-discretionary and sometimes a combination of both. Discretionary access control models allow the owner of an object to manage the access permissions. Non-discretionary or mandatory access control has permissions assigned by someone else other than the subject of the access control rule [53]. Subjects can not change the permissions on objects in non-discretionary access control models.

Three of the most popular access control models are discretionary access control, mandatory access control and role based access control. Role based access control can be either mandatory or discretionary but differs by assigning subjects to defined roles [53]. Roles are typically used to describe a subject's function within an organization; for example, a computer staff person that designs and maintains databases would be assigned the role of database administrator. The roles are then granted access to resources required to perform their function. Thus, a person who has the role of database administrator would be granted full control to the database server and files. Any number of subjects who need to design or maintain the database can

simply be assigned the database administrator role without the need to reproduce an access policy for every subject.

In this dissertation, I explore access control on social networking sites and mobile platforms from a human computer interaction perspective. In both cases, users of these systems are generally considered to own the data or privileges (objects) and configure them appropriately for applications (subjects). Thus, access control in these domains is primarily discretionary access control. Some social network sites use discretionary role based access control models by allowing subjects to form roles and assign access permissions. In this dissertation, I do not cover identification and authentication as they are beyond the scope of this work.

There is related research on access control for 3rd-party applications. For example, Apex by Nauman, explored extending the Android permission model by allowing users to enforce runtime constraints [41]. Apex allows users to have fine grained control over permissions at install time that are enforced during runtime. For example, users could choose not to allow an application to send SMS messages or they could choose to restrict the application to send exactly 4 messages before denying access to the SMS system. While the authors claim that their mechanism called Poly is user-centric, they provide no usability evaluation of these claims.

Shehab, et al. describe an access control control system that allows generalization of profile attributes to be shared with applications [55]. In their findings, they also do not explore the usability of such a system. However, generalization does help to blur the line between disclosure and non-disclosure by providing less detailed information. As a result of providing a generic non-identifiable attribute, the decision to be more

explicit can be put off until a more appropriate time.

In other work, Ongtang et al. present a fine grained system called SAINT [46], which is primarily concerned with providing the ability to create fine grained application to application policies. Another policy enforcement framework, CRePE, supplemented the Android access control with additional context to provide the ability to be selective in certain contexts [13]. In this case, context could be determined by factors such as light, location, time of day, etc. Both of these frameworks address access control shortcomings but do not attempt to examine the human problem. My work will attempt to inform the creation of these schemes with an understanding of how humans interact and understand access control in these domains. In the next section, I discuss both human computer interaction and usable security research that focuses on the human related aspects of security.

2.2 Human Computer Interaction and Usable Security

Users are frequently required to configure access control settings for various common activities such as using mobile devices, surfing the web and interacting with social network site profiles. Configuring access control can be time consuming and difficult for people to understand. In many cases when an access control task is difficult or is excessively time consuming, users adopt a number of different strategies to avoid properly configuring access control. Some of these strategies are: circumventing security, delegating security decisions to others, and accepting defaults policies [17].

Security failures often occur when users are asked to perform difficult security tasks. Research in the area of human computer interaction provides insight into reasons why

users struggle when asked to perform security related tasks. For example, in a study by Whitten and Tygar, users were asked to send a message using a secure email system named Pretty Good Privacy (PGP 5.0). The study finds users ultimately failed to properly configure additional security for emails within 90 minutes provided to complete the tasks [63]. The biggest reason for task failure was user inability to form correct mental models of how the security features worked. Whitten and Tygar define several qualities needed for usable security software. Security software systems should ensure users:

1. Are reliably made aware of the security tasks they need to perform.
2. Are able to figure out how to successfully perform those tasks.
3. Are not able to make dangerous errors.
4. Are sufficiently comfortable with the interface to continue using it.

In later chapters, I use these qualities to validate the improved usability from contextual access control. Whitten and Tygar also discuss some problematic properties of security. For example, the unmotivated user property describes how users do not primarily use systems to configure security, but configuring security is usually secondary to another task. This property has been observed in a number of studies [17, 5]. Another property they describe is abstraction, where configuration interfaces can abstract the lower level details of an access control mechanism. For example, document sharing mechanisms allow a user to share a document with another user

while abstracting the lower level details of generated access control lists that contain rules that specify what subjects are allowed to access which resources.

Existing research demonstrates improvement in easing the configuration burden for access control systems. For example, Expandable Grids built on the notion of how access controls can be thought of in terms of Lampson's Access Control Matrix [35, 50]. The Access Control Matrix lists subjects or roles on one axis and objects on the other. The intersection of these two points contains the access that is granted or denied for the subject on the corresponding object. Reeder's Expandable Grids use colors (red, yellow and green) at the grid intersection as a visual representation of the access policy. Reeder found this method of configuration to be superior to the existing configuration method offered by Windows XP [50]. Watson et al. also applied their visualization to Facebook profile privacy settings [38] and found it to be usable.

In other research, Lipford et al. investigate configuring access control to social network site profiles by providing the user with on-screen controls that relate to audiences. At the time, Facebook audiences were widely understood to be networks—cities or school communities. When comparing an audience-centric configuration to the existing Facebook mechanism, researchers noted an improvement in both accuracy and confidence that the desired access control policy was correctly configured [37].

Felt and Evans find many users accept permissive sharing policies on the Facebook application platform and suggest a novel method to protect user privacy when using social network applications [24]. They posit a privacy-by-proxy design in which social network applications use tags or markup to customize user experience. For example,

in a privacy-by-proxy design, an application wishing to use the user's name may include markup like: <uval id="[$id]" field="name"/>which would be transformed into something like "Andrew Besmer." Markup is used by application developers to reduce the user's burden of policy configuration. Privacy-by-proxy removes the need to prompt users at install time to configure application permissions because the application is not able to access actual profile data. The obvious downside to this approach is that it limits the functionality of the application. Applications that require processing of actual user data would not be able to function using the privacy-by-proxy design.

Work by Mazurek et al. explores a reactive access control mechanism. In a reactive access control system, negotiation over file access which typically takes place offline can be integrated into the mechanism itself [40]. In their study, they test a simulated access control system for a week where participants are prompted to allow or deny access to files requested by familiar associates. Initially, they feared that users would not tolerate frequent prompts requesting permission configuration. Surprisingly, they found the system to be quite practical and that users did not find the prompts annoying.

Expandable grids, reactive access control, and audience view are all interfaces that show how when users are provided more concrete and visual methods of configuration they are more successful. Reactive access control shares similarities to contextual access control proposed in this dissertation. Prompting and configuration are two such similarities that may improve access control decisions with context. The findings from the reactive access control research provide strong motivation to my work by

proposing that additional promptings in exchange for more desirable policies may be an acceptable and usable practice.

Current methods of policy configuration for applications can be grouped into two categories: those that are configured at install time and those configured at runtime. Install time policy configuration is shown to be problematic because users struggle to understand the reason why an application might need access to the requested attributes at the moment they choose to install the application [5]. As an example, consider a book recommendation application on an Android-based mobile device. At the moment the user installs the application, they are shown a set of permissions that the application is requesting to be granted access to. The user is required to make the access control decision based on knowing just a few details about the application, such as the name and perceived benefits. Because so little is known about the application and its permissions, there is no way for a user to appropriately weigh the risks and benefits of the application's requested configuration. Kelley et al. studied install time decisions around permissions and found that many participants did not consider permissions when downloading applications. Additionally, they concluded that it was better make the information more clear and to present information when users were making decisions [32].

Runtime access control configuration also has drawbacks. As previously mentioned, policy configuration is rarely the user's primary task [63] or the user may lack an understanding of the privacy configuration request [27]. Additionally, research on end user license agreements demonstrates there are issues of desensitization to policy configuration screens during software installation [28]. In work by Egelman et al, they

found that alerting users to policy misconfigurations was ineffective. They created a prototype interface based on Venn Diagrams to help users in making policy decisions. They found that participants were much more likely to over-share than under share with their prototype interface [19].

This dissertation seeks to improve access control for social network applications and mobile device platforms. To achieve this, I believe it is necessary to address the human factor of the system. The research presented here provides evidence that improvements can be made by addressing user concerns and designing systems that ease user burden. However, current systems, with the exception of Mazurek et al.'s reactive prototype, are examples of systems where access control configuration is removed from the sharing context. Furthermore, many interfaces and access control mechanisms that claim to be more user-centric have not actually been tested to validate improved usability.

2.3 Privacy

Privacy is a motivating factor for improving usable configuration of 3rd party applications. The type of data consumed in these platforms is often sensitive in nature, for example: photographs on a user's phone or social network site, a list of friends or contacts and even location and microphone data. Third party application access to such sensitive information can form an environment where privacy breaches are possible and in many cases likely to occur.

What privacy means can be defined and interpreted in different ways depending on the context or research domain. Altman, a social scientist, defines privacy as

a dialectic process of dynamic boundary management [2]. Individuals alter their behavior to disclose or withhold information to manage their identity and allegiances with others over time and through social interaction. Altman argues that we manage these boundaries based on our expectations and interaction with others. Palen and Dourish build on Altman's work and create a model to illustrate the privacy tensions that exist in the modern day and how technology changes this privacy theory [48]. They describe three boundaries that exist: the disclosure boundary, identity boundary and temporal boundary.

The disclosure boundary is the tension between being private and public [48]. Everyday we come across many situations where we decide to withhold or disclose information. In online social network sites, users decide to intentionally share interests, religions and sexual orientations to help illustrate and describe themselves to others in their social network. Palen and Dourish also point out that disclosure is sometimes necessary to gain more privacy. They use the example of a celebrity disclosing certain pieces of information to become less accessible.

The identity boundary is the tension between self and other [48]. An example of this tension that frequently arises is the conflict between a person's opinions and that of their employer. For this reason, many companies forbid using their email accounts when publicly stating opinions. Similarly, we may act differently around different groups of people. For example, we might disclose different types of information to our doctor than we would to our car mechanic. We expect that our doctor would protect confidential health information while we might not assume the same from a car mechanic.

Finally, the temporal boundary is a tension between past and future [48]. While it may seem that a decision to share a piece of information is appropriate when we share it, we do not know what effect that sharing decision might have in the future. The problem is magnified with technology because archival of perfect quality is possible for a potentially infinite amount of time. In addition, sources of information from the past, previously thought to be unarchived, are brought into the present and begin to make up our identity. Facebook's recent introduction of timeline is an example of this [65]. Wall posts between friends which have always been available on the user's wall suddenly became easily accessible. As a result users felt their privacy had been breached. Palen and Dourish assert that managing this boundary is very much part of the management of our privacy. These boundaries are useful for examining privacy issues that occur in modern systems like online social networks. The boundaries are also relevant to 3rd party applications, particularly with social applications. Social applications not only share profile data from the user to application but also from user to user through the application.

Contextual integrity is another method used to examine privacy with 3rd party applications. Contextual integrity is useful for examining how certain types of information sharing can create privacy issues [42]. Contextual integrity includes the concept that all information sharing has certain expectations and that these are directly related to the context of sharing. All types of sharing include these expectations or norms. Nissenbaum asserts that there are two types of informational norms: norms of appropriateness and norms of information flow [42].

Norms of appropriateness are governed by what is appropriate to be shared in a

given context. For example, sharing personal information with friends is appropriate while sharing the same information with co-workers might not be appropriate. The second—norms of information flow or distribution—deals with how information moves or changes hands beyond the initial disclosure. An example of this is if a friend were to share personal information given in confidence with a stranger. The flow of information from your friend to the stranger causes a privacy breach because it was originally shared in a confidential context. Of course, over time we may find that informational norms change and therefore using contextual integrity to identify privacy problems could also change [42].

Lederer also notes that policies are highly influenced by the context and explains that users are unable to anticipate privacy needs ahead of time [36]. To exemplify this, a study by Rabkin examined bank challenge questions. Challenge questions are typically used to recover a lost password. He illustrated how common information disclosed within one context can become extremely problematic when the same information is used in a different context. In his study, he found that a large number of challenge questions for banks could be answered by using data gathered from a social network site shared within the context of a group of friends [49].

Since applications consume this information, it is possible for third parties to answer these questions in an automated fashion. It is clear that the informational norms of distribution do not match user expectations and therefore violate contextual integrity. I believe contextual integrity is more useful for identifying problematic privacy breaches with social applications and information flows. For example, users may not realize the context of sharing or the types of information being shared. Therefore,

they may behave in ways that contradict their desires.

Stutzman reviewed privacy changes over time on the social network site Facebook. He found that as time went on users became less willing to publicly disclose information on the site. At the same time as users were willing to share less publicly they interestingly began to share more privately [59]. Therefore, effective privacy management tools may have the ability to actually increase social network site utility not reduce it.

2.4 Information Flow

Lederer, et al. developed five pitfalls that ultimately cause privacy designs to fail [36]:

1. Obscuring potential information flow.
2. Obscuring actual information flow.
3. Emphasizing configuration over action.
4. Lack course-grained control.
5. Inhibiting established practice.

The first two pitfalls describe usability problems pertaining to information flow. As explained above, users must understand to whom their data is going and also to whom it can go in the future to understand the context. The third pitfall states that users should not be forced to do an excessive amount of configuration in order to protect their privacy. Instead, privacy should be designed as an integrated part to the

primary task. The fourth pitfall is the lack of a course-grained control. For example, systems might include a feature to disable all GPS access on a device regardless of applications that have been configured to allow access to the GPS [36].

Lederer's pitfalls and Nissenbaum's contextual integrity emphasize how important user understanding of information flows is to protect privacy in online social interaction. It is therefore necessary to find acceptable ways to present information about how information is shared to users. One of the contributions of my dissertation is to explore ways of notifying users and discover the frequency limit when they begin to become excessively annoying to the user.

Previous research examines how information flow can be used to improve user awareness and understanding. The hypothesis is that increased awareness of actual and potential information flows can aid users in making better disclosure decisions when sharing personal data. In work by Kowitz and Cranor, they experimented with creating a peripheral display that projected words being transmitted over an insecure wireless medium onto a wall [33]. While no significant changes were found in network traffic or users' perception of privacy, a qualitative analysis found that participants changed their expectation of privacy after seeing the information. As a result, their behavior may have changed to be more cautious of revealing sensitive information over the unencrypted wireless medium. However, it appears that users may not have learned that the medium was insecure, but instead feared the information being displayed on the wall.

In a similar study investigating informed consent, Friedman et al. developed a system, Mozilla Cookie Watcher, for making information flow—in this case cookies—

more visible [26]. Similar to Kowitz and Cranor they focused on a peripheral display design to avoid disrupting the user's primary task. Mozilla Cookie Watcher results suggest that even though users did not self report an increased understanding, they did actually gain an increased understanding of cookies. Another positive finding is that more than half of the participants reported exploring the mechanism. Unfortunately, some of those that did not explore it still had an incorrect or limited understanding of cookies.

Consolvo et al. created a WiFi Privacy Ticker similar to Kowitz and Cranor. In contrast to Kowitz and Cranor's implementation, Consolvo's peripheral display contains only information about the user and not all users of the WiFi network [12]. Additionally, instead of being publicly displayed, the ticker is visible only to the user. The results suggests the WiFi Privacy Ticker increases people's awareness of the dangers of unencrypted WiFi use. Quite surprisingly, the system was well liked and many participants asked to continue to use the software after the conclusion of the study.

These studies illustrate two important findings. First, users desire a clearer visual indication about their informational data flows. Secondly, when given better visual indicators, awareness is increased and behavior changes to reflect improved awareness. This work deals with information flow on non-mobile networks. Mobile information flow is different for a variety of reasons. Mobile devices are smaller with less screen space to use, data can move about the phone without actually leaving it, and these devices are not constantly in use yet still able to transmit.

Related research in mobile devices has taken steps to improve information flow with the Android application platform. Enck, et al. built an information flow tracking system for the Android platform. By modifying the Android API, the researchers taint the data and monitor the information flow within the device [20]. When data is sent to certain "sinks", such as the Internet or the device's storage card, the action is logged allowing for a complete track of the actual flows of data on the mobile device. While Enck, et al. are not necessarily concerned with the usability of presenting the information flow, they did provide a platform for researchers to build flow notification systems. Enck provides a simple application that displays a notice for each information flow event. Such a system is not practical because it generates an excessive amount of notices that eventually will frustrate users and cause them to stop paying attention. In work by Balebako et al, notifications and visualizations are used to try to inform users of data flows occurring on mobile devices, particularly the Android mobile phone. They find that users currently do not understand the data sharing occurring on the devices and are largely unaware that it is even happening [3].

2.5 Warnings and Notifications

Systems present notifications to users that are either active or passive. A passive notification includes notifications like the WiFi Privacy Ticker and WiFi wall. It can be anything that does not force the user to make a decision or react to the notice, such as the system simply changing the color of an indicator on the display. An active notification, in contrast, forces a user to respond to the notification. A good example of this is a popup notification that requires the user to click a button to continue

using the system.

In a study about phishing indicators, Wu et al. test phishing indicators and note that active notifications are far more effective than passive ones [64]. They note that managing the passive security indicator is not the primary task of the user and so it is easily missed. However, they support a previous suggestion by Don Norman that warns that overusing active indicators can be detrimental and ineffective because users eventually ignore and disable them [43]. Sunshine et al, examined SSL warnings and found that they could improve the warnings. Unfortunately, even the improved warnings still resulted in dangerous behavior and so they recommend preventing a warning from being shown when possible [61]. They also point out that small numbers of users may abandon the software being used for other software because of the warnings. Bravo-Lillo et al. studied different types of warnings and found that novice and advanced users considered risks at different times. They argue that to improve warnings we as designers should look at all the parts of the warning process and only produce them when necessary [9].

Egleman et al. expand this work with a study that compares the effectiveness of active and passive notifications to prevent phishing [18]. Their study examines notifications within the context of a warning model—Communication-Human Information Processing Model (C-HIP model). The C-HIP model attempts to describe the factors contributing to the overall effectiveness of a warning [14]. The C-HIP model includes variables such as behavior, motivation, attitudes, attention, stimuli, etc. making it useful to evaluate warning failures. The model outlines all the steps involved with users being presented with and responding to a warning. It helps to highlight the

importance of noticing, understanding, and reacting to warnings. In related work, Cranor develops a set of questions to examine steps in the C-HIP model [15] which Egleman also uses to test the effectiveness of active and passive warning notifications. Egleman's analysis suggests that active warnings far outperform passive warnings to prevent phishing attacks [18].

While it may seem that active indicators are more effective to warn users about information sharing, it should be noted that these studies are in the context of phishing where the risk is high and immediate action is necessary to prevent the attack. It is useful to understand how users react to increased habituation to active warnings and when users are better protected by active warnings. However, information flow in application platforms differs from general Internet use and passive warnings seem better suited for these platforms based on research previously mentioned by Consolvo [12] and Kowitz [33]. In Chapter 6, I test both active and passive notifications about information flow. My purpose is not to debate which warning method is generally more effective, but to determine which method works better for contextual notifications on application platforms.

CHAPTER 3: UNDERSTANDING THE USER

I begin by performing a qualitative semi-structured interview in order to better understand users. I gain an insight into how users perceive application platforms and the data sharing that occurs within application platforms. I will then be able to inform the design of future prototypes to help address those perceptions and make improvements in user policy configuration for application platforms.

3.1 Social Network Application Platform Usage

Online social network sites have hundreds of millions of users sharing a variety of personal information. In order to expand functionality, many of the leading social network sites have created a platform to allow 3rd party developers to create applications that utilize and enhance users' profiles. These applications have been widely successful, with very high adoption rates. Applications add value to the users of social network sites by allowing them to add additional content to their profiles, participate in games, share photos, and much more.

In order to provide engaging experiences, these platforms provide the ability for applications to consume users' profile data. This data commonly includes attributes like name, birthday, interests, and more. Applications are also commonly allowed to access this same information about friends of the user who have not yet directly accessed the application. The result is that a large set of data is made available to

3rd parties that might not necessarily need it. To demonstrate the risk to users, the BBC News developed a malicious application that had the potential to harvest large amounts of user profile data in just three hours [4].

To further complicate the matter, information on the user's friends is given without the adequate knowledge or consent of the user. For example, on Facebook a dialogue such as Figure 1 is shown to users on the first access to the application. I do not believe this message adequately conveys the data sharing that occurs. The message is generic at best and likely ignored because the message dialog looks identical each time. Previous research has shown that over 98% of users rarely read EULA's [28]. While this dialogue is much shorter, I think users are similarly going to become habituated to the dialogue box and ignore or not understand the information in it, in order to get to the primary task of accessing an application.

For example, consider Katie, a typical social network site user. Katie just signed on and sees that her friend Jason sent her a bumper sticker for her profile. Katie is interested in what the bumper sticker might be, so she clicks a button to see it. Since Katie has never used the bumper sticker application before she is prompted to review the agreement shown in Figure 1 and accept it. Katie quickly clicks the allow button in order to view what Jason has sent. Most of her profile data, as well as much of her friends' profile data, is now available for the application to query and use.

Existing application platforms also provide little in the way of tools for limiting the profile data that is given to applications. For example, in 2009 Facebook, one of the most popular social network sites, took an all or nothing approach, where in order to access an application the user often must consent to sharing most of her profile

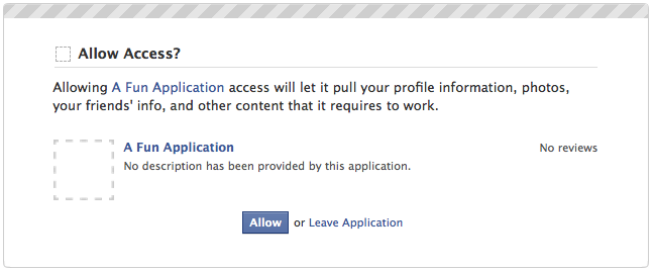

Figure 1: Adding an application (2009) - This screen warns users that they are permitting the application to access most profile attributes. This image has been modified with a fake application name.

data. This violates the well-known principle of least privilege [52]. A more recent implementation I explore in Section 4.1 allows for some selective permission granting.

Some users might expect that not using or visiting the application would protect themselves. Even this can violate expectations, as applications can request users' information by utilizing a friend who did authorize the application. On several sites, users are able to set a default policy on what information is accessible to applications their friends have authorized. These policies are highly permissive by default to encourage social use, and users may also not understand the purpose of these settings or set them appropriately.

It is not hard to believe that privacy problems are occurring given users' lack of control over their information and their difficulty in understanding the implications of application use. These motivations paved the way for researchers to begin examining alternative access control tools that may help to reduce the risk in utilizing these applications. Proposed solutions have ranged from models where no information about a user is given [24] to sharing generalizations of the personal attributes [55], to providing fine grained privacy settings [5]. While these solutions may address

some of the problems, there has been no study of users' motivations and perceptions to inform these efforts. What is still lacking is not an articulation of what problems are occurring, but an understanding of why. Understanding users' behavior and conceptions around social applications can help determine why users are struggling and aid in the design of effective and usable tools. I later draw from this knowledge to inform the design of access control prototypes for contextual access control.

Even digitally, people do not wish to share everything, with everyone [45]. As mentioned previously, I believe that users do not understand the context they are in on the application platform, and as such, cannot make privacy-appropriate decisions about information they do wish to share. This makes boundary maintenance difficult as users do not fully understand the scope of the information, actors involved or how the boundaries could be violated.

It is currently unclear where the balance lies between privacy and social value for these social applications. It is clear however, that little is known about why users use applications and how they perceive them operating. This understanding is crucial in order to suggest socially acceptable and understandable methods for using applications while maintaining adequate and desired privacy. This chapter examines how users are currently finding, adding, and removing applications as well as what data they perceive is shared as a result of this use. Results indicate that the social nature of these applications colors user perceptions, leading users to be very unaware of what data applications can access. I then discuss potential implications of these results and the impact on the creation of new privacy tools for social network site platforms. I draw on these findings in later chapters to inform the design of

mechanisms to aid users in making more informed decisions about data sharing on application platforms.

3.1.1 Background

This study focused on Facebook, as it was and still is the most widely used and oldest application platform. In 2009 when the study was conducted, Facebook reported having 400 million active users with 50% of those users signing in any given day. Today, Facebook claims 1.11 billion users are active on the site each month. Facebook was the first major social network site to offer an application platform to its users. When creating the API they also created a SQL-like querying language, aptly named FQL, that allows applications to access the vast majority of a user's profile information, including a user's name, birthday, education, work history, current location, photos, and more.

The platform has been very successful; at the time of the study Facebook reported that there were over 1 million developers who had created over 500,000 applications [22]. In addition to a large producing developer base there was also a very high adoption rate. Every month 70% of Facebook's users would use at least one application [22]. In fact, many of the features Facebook provides such as the marketplace, photo sharing, and events were written as Facebook applications by Facebook themselves.

Research on social network sites has shown that users tend to share a large amount of personal information including contact information, associations, photographs, and more [31, 58] which has motivated their friends to participate as well [30]. One study

at Carnegie Mellon University found that over 70% of students disclosed their picture, birthday, hometown, and high school information in their profiles [29]. Other research has also examined how various profile information on these sites is used for managing identity, communication, and strengthening relationships [8, 7]. Furthermore, because of the ever increasing amount of personal information, users are putting themselves at risk of identity theft, spear phishing, and stalking [56]. For example, in a 2008 study, Rabkin showed that many challenge questions used to reset or remember passwords can automatically be answered by using Facebook profile data [49].

Additional work has focused specifically on the application platforms I am discussing. Currently these applications are being given access to a large set of the information on a user's profile. In addition, applications are able to request similar information on user's friends. Yet a study by Felt et al of the top 150 Facebook applications showed that most only needed access to the users' name, list of friends, and networks they belonged to [24]. This suggests that as many as 91% of applications that are currently available have access to data they do not need, violating the principle of least privilege [24, 52]. This led Felt et al to propose a framework, privacy by proxy, that severely restricts the information available, yet may also restrict value in social applications. Others have suggested generalization of personal attributes in order to help users maintain privacy yet still be able to benefit from sharing information [55].

Research regarding the social application platform has primarily been concerned with the design of access control models. Yet while this work is trying to improve privacy, few have examined why applications are used and what users actually think

of them. This understanding is critical both to expand and enhance social applications as well as mitigate the risks currently associated with using them. This chapter aims to show how users interact with applications and examine users' comprehension of data sharing through the platform in order to inform the design of privacy management tools for application platforms.

3.1.2 Study Methodology

A semi-structured interview as well as a survey was administered to examine key points in an application's life-cycle: its discovery by users, addition and use, as well as possible subsequent removal. Users' conceptions of what information is shared, and with whom, and what they would be comfortable sharing with applications were also examined.

In the summer of 2008, 15 students were recruited from the city of Charlotte and from the University using paid advertising on Facebook and flyers to participate in an hour long semi-structured interview. Inclusion criteria for the study included the participant being 18 years of age or older, having an active Facebook account, and having used at least one application other than photos, marketplace, or events within the last three months. Interested participants were offered a small financial incentive in exchange for their participation. An additional 11 participants were recruited to take just the survey regarding data sharing with applications.

To conduct the interview, I sat down with participants in front of their Facebook profiles and framed questions around each of the applications the participant had authorized. I asked questions about how the participants found the application, why

they decided to authorize it, and if they had removed any applications, examining any factors that were important in making their decisions.

I asked participants if they had ever invited others to use the application, and why. In addition, I asked what they like most/least about the application, and any concerns they might have with it. I also asked participants to explain how applications fit into their usage of the social network site as well as what they consider a good application and a bad application. Since participants were asked to self report, they did not always remember details for every application. When this occurred, I simply moved on to the next application on their profiles.

After discussing all the applications, I administered a survey to participants to gather information about their demographics such as age, sex, race, and education. I used Westin's privacy segmentation index [34] as part of the survey to identify participants as privacy fundamentalists, pragmatists, or unconcerned. In addition, I used a series of questions with Likert scales to gauge participants' understanding of which profile attributes were shared with applications. I tested their understanding for applications their friends had authorized as well as applications they had authorized. I included items that are not on Facebook profiles like social security number and files to prevent participants from assuming the correct answer was everything and ensure that participants were truly answering the questions and not simply checking a column. This survey was administered after the bulk of the interview was completed to prevent a bias towards a privacy-centered interview. I then finished with additional interview questions regarding their conception of data sharing in applications. I later invited an additional 11 participants to complete the survey.

3.1.3 Results

I interviewed 15 participants with ages ranging from 18-25. Most participants were male (9) and undergraduates in college (13). This is a limitation of this study and future studies examining those who are outside of college would be necessary in order to determine if these findings hold for all groups of people.

Participants had an average of 13 applications each ($min = 3, max = 26$), for a total of 195 applications or cases. Participants were unable to recall any details in 36 cases ($cases = 18\%, participants = 10$). I did not include these applications in the analysis. While this is by no means the majority of applications I was still surprised by the number of applications that users could not remember any details of. I suspect at least some of these included cases where the participants might have been uncomfortable discussing the details of an application, such as an application used specifically for dating.

I will present the results of the interview, followed by the survey results. I will first discuss users' motivations around finding, adding, and removing applications. I also discuss users' concerns with applications as well as perceptions of data that is shared with applications. Finally, I discuss what information participants expressed they would be comfortable in sharing.

3.1.3.1 Finding Applications

There were several ways in which applications could spread, including a directory, friends inviting other friends, activity of others displayed in the user's news feed, and a box displaying the application on the user's profile. For applications the participant

remembered they could not remember the source 17% of the time. Most participants found an application by being invited by a friend ($cases = 39\%, participants = 14$). These invites were broken down into two different categories: those that were based on interaction, and those that were not. Invitations based on interaction inform the user that a friend has shared information or performed some action for them. In order to view the information, or return the interaction, the user must access the application. For example, a bumper sticker application allows users to send interesting bumper stickers to friends to be displayed on their profiles. The act of sending the bumper sticker notifies the other user and serves as an invite for those who have not yet authorized the application. As one participant stated:

> *P10:"I wanted to see what I had gotten and then I wanted to send them something back."*

That user can then send bumper stickers to other users, further propagating the application.

Other invitations are explicit messages, sent after a user is asked to choose a set of friends to invite. Participants reported being forced into inviting in order to use the application or get a result from it ($cases = 6\%, participants = 7$). Participants generally referred to this as spam, and many of them mentioned that they do not like invitations as a result ($participants = 9$). Participants mentioned a specific reason or objective for sending, despite seeing them as spam:

> *P15:"I invite the friends to add the application to earn extra points for something and for each person you invite you get a certain amount of the*

thing. When you join you earn that extra certain amount too."

Some participants stated that they would never invite or add an application that required them to invite because of too many unwanted invitations from others. In one case a participant expressed frustration with invitations:

P7:"I never do that because its annoying. Everyone does it to me and I'm like ignore ignore ignore ignore so I don't want to do that to other people."

However, this participant later went on to tell me about applications where he had sent things like bumper stickers or gifts to others. While this can also act as an invitation, the perception is less negative and seen as sharing rather than inviting.

Another common way participants found applications was browsing the profiles of their friends ($cases = 29\%, participants = 12$). Participants did not indicate that they were browsing the profiles for the purpose of finding applications. Instead, they indicated that in the course of viewing the profiles of their friends they found an application which was of interest to them. Participants mentioned several other ways they found applications: newsfeed, advertisements, and off-line communication.

Interestingly, browsing for an application in the application directory was the least cited method of finding applications for participants ($cases = 1\%, participants = 2$). Thus, the most commonly cited ways which participants found applications were social in nature.

3.1.3.2 Adding Applications

Participants cited a variety of reasons for adding applications, such as to play games or add interesting information to their profiles. A desire for additional inter-

action with others was the most common reason for adding an application ($cases = 25\%, participants = 14$). This interaction varied from viewing a picture made by a friend to playing an online chess game with friends. In several cases this interaction was a one time event in which a user sent a picture, opened a virtual gift, or viewed another user's interaction with an application. In many cases, participants mentioned that interaction was based on the need to meet new people or strengthen ties with existing social circles ($cases = 11\%, participants = 12$).

> *P14:"I added that because I knew I was gonna be in college and I put my courses on there so I could network with people that had my same classes."*

Thus, many applications are facilitating an interaction between two people, enhancing the social experience on the site. Participants reported liking the social interaction created by applications ($participants = 15$). They frequently discussed how they would compete, play, collaborate, or have contact with others through their applications.

Although social interaction seems to drive much of application use, interest ($cases = 22\%, participants = 14$) and entertainment ($cases = 21\%, participants = 10$) were also mentioned, frequently overlapping interaction with one another.

> *P9:"Cause I thought it would be cool to compare my best friends with it and my friends that were with it... kind of like a six degrees of separation thing almost."*

Applications revolving around interests commonly include categories like: sports, horoscopes, TV shows, music, movies, and more. Entertainment generally referred

to participants who mentioned adding the application to reduce boredom or fill time. Interaction, entertainment, and users' interests accounted for the majority of the motivation behind application authorizations.

3.1.3.3 Application Removal

Many participants reported regularly going in to remove applications (*participants* = 6) every couple of months. Their reasons for removing applications during this time included inactivity, cleaning up their profile, and an application that looked cluttered.

> *P14:"Whenever there is some applications that I feel are cluttering up my Facebook profile I just delete them basically. Just for aesthetic reasons."*

Participants occasionally removed an individual application immediately after authorizing it. These applications were removed for not meeting the user's expectations, taking up too much space or also having a cluttered look.

> *P12:"I just installed it, looked at it, realized I didn't like it and deleted it."*

Several participants mentioned receiving unwanted messages due to an application, which prompted removal. Participants also reported blocking applications, usually to stop recurring invitations from other users. However, this was not the main purpose of the blocking feature. The block feature was supposed to be used to prevent any information about the user from being consumed by an application. There was a separate button for blocking all invitations from an application. Participants did not appear to understand the true purpose of the blocking feature, although its use did have added benefits in that it prevented any further data sharing with the application.

3.1.3.4 General Concerns

I asked participants what concerns they had with each application they had authorized or removed, or concerns they might have in general. The concerns reported included profile space, others' perception, privacy, time investment, and inappropriate content. Privacy is specifically discussed later, for now I will discuss the other issues.

Participants reported profile space as the number one concern of having an application ($cases = 19\%, participants = 10$). They frequently questioned how much space would be required in order to have an application as well as if that space would later be recoverable.

> *P10:"Well I wanted to make sure it didn't take up too much room like the fun wall and it looked alright so I kept it and I liked it."*

Once deciding that they would authorize or keep an application, they would become concerned about where to put the application on the profile. Some participants even strategized by speculating on where best to place the application in order to complement their profile. This is less likely a concern today because, since this interview, applications on Facebook are not given a box on the user's profile automatically, and application boxes are now displayed on a separate tab on the profile.

Participants were also largely concerned with others' perceptions of them ($cases = 13\%, participants = 10$). This concern was based on the applications they had authorized or their activities within them. It was magnified at times based on stereotypes about groups of people. For example, a participant who referred to himself as

Mexican, had serious concerns about an application involving criminal activity. He mentioned that the activities he would engage in on an application involving 'mob activity' did not reflect his activities in the real world.

> *P2:"[People might say] alright well if he would do that on there... would he do that in real life."*

Other participants were concerned about the conversations others were having about them inside of applications. Examples of this are the 'hot or not' application, the compare friends application, or an application involving the buying and selling of friends. In cases like these, participants would authorize the application not to use it, but to monitor the perception others could have about them ($cases = 2\%, participants = 3$).

> *P9:"I wanted to see what everyone else had to say. I mean who doesn't want know what other people think about them."*

These participants added the application they were concerned about, then removed it from the profile and newsfeed, just to monitor others' activities about them occurring inside the application.

The time involved in using an application was another consideration for participants ($cases = 10\%, participants = 6$). Participants referred to the amount of time required to use the application almost exclusively as a waste of time. Many participants did not want to be bothered with an application that would take up too much of their time, instead preferring applications where they could decide how much time

was appropriate. Some cited their own habits about procrastination and how these applications would help them perpetuate this.

Some participants were concerned with the inappropriate content that might be generated due to applications ($cases = 4\%, participants = 3$). This included both viewing the inappropriate content and/or having it posted to their profile.

> *P7:"[A bad application has] dirty stuff. Umm they all do kind of. Bumper stickers has really nasty stuff..."*

One of the participants surprised me though by mentioning her use of an application in which she would poop on other friends' profiles, which she found humorous. Her biggest complaint: many others were unwilling to authorize it.

3.1.3.5 Potential Problems With Use

I asked participants what problems they had heard of or could see occurring as a result of using applications. I expected a large number of privacy-oriented responses, yet I was surprised with few of them. The leading response was that participants were unsure or had never heard of any problems ($participants = 7$). Being unsure was followed with concerns about identity theft ($participants = 3$). One participant simply casually mentioned it as part of a response with no further explanation as to why it might be applicable. The other mentioned that she might be redirected off the Facebook site where someone could attempt to steal her password. Another said that people are posting too much public information that could be consumed by others and result in a compromised identity.

Other miscellaneous problems included copyright violations, where the participants

Table 1: Perceived disclosure rates

	Applications I Installed			Applications Friends Installed		
	No	Don't Know	Yes	No	Don't Know	Yes
Name	4%	8%	88%	0%	16%	80%
Picture	8%	8%	84%	12%	16%	68%
Friends	8%	24%	68%	8%	24%	68%
Status	12%	24%	64%	16%	28%	56%
Hometown	20%	16%	60%	24%	40%	32%
Education History	20%	20%	60%	36%	32%	32%
Birthday	20%	24%	56%	20%	40%	40%
About Me	20%	24%	56%	28%	40%	32%
Events	20%	28%	52%	28%	32%	40%
Relationship Status	24%	24%	52%	36%	36%	28%
Photo Tags	36%	16%	48%	40%	32%	28%
Current Location	40%	8%	52%	40%	24%	36%
Work History	40%	20%	40%	40%	32%	28%
Marketplace Listings	40%	32%	28%	44%	36%	20%
Photo Albums	44%	12%	44%	28%	32%	40%
Significant Other	44%	20%	36%	44%	36%	16%

specifically mentioned someone profiting from remaking well-known games such as Scrabble. In addition, a concern that people would be able to make themselves appear to be a younger child rather than an adult. Malfunctioning applications, malicious applications, and concerns of SPAM where also mentioned once each.

3.1.3.6 Privacy

Participants said they desired being able to know what new information is exposed as a result of using an application. Many applications are quizzes, for example, to determine what type of beer or cartoon character people are most like. Participants using such a survey application can give others more information about themselves than they would be comfortable with sharing. It is very important to note this is not a concern about profile data being given to the application or its developers, but rather about additional information flows within their social spheres generated as a

result of an applications use.

> *P8: "[Surveys contain] what attributes you are looking for potential boyfriend, girlfriend, you know I really don't want everybody to read that."*

Only one of the participants was concerned with an application built with the purpose of maliciously collecting information about himself. This participant thought the way such an attack would be executed would include someone sending him pictures of a woman and starting a conversation in which he would divulge his personal information. He was not in any way referring to the automated collection of his profile data as a result of having authorized the application.

In general, participants' concerns stemmed from the social interactions that they participated in online, and managing identity and impressions appropriately through applications. They were largely unaware of privacy dangers due to data sharing with applications.

3.1.3.7 Who Gets Data

I administered a survey, described in Section 3.1.4, in which we asked participants to select which attributes they would be willing to show applications and indicate their understanding of what was already shared. When participants were asked why they indicated that they were willing to share each of these items they usually responded by saying that these items generally did not say too much about who they were. The information was not considered very personal. I also asked participants several questions including whom they envisioned they were giving this information to when they gave it to applications. It was clear from the responses that users were not very

sure.

Participants said that they would be giving personal information to friends, businesses, and developers. The most cited group participants said would be receiving information is their friends (*participants* = 5).

> *P5:"Well I mean based on my privacy settings I think my profile is limited view, so only my friends."*

Others felt the same way believing that only friends and possibly friends of friends would be given access to their data through the applications.

Some believed that Facebook would be using this information for advertising on Facebook. They believed that some companies would then be given this information in order to solicit them for products or services (*participants* = 3). Most seemed to view this in a strictly professional light, even citing that this information would not be used for identity theft. This point of view is itself paradoxical. Indeed, if Facebook is giving the information to applications why would it need to re-consume the data to provide ads? I believe this demonstrates users' lack of a mental model for data sharing and data flows on these platforms.

Three participants believed that the information would be given to the developers of the application (*participants* = 3). Even then however, two of those participants mentioned that this information would be used to "keep the application running" and was "mostly computer generated stuff". Thus, many participants who indicated in the survey that data was being shared, did not have an accurate conception of who potentially data could be shared with.

3.1.4 Survey Results

In addition to the 15 participants I recruited for the interview/survey, I invited another 11 participants from another study to complete just the survey. One participant did not fill out all the questions completely so their data was discarded. The remaining participants were predominantly female (15) and ages 18-25 (23). Using Westin's 2003 classification criterion [34] I separated participants into groups of fundamentalists (7), pragmatists (17), and unconcerned (1), although I found no differences in response based on this classification.

The survey asked participants to use a series of 7-point Likert scales to answer whether they thought each profile attribute was shared with applications they had authorized. After answering this series, I then asked if they thought applications their friends had authorized could see the same data using the same 7-point Likert scale. The Likert scale indicated that a value of 1 meant that the attribute was definitely not shared, a value of 4 meant they did not know, and a value of 7 meant they definitely knew it was.

For the purposes of reporting in Table 1 responses of 1-3 were grouped into one category of no, for responses of 4 the category was left as did not know, and 5-7 were categorized as yes. The table has been sorted by no responses, ascending for applications that the participant has installed on the profile. For applications the user has installed the answer should have been yes 100% of time, provided the information was filled in on their profile. None of the participants responded correctly for all data. Data sharing for friends' applications depends upon the user's application privacy

Table 2: Differences in perception of disclosure to applications

	Applications I Installed		Applications Friends Installed	
	M	*SD*	*M*	*SD*
Name	6.12	1.453	6.21	1.179
Picture	6.08	1.552	5.46	1.769
Birthday	5.08	2.272	4.52	1.982
Current Location	4.12	2.438	3.80	2.217
Photo Albums	4.12	2.351	4.24	2.067
Photo Tags	4.36	2.361	3.80	2.255
Hometown	5.29	2.177	4.25	2.027
Friends	5.60	1.683	5.36	1.68
Status	5.40	2.082	4.68	1.909
About Me	5.00	2.217	4.08	1.977
Education History	5.08	2.308	3.84	2.154
Work History	3.96	2.491	3.60	2.041
Events	5.08	2.159	4.44	2.162
Marketplace	3.68	2.155	3.28	1.815
Relationship Status	4.72	2.441	3.76	2.006
Significant Other	3.72	2.475	3.08	1.792

settings. During the study I viewed the settings for the 15 interview participants and 13 had default open policies, indicating that their data would be shared with any friends' applications. Thus, the correct answer is not 100%, but should still have a high percentage of yes responses provided the information is filled in on their profile.

Table 1 shows that participants generally agree that their name and picture are being shared with applications they have authorized. After these two attributes we see a large increase in the number of participants who specified that they do not know. There are two exceptions to this, current location and photo albums. In both cases, participants are often wrong, with many who did not feel or did not know the data fields are being shared with applications.

Table 2 reports the average score for each attribute. In this table, we see that in

most cases participants felt less sure that data is shared with friends' applications than their own. For example, more participants perceived that their hometown, status, about me, education history, and information about their significant other was given to applications they had authorized than applications their friends were using. It should be noted that the default policy protects information about who the user's significant other is from friends' applications (summer 2008). However all other attributes in Table 2 are not protected by the default policy, which was employed by 13 of the 15 interview participants.

I also asked participants to specify which attributes they would be willing to share with applications. The results are shown in Table 3. We can see that participants were almost universally willing to share their name and profile picture. This reflects current practices in Facebook applications. Many also were willing to share their education history. Since most of the participants were of the college age and using their university email in order to sign up for Facebook was a requirement, this may have impacted their willingness to share educational history.

There are still some troubling findings. First, participants do not have the correct mental model of information flow for applications. As such, it is highly likely that they are not referring to sharing these attributes with developers, but rather with other users. Still, there are many attributes that participants are unwilling to share. So even though they do not understand how far the information is being spread, they still do not want complete disclosure. This clearly does not match their behavior or the platform's design. Second, a few fields users are willing to share are particularly sensitive for security, namely their birthday and hometown. Again, users may believe

Table 3: Attributes participants were willing to share

Name	100%	Events	40%
Picture	80%	Time zone	40%
Education History	68%	Photo Albums	36%
Favorite Music	64%	Photo Tags	36%
Birthday	60%	Work History	36%
Friends	60%	Relationship Status	36%
Favorite Movies	60%	Political Views	36%
Status	56%	Current Location	32%
Sex	56%	Affiliations	32%
Favorite TV Shows	56%	Interested in	24%
Hometown	52%	Wall Count	24%
Interests	52%	Pages Your A Fan Of	24%
About Me	48%	Profile Update	20%
Activities	48%	Notes Count	20%
Favorite Quotes	48%	Listings	16%
Groups	48%	Looking for	16%
Religious Views	44%	Files on my computer	4%
Favorite Books	44%		

they are only sharing these with friends, or may not realize the risks in disclosing these data fields.

3.1.5 Discussion

The results of this study identified several important perceptions of social application usage that are now summarized:

- The use and perceptions of social applications are primarily driven by social interaction.

Participants of the study used applications for several purposes, reflecting the general goals of social network site use [8, 7, 30]. Some applications were used for identity management, to share additional information, affiliations, or interests. Others simply relieved temporary boredom and filled time. Yet interaction with others was still

key for most successful applications. Users mentioned many different ways they are interacting with other users, including challenging them, communicating with them, playing games, and strengthening their ties by sending virtual gifts back and forth. Participants' most favored and most used applications were those that generated repeated interactions with others. Thus, applications on social network sites are truly social in nature and interaction through the social network is driving the spread and use of applications.

The concerns that users do have about privacy are also based around interaction with others, similar to the general privacy concerns with overall social network site usage [57]. Thus users were concerned with monitoring and controlling the flow of information to the social network to further manage their identity.

- Users do not realize they are sharing so much information with applications, and do not wish to do so.

The interviews and surveys made clear that users do not realize they are sharing so much personal information. This lack of awareness is driven by the expectations that are created by the application platform. For example, particular parts of the user's profile such as name, picture and friends are commonly used within applications to customize them towards the user as well as provide the application with a way of communicating and interacting with the users' friends. Users can actually see that information flow within the application itself. As a result, most expected that this information can be accessed and were comfortable with that. Yet all potential data accesses were not visible and thus not understood.

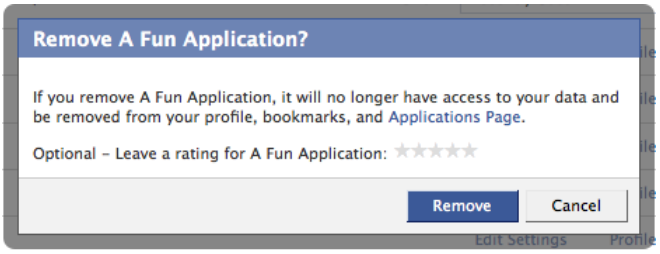

Figure 2: Removing an application - This screen incorrectly informs the user that removing the application will prevent access to their data. This image has been modified using a fake application name.

The survey and interviews indicated that users do not wish to share all their data. They only consent to sharing because it is impossible to interact with others using applications without doing so. Users did assume additional information is shared and are indeed informed of the data sharing every time they access a new application (Figure 1.) This warning does not seem to be influencing their mental models. I believe this highly relates to previous findings within the phishing domain, where users are more focused on their primary task than that of privacy or security [16]. As such, the screen warning of data sharing is simply skipped as something the user must do in order to accomplish their primary goal. Thus, relying on dialogue boxes and warnings is likely not sufficient to inform users of the data sharing.

Unfortunately for users who do read the warnings and notices and attempt to understand their content, they may still arrive at an incorrect model. For example, if they were to remove an application on Facebook they will see Figure 2. This warning clearly states that by removing the application it will no longer have access to their profile data. This is simply not true since if a friend has the application authorized it may still query the users' data. While this is a very site specific observation, it underscores the need for a consistent interface that helps to build an accurate mental

model. This leads to the next perception problem:

- Users do not realize that they are sharing so much information to the applications of their friends.

Users are completely unaware of the privacy dangers that the media and security community have identified, that of third party developers having access to so much information.

- Users do not realize that they are sharing information with third party developers outside of Facebook.

These misconceptions have important implications. Users shape their behavior by their perception of the social context [42]. Without a clear understanding of how information is shared, users can not weigh the risks and rewards of application usage and behave as desired. Users' underlying trust and motivation gained from their friends' application use, combined with their lack of understanding about the data flows, leads users into a very dangerous situation. There is a potential for a malicious application to quickly spread and harvest a large amount of valuable user data.

This study helps explain user behavior and motivates the need to take a different approach to the design of privacy mechanisms on a social network site application platform. The results indicate that platform changes or new tools are needed that reflect these results:

1. Social applications and privacy mechanisms for them need to reflect the social needs and motivations driving application usage.

2. Application platforms need to provide users with more control over the flows of information between users and their friends and developers.

3. Application platforms need to provide users with more accurate mental models of the sharing that occurs, both with their friends and with developers. Users need to understand all parties involved in the sharing of profile information.

External pressure has added to the haste in which changes must be made. In mid-2009 Canada's Privacy Commissioner objected to several of Facebook's practices including the third party use of user data described here [47]. Facebook agreed to address the disclosure of information to third party applications by giving users more control. This access control seems needed as simply restricting the platform from releasing all information is in direct contradiction with the needs and motives of both Facebook and its users. For example, horoscopes are popular, but could not operate without a user's birthday. Still, it is not trivial to properly determine the least amount of information necessary in order to perform a task and balance privacy needs against the social benefits of individual applications. Similarly in 2011, Facebook settled with the FTC for making data users thought would be protected public [65]. Facebook agreed to better keep their privacy promises as well as conduct independent privacy audits for the next 20 years.

The results of this study indicate that currently proposed solutions do not adequately address the misconceptions uncovered. For example, the study performed in Chapter 4 Section 4.1 examined behavior that occurred when the user was provided with fine grained access control for social applications. The proposed solution did

attempt to address the 3 guidelines outlined above with an interface to improve the user's mental model by presenting the concrete information to be shared, including information from a random friend, and allowing the user to limit disclosures to individual applications. The result led some users to take steps in order to protect their information. However, a significant number of users still did not, even in scenarios that were created to appear malicious.

The results in this section help to explain this behavior. While this proposed access control interface improved the transparency of information flow, it still did not convey who the third party is and the risks of sharing information outside of Facebook. Thus, simply adding fine grained access control is not sufficient. The application platform needs to make all data flows transparent so that users can build accurate mental models to make informed privacy and sharing decisions. And as we have already observed on Facebook and in other security domains such as phishing, warning dialogue boxes and messages are likely to be ignored. Instead, users build their mental models through interacting with applications. Thus, making decisions within the context of a running application may lead to more informed decisions and more motivated users.

While this particular study focused on Facebook, other platforms likely have similar uses and suffer similar misconceptions. There are two caveats to this. First the population studied on Facebook was generally in the 18-25 range and this does not represent the entire Facebook demographic. Additionally, Facebook friends tend to represent real world friends whereas other social network sites like Google+ commonly foster friendships between users who have never met. Future studies might examine

not only whether these findings hold across sites and different demographics, but also how users' perceptions are (mis)informed by the design of the platform and site interfaces.

3.1.6 Conclusion

Applications on social network sites are now ubiquitous. Users are accessing these third party services at an unprecedented rate, yet little research has examined what users think about accessing these applications and the overall platform. In this chapter I have contributed an understanding of users that use social network site application platforms. The results indicate that similar to social network site use, social interaction is driving the spread and use of applications, and coloring users' perceptions about privacy and information disclosure. Users are interacting, competing, communicating, and entertaining themselves. And their privacy concerns are centered around sharing data with other people on the social network, with almost no understanding of the data sharing that occurs with the application developers. The end result is that there are serious risks of applications maliciously harvesting profile information, and users are not truly understanding and consenting to these risks.

The solution to this problem may lie in better integrating privacy settings into the user interface during the primary interaction tasks, rather than only during the initial installation, conveying a more accurate mental model of data sharing throughout an application's lifecycle. In Section 5.1 and 5.2 I explore pushing the configuration of user policies into runtime and closer to the context of use. Results presented in this chapter of the users' motivations, concerns, and naive understanding of data sharing

can lead to improved platform designs, with alternative privacy and data disclosure mechanisms which provide transparent information flow.

In the next chapter I begin to explore contextual policy configuration. I begin by testing users reaction to contextual configuration at install. I provide users with an interface that is presented during an applications first use. I then push the decision out of install time and into runtime.

CHAPTER 4: INSTALL TIME CONTEXTUAL CONFIGURATION

In Chapter 3, I provided an analysis of the use and perceptions around the social network site platform. In this chapter, I address some of these issues using what I learned about user perception to inform design. I believe that by providing a more concrete mental model through increased contextual information users will make policies that more accurately reflect their desires.

I begin in Section 4.1 by providing users with context in the form of more explicit information about user data being consumed, including introducing users to types of data shared about their friends as a result of application use. I find that users who are motivated enough to configure policies do attempt to configure policies that reflect what might be considered reasonable use. They base these decisions off the provided explicit sharing information as well as the application name. However, participants did not utilize a feature which would provide them with additional reasoning on the information requested.

4.1 Install Time Configuration

In this section I present and formally define the current access control model used by social network site application platforms. I then attempt to improve upon this by introducing a new user-application policy to provide protection while still allowing desirable information access. I explore users' behaviors in utilizing a potential inter-

face for this new access control model. Finally, I discuss the potential implications for protecting personal information on social application platforms, extensions, and improvements to the current prototype.

4.1.1 SNS Architecture

Given the similarity of OpenSocial to Facebook's API circa 2009, it is possible to generalize the architecture of the access control. Since this is a generalization certain specific aspects of each platform may be lost. However, I attempt to generalize in a way that includes as much as possible. To begin, the subjects in online social networks are users, and applications act on behalf of users to provide services. Each user i maintains a user profile $Profile_i$, which is a representation of the user on the social network and includes information such as the user's name, birth date, contact information, email addresses, education, address, interests, photos, music, videos, blogs and many other attributes. In addition, the profile includes the user's set of friends and installed applications. Current social network architectures have enabled users to specify fine grain access control policies on the profile attributes to control user-to-user interactions. In a user-to-user interaction, the user making a request is referred to as the *viewer* and the requested user's profile is the *target*. The user-to-user interaction is governed by the user-to-user access control policy.

Applications can provide services to users and interact with users by combining and aggregating attributes from multiple target profiles. For example, an application that provides birthday gift recommendations for a user's friends needs to access the user's profile and her friends' profiles to access their birth dates and interests to be

able to make an appropriate and timely birthday gift recommendation. Social network frameworks provide a set of API's that enable third party applications to interface and access user profile attributes. We refer to the set of attributes accessible through the exposed APIs by S. For example, Facebook allows most profile data to be accessed. Their framework adopts what can be considered an all-or-nothing policy when it comes to applications. In other words, upon installing an application, the application is given access to all the attributes in S. When a user i installs application App_j, the application has access to user i's profile S and to the target profiles of user i's friends by acting as a viewer on behalf of user i. This introduces the ability for an application to access profile attributes of users that have not installed the application. To resolve this issue, current social network frameworks enable users to specify a default access control policy for all non-installed applications. For example, in Facebook the set of attributes available for an application App_j accessing the profile of target user t through the viewer user v can be described as follows:

$$R_{App_j,v,t} = \begin{cases} P_{t,v} \cap \ S, & t.App_j = 1 \\ P_{t,v} \cap D_t \cap S, & t.App_j = 0 \end{cases}$$

Where:

- $t.App_j$: Is a binary attribute, which is set to true if user t installed application App_j and false otherwise.
- S : Set of attributes available to the application through the API's exposed by the social network.
- $P_{t,v}$: Is the user-to-user profile policy for target user t and viewer user v. *Bob*

(the *target*) specifies a policy outlining what attributes *Alice* (the *viewer*) can see of his personal profile. For example, many sites let Bob choose whether to share aspects of his information publicly, with *"Only Friends"* or with *"Friends of Friends"*. Facebook provides users with the ability to set policies based on particular groups of friends as well. Note, that if viewer and target are the same user, then $P_{t,t} = S$ which is effectively everything provided by the API framework.

- D_t : Is the default application policy, the set of profile attributes specified by the target user t regarding what applications that are not installed by user t are allowed to access. The target, *Bob*, may also allow or restrict other profile attributes, e.g., Profile Picture, Education History, Work History, etc. For example, *Bob* may specify that in addition to his Name and Lists of Friends, his Profile Picture be accessible to applications that he did not install. This policy is applied to all applications that are not installed by *Bob*. The default policy provided by the sites is usually extremely permissive.

- $R_{App_j,v,t}$: Is the profile attributes for target user t accessible to application App_j through viewer user v.

For example, *Alice* (the *viewer*) installs application App_1. This action makes all of *Alice*'s user profile attributes available through the API framework (S) accessible to App_1 both for the user and her friends. *Alice*'s instantiation of App_1 requests *Bob*'s (the *target*) profile attributes (e.g., Profile Picture and Education History). If *Bob* already installed App_1 then this access is granted as App_1 will have access to *Bob*'s

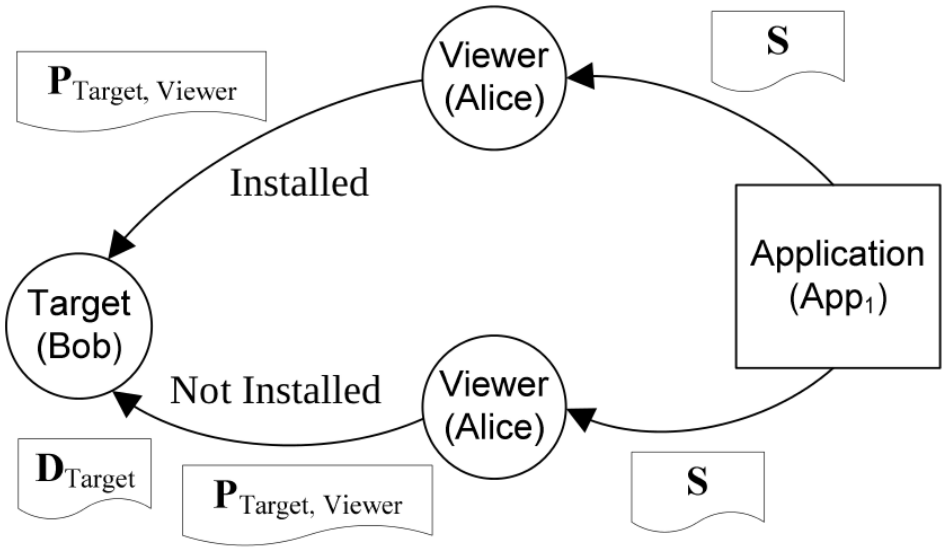

Figure 3: Facebook's application access control model

profile attributes in S; otherwise, if Bob did not install App_1 then the intersection of $Alice - Bob$'s user to user profile policy ($P_{Bob,Alice}$) and Bob's default application policy (D_{Bob}) are applied before access is granted to App_1. So consider for example an application that requests Bob's profile picture and education history, $P_{Bob,Alice}$ is setup to "Only Friends" ($Alice$ and Bob are friends) and D_{Bob} is set to all of Bob's user profile information except Education History. Therefore, the set of data returned to the application will just be Bob's Profile Picture. Figure 3 shows the different policies applied between the application, viewer and the target profile, in case the application is installed and not installed by the target.

When the viewer is the same as the target the $P_{t,v}$ policy is equal to everything, which implies that the installed application has access to all the user profile attributes exposed through the API framework. Furthermore, when the target users create the policy $P_{t,v}$, their intention is that they are granting access to other people, and not to installed applications accessing their profiles through their friends. Thus, this model

treats an application that is not installed by the viewer as one of the viewer's friends, and as the viewer himself if it is installed. This violates users expectations and provides little protection to users who wish to use any applications. This also makes it very easy for a seemingly useful application to maliciously access large amounts of personal information.

Given a target profile t, a set of viewers V, and an application App_j, the effective set of attributes accessible to the application through all the viewers $v \in V$ is given by:

$$R^{Eff}_{App_j,t} = \begin{cases} \left(\bigcup_{v \in V} P_{t,v} \right) \cap S, & t.App_j = 1 \\ \left(\bigcup_{v \in V} P_{t,v} \right) \cap D_t \cap S, & t.App_j = 0 \end{cases}$$

The term $\left(\bigcup_{v \in V} (P_{t,v}) \right)$ represents the effective set of attributes contributed by treating the application as a normal viewer and adopting the target to viewer profile policies. Note, that this model assumes that the application access is controlled solely by the user-to-user policy. I now present a more granular framework that would restrict the ability of such an exploit to occur.

4.1.2 A Granular Framework

I now introduce a social network application access control model that addresses the shortcomings of the current application access control model. The goal is to maintain as much of the current architecture as possible, as current platforms are already widely used. Facebook, for example, reported having over 24,000 applications by 400,000 developers in 2009. Since that time, the number has increased to nine million applications and websites in 2012. This model improves upon the model presented

in the previous section by introducing a user-to-application policy. This restricts the application to access only those attributes specified in the user-to-application policy. The model presents several restrictions on the access granted to the application in order to enforce user preferences and create a more secure execution environment.

4.1.2.1 User to Application Policy

The user-to-application policy is defined as:

- $A_{App_j,i}$: The user-application policy is the policy specified by any user i outlining the specific access restrictions for the application to the user's profile attributes. This policy has two effects. First, it restricts what information the application can access for the person who installs it. For example, Alice specifies what attributes application App_1 can access of her profile. This policy also supersedes the default policy when created for a particular application. Second, the policy also restricts what information the application can request on behalf of a user. This is referred to as *friendship based protection*. This behavior is summarized in the overall model below.

The proposed application access control model is summarized as follows:

$$R_{App_j,v,t} = \begin{cases} P_{t,v} \cap A_{App_j,v} \cap A_{App_j,t} \cap S, & t.App_j = 1 \\ P_{t,v} \cap A_{App_j,v} \cap D_t \cap S, & t.App_j = 0 \end{cases}$$

Where $t.App_j$, S, $P_{t,v}$, and D_t are the same as defined in Section 4.1.1 , and $A_{App_j,i}$ is the user application policy assigned by user i for application App_j. The application is installed by the viewer, and is used to access the profile of the target user. If the target user already installed the application then the attributes accessible to the

application are computed by intersecting the user-to-user policy of the target and the viewer, $P_{t,v}$, the viewer application policy $A_{App_j,v}$, and the target application policy $A_{App_j,t}$. Incorporating the target application policy enables the target to explicitly specify the set of attributes to share with the application.

By additionally including the viewer application policy it is possible to ensure that if an application App_j uses a viewer v to access the profile of a target t, then the policy specified by the viewer $A_{App_j,v}$ provides an additional set of restrictions protecting the target profile. Similarly, the viewer's application policy $A_{App_j,v}$ is incorporated, in addition to the user-to-user policy $P_{t,v}$ and the default target application policy D_t, when an application that has not been installed by the target attempts to access the target profile. Note that when the viewer equals the target, this model effectively reduces to the viewer application policy instead of everything as in the previous model. This is still subject to what is allowable by the API. In addition to providing a mechanism to enforce fine grain access control on applications, the proposed model also provides a mechanism for social collaboration in making policy decisions, where the viewer is able to influence the set of target attributes accessible by the application, which is a fundamental difference when compared to the current social network implementations, including Facebook.

Note that the model gives both the target and viewer finer control in deciding their exposed attributes to applications either installed or not installed. The model incorporates the user to user profile policy between the viewer and target whenever the application attempts to access the target profile, this is intuitive and builds on the fact that the application is using the viewer user to be able to access the target

profile. This ensures that if the target has restricted the viewer in some way the application will be restricted as well when making calls on behalf of the viewer.

Since the viewer application policy $A_{App_j,v}$ is intersected with other policies to compute the set of attributes available to the application, it follows that the viewer contributes to protecting the target profile, which is a form of *friendship based protection*. For example, target users who do not spend time and effort to configure restrictive default application policies can still be protected by their friends who do specify strict user-application policies.

For example, *Bob* (the target) is a careless user who does not pay close attention to protecting his profile privacy and leaves his default application policy to be very permissive such that D_{Bob} is the same as S. *Alice* (the viewer) is *Bob*'s friend, and she installed a horoscope application App_1 which is not installed by *Bob*. *Alice* is security conscious and she setup her application policy $A_{App_1,Alice}$ to allow access to only the Birth Date attributes.

The application will now only be able to access *Bob*'s birth date when requested by *Alice*, and nothing more. *Alice*'s awareness does not only protect her but it also protects *Bob*'s profile due to the fact that *Alice*'s policy $A_{App_1,Alice}$ is incorporated when the application App_1 attempts to access *Bob*'s profile. Again *Bob* is further protected if he has specifically prevented his birth date from being known by Alice. In that case App_1 would not be granted access to *Bob*'s date of birth. Figure 4, shows the different policies applied between the application, viewer and the target profile, in case the application is installed and not installed by the target.

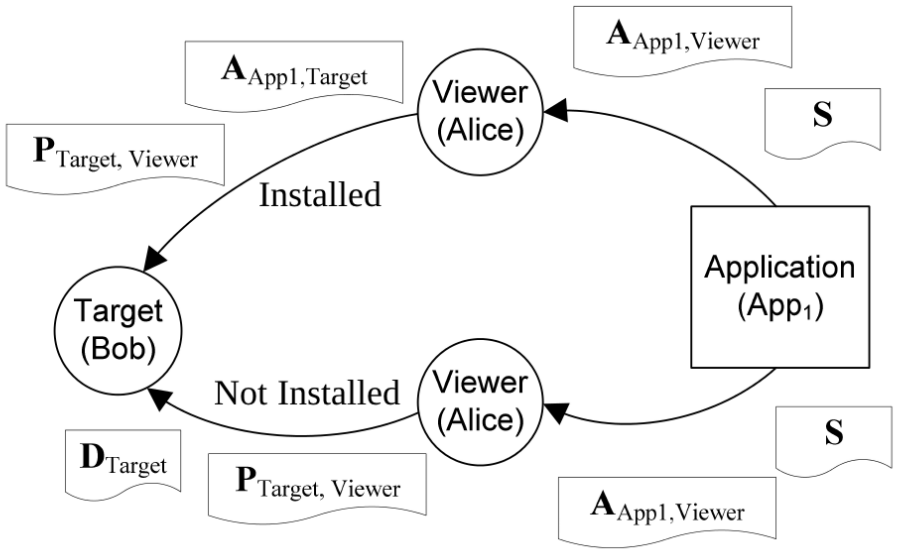

Figure 4: Proposed application access control model

In this model, friends' (viewers) application policies are incorporated when making access decisions, however the policy is upper bounded by the policies set by the target. The policy definitions for both installed and non installed applications are:

- $P_{t,v} \cap A_{App_j,v} \cap A_{App_j,t} \cap S \subseteq P_{t,v} \cap A_{App_j,t} \cap S$
- $P_{t,v} \cap A_{App_j,v} \cap D_t \cap S \subseteq \ P_{t,v} \cap D_t \cap S$

Thus the addition of the viewer's policy $A_{App_j,v}$ will only help limit the profile exposure and is upper bounded by the policies set by the target. In no case is it possible for a careless viewers' policy to reduce the effects of a protective targets' policy.

Given a target profile t, a set of viewers V, and an application App_j, the effective set of attributes accessible to the application through all the viewers $v \in V$ is given

by:

$$R^{Eff}_{App_j,t} = \begin{cases} \left(\bigcup_{v \in V} P_{t,v} \cap A_{App_j,v}\right) \cap A_{App_j,t} \cap S, & t.App_j = 1 \\ \left(\bigcup_{v \in V} P_{t,v} \cap A_{App_j,v}\right) \cap D_t \cap S, & t.App_j = 0 \end{cases}$$

The term $\left(\bigcup_{v \in V} (P_{t,v} \cap A_{App_j,v})\right)$ represents the effective set of attributes contributed by all the viewers, which represents the overall society contribution towards the target's effective application policy. It is possible but less likely in this model for a user-to-user policy $P_{t,v}$ to be violated by an application as a result of this collective information. This violation may be an unintentional one such as an incident involving the TopFriends application [10]. The application unintentionally allowed a user to see others partial profiles on Facebook that they should not have access to because the application cached $R^{Eff}_{App_j,t}$. This can be completely mitigated for a given target in the new model by a strong default policy, and if they install the application, a strong policy on that application. In the previous model users who have installed the application, even if briefly, have no way of preventing this.

4.1.2.2 Setting the User to Application Policy

This access control model requires three different policies to be created: a user-to-user policy, a default application policy, and the new user-to-application policy. Most social network sites already have controls for user-to-user policies, and other researchers are focusing on improvements to these controls [37]. So I will not address this policy in the current work. Many sites also already provide a default policy, which users can customize. For example, on Facebook, the default policy settings are one part of the general privacy settings, and users can select which pieces of

information they are willing to share or not share with applications they have not installed. Current default policies tend to be permissive to encourage sharing, but can be modified to be very restrictive.

This model adds a new policy, the user-application policy, $A_{App_j,i}$. The most basic method for creating this policy is to ask users to indicate their choices for each application they interact with. There are several times we could collect this data from users. First, we could ask users to indicate all of their preferences the first time they access or install an application. However, this may take too much time to specify a complex policy. Alternatively, an application could request confirmation for each individual access to a data item, or at least for the first access of that piece of information. I explore this idea in Chapter 5. Finally, users could have special configuration settings for each application they have accessed to view or modify their policies at any time.

There may be other more advanced methods for setting this user-application policy. For example, policies could allow the generalization of personal attributes [55], e.g., providing state of residence instead of a complete address. In such cases, the user could opt to allow the application to view generalizations of information on their behalf. This generalization might serve to be useful as the user can completely harness applications such as horoscopes and location-based services without exposing the specific data as to their whereabouts or actual date of birth. However, providing this capability creates an even more complicated set of controls and decisions for the user.

Another alternative would be to rely on a voter model, in which more informed users make decisions about the access policies for themselves and for others in the

Figure 5: Fine grained access control prototype

community. Those in the community could then apply a policy based on the number of votes. This would reduce the burden of choosing a policy for a large number of users, yet potentially still provide a useful policy. Of course the down side to such a scheme is that it is subject to gaming, in which several malicious accounts are used to defeat the system.

In this section I present the first exploration of this model, I chose to ask users to set all of their preferences upon installation, expanding upon the current screen for providing consent that most if not all platforms already employ. This initial study investigates a basic selection of data fields, without aggregation or voting. With an understanding of how well this method works and how users respond, I am able to explore more advanced methods in the next section.

4.1.2.3 Prototype User Interface

As a step towards understanding and refining the model, the first prototype focused on the user interface for specifying the user-application policy, and mechanisms to help users choose what information to share or protect. A screenshot of this interface is shown in Figure 5. Users view this interface on installation, the first time they access the application. The interface includes several features to aid users in making decisions about their policy. First, applications are allowed to request both required and optional information fields. The policy would then automatically restrict any information not requested. When adding or accessing an application for the first time, users would be shown the information that the application requests, and given the opportunity to modify these fields or opt out of the application.

In the interface, required fields are stared and the checkboxes grayed out for those fields. All requested fields are by default selected to encourage sharing, but the user could uncheck any of the optional fields to protect that information. This is also done in part to preserve the current nature of the applications for users who do not care to protect their privacy. By simply skipping to the continue button users are opting for a full disclosure, such as the one described in the previous model, with little extra time involved. These users however, are still offered greater protection as the application has constrained its own level of full disclosure to those attributes it needs to optimally work. And applications that request extensive amounts of information may get a second look from users seeing so much requested data.

To make each decision more concrete and clear I add more context by using the

user's own personal information that will be shared. This information is added to the interface, and also information from a randomly chosen friend. This provides a more accurate mental model of precisely what will or will not be shared with an application. In addition, it serves to catch the user's attention to try and prevent them from skipping this screen even though setting a policy is not their primary task.

A community indicator bar is also added, which is an indication of what percentage of the user's friends have allowed access for this application to each information field. Based on a separate evaluation, I have shown that this influences users' decisions regarding which pieces of information to share [6]. Finally, users could read an application-supplied description of how each of the information fields would be used by clicking for more information.

For this prototype, I emulated and implemented on the Facebook platform, as that is the most active application platform, and Facebook's API allows access to a greater amount of user data than other social networks. This also allowed me to more easily recruit users for the user study and provide an extremely realistic user experience.

The wording chosen and buttons to add an application are presented in a similar appearance to Facebook's 2008 application interface. Thus, users click on a "Continue" button to indicate consent and install the application, or "cancel" to leave the application and not set their policy. A method for creating a user-application policy when the user decides not to install the application has not been explored. One possible action is to immediately lock down the application policy to prevent any data from being shared. This certainly needs to be explored further.

The prototype shown in Figure 5 was implemented as a Facebook application.

Thus, users can "install" this prototype on their profile, giving the prototype access to their real profile information to add to the interface. In an actual implementation, the information fields required by an application can be modeled in XML to automatically generate this screen for each application. While a number of real applications were chosen for the prototype and user study, the information fields the applications would want were made up. In addition, the explanations and the values for the community indicator bar were also artificially created.

4.1.3 Study Methodology

The model proposed relies on user input to regulate the amount of exposure to both the user and the user's friends. The user study was developed to see to what extent users would set policies restricting or allowing that information on Facebook to be released to an application. To accomplish this a prototyped implementation previously described, which is depicted in Figure 5, was created. A user study was then performed, asking users to go through the process of adding a number of applications to their profile. Participants were told that I had created a new application container that I was testing, and that they would be adding applications to their profile using that container, which allows them to limit the information given.

Participants were also informed that they did not have to add any applications they were not comfortable with, and that I would help them remove the ones they did not want to keep at the conclusion of the study. In reality, no applications were added, the prototype was merely a simulation. As a result all the participants were completely protected from malicious applications that might have existed. I did want

users to believe they were actually adding the applications in order to make real privacy decisions, and all users appeared to take the task seriously and believe that they were adding these applications to their profile. After the tasks, participants were informed of this and the prototype application was removed from their profile.

All participants were first given a survey which collected demographic information, as well as used Westin's [34] method for categorizing people into privacy fundamentalists, pragmatists, or unconcerned. After participants completed the survey they signed into their own Facebook account and I installed the prototype. They were given no training on the interface.

The prototype presented users with eleven application installation scenarios. All applications were real Facebook applications to enhance realism, but the data fields that each application was "requesting" from the participants were created. Applications chosen may or may not be malicious, as there is no way of truly knowing; this was a major reason for choosing to simulate the experience. A wide variety of application functionality including horoscopes, flowers, games, and others was represented. All scenarios were the same for each user, except that the users' own profile information was inserted into the interface. Participants began with three training scenarios that asked for simple and basic information. The remaining scenarios included two that asked for information that was realistic within the context of the application, two that were unrealistic but seemingly harmless to allow, and two that requested sensitive information that did not fit the applications' context. There was also one application that asked for twenty-eight different attributes and another that did not ask for anything except for name and networks to which the participant belonged.

The realistic scenarios asked for data that made sense given the context of the application. For example, a book recommendation application requested the user's interests and books liked. An unrealistic one was something like a flowers application that asked for political interests. An out-of-context application asked for sensitive information such as a hometown or date of birth, but that request did not fit with the theme of the application and was not justified. Within each category, I presented users with one shorter and one longer scenario of requested information fields.

When participants completed all eleven scenarios, they were then interviewed and asked several questions including what they liked and disliked about this method of installing applications, who they thought they gave their data to, and feedback on other elements of the interface. I recorded both the computer screen and audio using usability software for later analysis.

4.1.4 Results

Participants were recruited through the use of Facebook ads on two campuses and in Charlotte. However due to the requirement to participate in the study on campus, all of the participants ended up being students. A total of seventeen participants were recruited from two local universities who were active members of Facebook. There were sixteen undergraduate students and one graduate student who participated. Eleven of the participants were females and six males. With the Westin survey, ten of these participants were classified as being privacy pragmatists: those that are concerned about privacy but are willing to trade it for benefits. Out of the others, five were classified as being fundamentalists: those who are distrustful of organizations

and worry about how their data is used. The remaining participant was classified as unconcerned.

In general, participants who added applications had two strategies they could use for releasing data to applications. They could use the application's default policy, releasing all required and optional data requested. Or they could use a custom policy, restricting some or all of the optional information. Some participants were minimalists and unchecked every optional field, others made case-by-case decisions.

Table 4 shows the number of applications added, number of times each participant used the strategies described, and average times, for their eleven scenarios. Users generally stuck with one strategy or the other, either accepting all information with little thought, or using custom policies to restrict information or not adding the application at all. Participants are grouped by their general strategy, as indicated by the time spent on each scenario, use of custom strategies, and behavior observed on the recordings. This is not to say these groups are statistically different, rather they are categorized based on observed behavior, both quantitative and qualitative. I refer to these two categories as motivated users and unmotivated users. This categorization was by no means meant to be definitive, but used to illustrate the differences in behavior and discuss results. Westin's survey had some correlation to these groupings. All fundamentalists fell into the motivated group, along with two pragmatists, and the one unconcerned.

A closer look at three participants labeled P6, P7, and P8 reveals that they used a minimalist strategy. This minimalist strategy shows that they are motivated in restricting applications from acquiring their data. P7 and P8 were both pragmatists

Table 4: Policies configured

Motivated	**Privacy**	**Total Added**	**Default Policy**	**Custom Policy**	**Avg Time**
	P1	10	8	2	0:49
	P2	10	3	7	0:22
	P3	7	1	6	0:23
	P4	4	4	0	0:11
	P5	5	4	1	0:26
	P6	11	3	8	0:16
	P7	11	2	9	0:19
	P8	10	3	7	0:21
	Average	**8.50 / 11**	**3.50 / 8.5**	**5.00 / 8.5**	**0:23**
Unmotivated	**P9**	11	11	0	0:16
	P10	11	7	4	0:14
	P11	11	11	0	0:11
	P12	11	11	0	0:09
	P13	5	5	0	0:08
	P14	11	11	0	0:08
	P15	11	11	0	0:05
	P16	3	3	0	0:12
	P17	8	5	3	0:12
	Average	**9.11 / 11**	**8.33 / 9.11**	**0.78 / 9.11**	**0:10**

Table 5: Comparison of groups disclosure

Privacy	Total Added	Default Policy	Custom Policy	Avg Time
Motivated	77.27%	31.82%	45.45%	0:23
Unmotivated	82.83%	75.76%	7.07%	0:10

and P6 was the sole participant categorized as unconcerned. Others like P13 exhibit behavior that in Table 4 would suggest they belong to the motivated group. P13 however made the decision not to install applications based on if she had seen them before, as she did not want more stuff on her profile. This is further confirmation that participants also decide to not to add applications for social reasons, not just for privacy reasons, as discussed in Section 3.1.3.2. Analysis of the recordings left P16's intentions unclear. As such we grouped P16 with the unmotivated group based on Westins' classification as well as the similar timing data.

The data in Table 5 summarizes the results for the two groups, and shows that there are stark differences between the motivated and unmotivated users. Motivated participants added fewer applications, did not grant applications access to the attributes they requested, set custom policies far more often and on average took more than twice the time to add applications as their unmotivated counterparts.

Specifically, unmotivated participants used whatever the applications asked for 76% of the time they installed, while the motivated users only used that strategy 32% of the time they installed an application. In fact, the unmotivated group only set a default policy 7% of the time. The motivated group instead set a custom policy 45% of the time, or allowed the application access to some subset of requested attributes. The unmotivated group rarely used these strategies. This accounts for the

timing difference between the two groups, the motivated group spent time reading and unchecking information fields, while the unmotivated group did not.

Table 6 shows the results of these strategies on four of the eleven different scenarios or tasks. These are representative of two context appropriate and two context inappropriate scenarios. Horoscopes and Books iLike asked for information that seemed somewhat appropriate to the scope of the application. However SpringWater asked for sensitive information, provided no explanation for data it required, and the application's description was written in Hebrew. The HighSchoolBuddies scenario seemed less suspicious, but still asked for several pieces of information that did not fit the application and were not justified.

For both context appropriate scenarios, the motivated and unmotivated groups had high rates of adding the application and high rates of allowing access to the data. The exception was hometown information for the horoscope application, which might not have made as much sense as originally planned. The description stated that the hometown information was used to see what stars the participant had been born under. Motivated users usually restricted this information.

SpringWater, a context inappropriate scenario, requested a variety of sensitive information. The motivated group mostly refused to add this application, while the unmotivated group had a startlingly high rate of adding the application and allowing their information to be accessed. However, a significantly different number of motivated participants were willing to add the HighSchoolBuddies application as a result of not being forced to release personal information, something that the SpringWater scenario required. Instead, the motivated users restricted the optional fields to

Table 6: Comparison of disclosure scenarios

	Context Appropriate				**Context Inappropriate**			
	Horoscope		**Books iLike**		**SpringWater**		**H.S. Buddies**	
	Motivated (M)	Unmotivated (U)	M	U	M	U	M	U
Added	100.00%	88.8%	87.5%	88.8%	37.5%	77.7%	62.5%	88.8%
Birthday	*100.00%	*88.8%	-	-	*37.5%	*77.7%	-	-
Hometown	37.5%	66.6%	-	-	*37.5%	*77.7%	-	-
Location	-	-	-	-	0.00%	77.7%	25%	88.8%
Work Info	-	-	-	-	*37.5%	*77.7%	12.5%	88.8%
High School	-	-	-	-	0.00%	77.7%	25%	77.7%
Books	-	-	62.5%	88.8%	-	-	-	-
* Indicates required data to install								

protect themselves.

In general the users in the motivated group protected their personal data more than the unmotivated group. Other scenarios had similar results where the unmotivated participants just accepted the defaults. As Table 6 shows, there were, however, still times when the motivated group disclosed personal data to applications which might not have needed it. When reviewing the videos, in some cases motivated users had not entered the data into Facebook, so they might have mistakenly considered themselves protected. For example, if a participant never supplied their favorite books to Facebook they would not consider that option while installing. There is danger in this strategy which I discuss later. In addition, the three participants P6, P7, and P8 used a minimalistic strategy to account for the privacy decision and may have incorrectly assumed that their information was being protected even though the application may have required sensitive information it did not need.

On applications that asked for data which it did not seem to need but could be harmless, the motivated group of users used the custom policy to stop that disclosure. For example, in another scenario a flowers application requested participants' political affiliations. Seven of the eight motivated participants added the application and five of those set a policy to remove political affiliations from being disclosed. In contrast unmotivated participants who installed the flowers application allowed the disclosure with none restricting it.

To summarize the results, participants generally behaved in two different ways. The first group made efforts to minimize the impact of installing an application as well as attempted to make informed decisions about the application's data usage in

context. For these users, the interface and the model would work reasonably well. The second group did not, and generally accepted the default policy supplied by the application.

I did not evaluate the potential effectiveness of the community bars in this study. There was too little use of custom policies, and thus too small of an impact from the bar to be observed. Instead, I evaluated the community bar separately isolating factors that may contribute to its effectiveness. I found that given a sufficiently strong negative indicator behaviour can be changed [6].

4.1.5 Discussion

The model presented seeks to find a balance between sharing information with applications and protection of user privacy. As such, the model does not offer perfect protection. The results presented here inform the next iteration of this model presented in the next section and may inspire others to do the same. I feel users should be able to choose to share information with applications if they desire, but this may lead to inadvertent disclosures. However, the model does potentially reduce the risks by restricting the amount of information disclosed both for individuals using the application and their community of friends. This model is thus highly dependent on the success of setting an appropriate user-application policy. In this section users were asked to set this policy on installation. In the next section I ask users to set this policy at runtime.

A limitation of this study is that participants were aware that they were participating in a study, and may have made more restrictive or careful decisions than they

would make when faced with installing an application that they are really interested in. Thus, further work is needed to verify whether this model would work in deployment. The prototype attempted to make the scenarios as realistic as possible to have participants believe they were installing these applications.

A number of users were motivated to take the time to review the interface and further restrict the information they shared. As a result, these users reduced their threat, yet still did share some useful information with applications. In addition, they would have added those same protections to their friends and informed the greater community of their choices. This set of users would likely also be motivated to set a restricted default policy, further protecting them from all unknown applications. Thus, the model and the interface would achieve the desired balance for social applications for those who truly care about their privacy.

However, there are still improvements that could be made. There were still instances of motivated users sharing sensitive data with applications outside a seemingly appropriate context. And while several users were minimalists, and restricted all optional fields, they did still allow sensitive information to be shared when it was required by the application. Thus, minimalists may feel more protected, but could still release their data without full consideration. In the next chapter, Chapter 5, I attempt to improve this by increasing the amount of information available through context of use at runtime.

Many users monitor their privacy on their profile by not entering information in the first place, or entering in false information. Thus, the application policy for that data is irrelevant, as it would not impact their real information. However, if users

later update their profiles, they may then accidentally disclose that information to applications already installed. This problem is reflected in the current interface as non-disclosed user attributes result in blank spaces within the interface. There were also several cases observed where a custom policy was set but a personal attribute such as date of birth or hometown was not allowed to be shared with applications or had not been filled in on their profile. In many of these cases users would not make a decision to restrict this attribute.

About half of the participants were not motivated enough to take the time to set their policy or consider whether to add an application in the first place. This unmotivated group left themselves open to disclosures of very personal information, and as a result, also exposed their friends with weak default policies. Arguably the only ones they are putting at risk are others like them who are not going to be willing or do not want to set either the default policy D_v or the application policy $A_{App_j,v}$. So the risky users will be hurting other risk taking users.

Given the population of these social network sites, types of data involved, and ease with which the information can be collected, I find this to be unacceptable. With 1 billion active users currently on Facebook, even if only 10 percent of users engage in risky behavior, that leaves 100 million users subject to greater risks. That is still large number of users and the numbers continue to grow every day. A better social feedback mechanism might be needed to better inform those users when they are making a decision that is contrary to the community's decisions.

There are several reasons users may not be taking the time to set these policies. I believe that users are unlikely to fully understand the risks associated with the

disclosure of certain pieces of information, such as their hometown or work history. Thus, they may feel comfortable sharing that information, when perhaps they should not be. This leads me to believe that we either need to increase user motivation to influence more users to more appropriately set their application policies, or reconsider the means in which we set $A_{App_j,v}$ and D_v. In Section 5.1 and Section 5.2 I explore policy configuration at runtime which may provide a better means to configure $A_{App_j,v}$. The model presented in this section does not dictate how this is done, and with appropriate policies would effectively preserve the users' privacy and security and promote the social value of applications. This is a tricky balance though because if the policies are too restrictive, some applications will be effectively crippled. If the policies are too open, users who do not change the policies will expose themselves and be the weakest links in their social networks. Thus, more work needs to be done to explore appropriate mechanisms for setting these policies.

In a much larger sense the results in this section imply that any access control model that relies on users determining their own policies may be subject to similar results. That is, protecting those who are most concerned but inadequately protecting those who are less concerned or less aware of the risks. Perhaps by providing a better understanding of user information flow users can become educated on the types and frequency of data shared.

It is also possible but less likely in this model for a user-to-user policy $P_{t,v}$ to be violated by an application as a result of this collective information, such as the incident involving the TopFriends application [10] on Facebook. The application unintentionally allowed a user to see profile information of others that they should

not have access to, because the application cached user data $R^{Eff}_{App_j,t}$. This can be completely mitigated for a given target in the model by a strong default policy, and if they install the application, a strong policy on that application. In the previous model users who have installed the application, even if briefly, have no way of mitigating this potential attack.

4.1.6 Conclusion

This section presented a new model for access control for applications on social network sites. This model adds a new user-application policy which greatly restricts the information applications can access, while still allowing useful and desired information sharing. The model also adds few changes to existing application platforms, enabling this approach to be easily adopted. A new interface helps to make this model usable by providing concrete context of the information that is being shared. The success of the model, however, depends upon appropriately setting the new user-application policy.

This first prototype explored having users set this policy upon installation. In Section 5.1 and Section 5.2, I explore policy configuration at runtime. At runtime additional information is available such as the context of data use. In this study, during install time I observed motivated users who configured policies based on the name of the application provided to them. Those who were motivated to protect their information, were able to easily use the interface to set their policy and protect much of their personal information. These users would be able to reap the benefits of social applications and remain reasonably protected against the threats posed by them. In

fact, a user could even be using a malicious application without exposing themself to any risk. However, this interface did not work for half the users, who were still unmotivated to protect themselves, and in turn, their friends. For these users, more exploration is needed to explore alternate means to set their application policies.

Despite the mixed user study results, the overall model is favorable for several reasons. First, it lessens the likelihood that someone who is concerned about an application accessing his or her data will accidentally disclose that information. Second, it prevents applications that are poorly coded from being exploited by hackers to gain the entire set of a user's data. It also reduces the ability of an application to record all the user attributes and accidentally disclose them to others where it might violate the user-to-user $P_{t,v}$ policy, as was done recently by a popular Facebook application. For those who are very concerned, data is not ever released without their explicit consent. And those who are less concerned can opt to release everything just as they are able to in the current architectures.

4.1.7 Acknowledgements

I would like to thank Gorrell Cheek for his input and help with data collection on this project. I would also like to thank Gabrielle Bankston who assisted in data collection.

CHAPTER 5: RUNTIME CONTEXTUAL CONFIGURATION

In this chapter I examine contextual configuration at runtime. These studies build on findings in Section 4.1 by attempting to provide even more context and evaluate user decision making at runtime.

In Section 4.1, I provided evidence that increasing awareness of the sharing context improves motivation for users to make appropriate data sharing decisions. Participants interacted with a set of application scenarios and made decisions about what to share with each scenario, if anything. Participants were provided with the application name and a list of the requested permissions. To provide increased context, I utilized the users' data to illustrate the sharing that would be taking place within the application, included additional information on how each permission was relevant to the application, and included a reference to the application developer.

Participants largely ignored the additional justification provided for each permission. Instead, results suggested that motivated users were attempting to make access control decisions based on the application's name. By configuring the application at install time users are asked to holistically consider the policy of the application including features they may never use. By moving that decision into runtime I can include all the information available to users at install time and include additional context to help users make more informed decisions.

At runtime, additional types of information are available. For example, consider the Pandora internet radio problem, where users do not understand the reasoning for access to their contacts at install time. At runtime these same users would not have been asked to consider this decision until prompting the application to share a radio station with someone. Since they initiated the action they can further reason that the contacts are needed in order to identify who they want to share with. Likewise, if they indicate that they wish to attend an event that the application advertises, then the user would further be able to reason that access to their calendar is needed.

Configuration during runtime also has several other advantages. Users can run applications on their device that truly adhere to the principle of least privilege. Applications would not need every permission granted in order provide for partial application usage. For example, users who wish to listen to Pandora but not share stations would not have to grant access to the contacts. Furthermore, by binding policy decisions to the screen users can observe what benefits the application provides that they are missing out on and better weigh the risk vs benefit tradeoffs.

In this chapter, I explore providing the user with contextual runtime configuration. I provide an increased context by showing an onscreen icon and visualization to inform users of the need to configure the permissions. By default policies are set to deny all access and so users must allow access to use the functionality. In this chapter I seek to determine if contextual runtime access control leads to better policy creation. I also seek to understand how distracting runtime configuration is, if it results in increased memorability, and how users react to it.

5.1 Mobile Contextual Configuration

I chose the Android mobile phone platform to begin exploring runtime contextual access control. In regards to access control, Android's platform is similar to the social network site application platforms in that it asks users to consider policies at install time. During the application install process, users see the permissions being asked for such as contacts or access to the camera. However, mobile devices with operating systems like Android do not require the friendship-based protection discussed in Section 4.1.2.1. Generally users are considered to own all the data on their devices and there is no concept of a friends' phone sharing information located on their phone.

5.1.1 Prototype

In order to test runtime configuration of access control policies, two prototype applications were outfitted with contextual access control. The first application, Divide and Conquer, was an Android open source example that we added permissions to as well as created a profile feature to track the user and the user's friends' high scores. This application requested several permissions including access to the vibrator, contacts, Internet and phone identity. We bound each permission closest to their context of use. For example, the vibrator was bound to the game play area with a small icon in the bottom right hand corner and the Internet permission to the save profile area on the profile screen.

The second application, called LifeJournal, allowed users to take audio or picture notes and tag them with contacts, locations, notes, and more. Again, the permissions

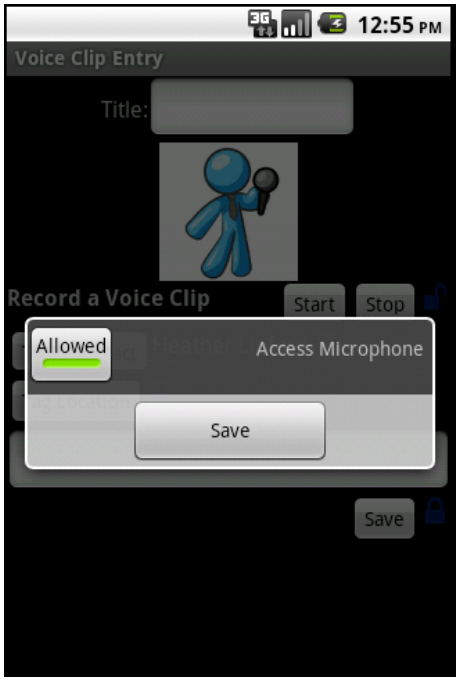

Figure 6: An example of contextual binding with the LifeJournal application.

are bound as close to the context of use as possible. Figure 6 shows an example of contextually binding access to the microphone to the creation of an audio note.

The two prototypes using runtime contextual access control differed in terms of functionality and requested permissions. Both prototypes allowed participants to set permissions at runtime. For both prototypes, permission dialogs could be initiated in two different ways. The first way was for the participant to touch the persistent icon on the user interface that indicated whether or not that set of permissions had been allowed. If the icon represented a single permission its state reflected that decision with locked or unlocked respectively. If the icon represented more than one permission then it shows unlocked if any of the permissions it represented had been allowed. Future iterations would likely contain three states reflecting the partially granted permissions.

The second way of initiating the permissions dialog was by utilizing functionality within the application. In both prototype applications, button listener events that required special permissions to be granted resulted in the production of the on screen configuration dialog. Once the dialog was rendered, the participant could configure their desired policy and select save. At that point the user's policy was updated and the execution of the application resumed. Once the policy had been configured, if those permissions were needed again, there was no need to re-prompt the user. As such this second way of initiating the dialog was automatically bypassed if all permission requirements were already met.

In some cases, a permission was required in order for execution to resume. For example, when a participant wanted to record audio, the microphone permission absolutely must have been allowed. In these cases, I notified participants using the Android toast notification system of which permission was required in order to move on in that step. I added this feature when pressed for time after some of the pilot participants were unable to figure out why they could not save a journal entry. This turned out to be a useful method to notify participants when required permissions were necessary to continue. It is a method I reuse in the next prototype, again with success.

In an actual deployed system I anticipate that there would be other places to configure the access control policy and review the policy in a holistic way. One example currently in use, is the Android application management tool. Users can use this tool to remove applications, move them to an alternative storage location, review permissions, and so on. However, my desire is to evaluate the contextual aspect of

configuration and I therefore did not include any of these features in the prototype.

5.1.2 Study Methodology

I designed this study to test several aspects of contextually configuring access control at runtime in comparison to standard configuration at install time. With this study, I attempted to explore the following questions:

- Effort - What is the difference in how difficult or complicated it is to configure access control in this way versus the standard installation screen provided by the platform.
- Comfort - How comfortable were users in making decisions? Did they find this contextual method of access control annoying?
- Comprehension - How well did users understand what was being asked of them?
- Distraction - Was configuring access control in this way distracting? All the tasks were designed to have users encounter access control during use which may interrupt their primary task.
- Memorability - Does providing access control in a contextual manner allow users to better remember the permissions they have configured for a program?

I designed a within subjects user study to examine these different research questions. Participants interacted with three different applications: the Divide and Conquer application, LifeJournal application, and CatchNotes. Both Divide and Conquer and LifeJournal were my prototype applications and participants would configure policies while interacting with the application during runtime. These two runtime

configuration applications did not request the exact same set of permissions and were tested separately for memorability. CatchNotes was an actual application available for free from the Android Market. Its features were extremely similar to the ones offered in my prototype LifeJournal. Because this application was from the Android Market, it was configured by using the standard install time installation method on the Android Platform.

In order to maintain a consistent study environment, I used a rooted Motorola Droid Android phone. I configured a custom phone image that contained: the study software, a configured test google account, a set of fictional contacts, and a few random images from the Internet. In the image, I configured the airport mode to be on, effectively disabling any phone calls, text messages, or other regular phone activity that might disturb the participants. I then enabled and configured the wireless Internet of the device. This allowed participants to use the Android Market to install the CatchNotes application. Before each study session the phone was restored to this original image so that participants' experience would be as similar as possible.

Participants were briefly shown how to use the Android phone they were given at the beginning of the study session. I reviewed how to turn on and off the phone, unlock the phone, and how to access the study related applications. I instructed participants on how to find the Catch Notes application using the Android Market and how to use both prototype applications. I then showed them the task sheet and survey software used to record the session. Finally, I gave participants a chance to ask any questions that they might have before starting the study tasks.

Participants' use of the standard install time configuration and both contextual runtime configurations were counterbalanced to reduce order effects. Participants performed a set of tasks on each of the interfaces. Tasks included creating audio journal entries, picture journal entries, playing the game, and configuring a profile. As participants completed their tasks across the different applications, they were required to interact with both the install time and the contextual runtime access control interfaces. After each task was complete, participants paused for a brief break to fill out a short survey. Participants were asked about the effort required to configure the access control they encountered in that task. Participants were also asked how comfortable they were configuring the access control. Finally, participants were asked to describe any other issues they wanted to discuss regarding the application and task.

After completing all three tasks, I asked the participants to take three short quizzes to assess memorability. For each application, the participant attempted to identify all permissions that they had been asked for while performing the study tasks. The test contained a list of 16 permissions for each application that asked participants to identify which of the permissions from the list were or were not requested by the application. The list of permissions on the quiz was the same for each application. The quiz included permissions requested by the application during the tasks as well as other permissions not requested during the task. Each permission item was scored as either a correct or incorrect response. Correct responses indicated that a permission was asked for when it was or was not asked for when it was not. Any other answer was deemed incorrect.

After completing the three quizzes, the participants completed demographic questions and three questions pertaining to the Westin Privacy Segmentation Index. I then sat down with participants and asked them to rate and describe their overall experience, understanding of access requested by the applications, and respond to the complexity of the access control configuration. I also asked participants to describe how annoying the configuration was, and the level of control that they felt. After completion of the study participants were compensated with a $15 Amazon e-gift card.

5.1.3 Results

I recruited 14 participants for the study using flyers and a recruiting table setup in the student union. They reported ages ranging from 18-28 with ($M = 21$). Most participants ($n = 11$) reported having some college or an undergraduate degree and some reported having at least some ($n = 3$) graduate coursework or degree. Most participants ($n = 11$) reported being smartphone users although some ($n = 2$) reported not using a smartphone at all. Only a few participants ($n = 3$) reported using Android daily. Most of the participants ($n = 8$) reported having not used Android at all. Participants self reported their ethnicities as predominantly White ($n = 8$) and Africian American ($n = 4$). One participant reported themselves as Hispanic ($n = 1$) and the other Middle Eastern ($n = 1$). Participants in this study were predominantly male ($n = 10$) compared to female ($n = 4$). Using Westins Privacy Segmentation Index I found participants to be mostly unconcerned ($n = 7$) and pragmatist ($n = 6$). One participant was classified as a fundamentalist ($n = 1$).

Overall most participants ($n = 11$) said their experience with the contextual access control was positive. Participants cited the ease of selecting permissions ($n = 3$) as well as liking increased control ($n = 4$). Some participants ($n = 3$) mentioned being initially confused, however, all who mentioned being confused described their overall experience as positive. A few participants ($n = 2$) disliked the repetition of the access control and one participant ($n = 1$) disliked the extra time associated with using the access control. Of the participants that described the experience as positive, two participants ($n = 2$) described the access control interaction as annoying.

Most participants ($n = 12$) felt it was easy to understand that applications were requesting permission using contextual access control. Many of these participants ($n = 11$) said it was obvious because the notification made it clear when the application displayed the access control message. One participant ($n = 1$) found it easy to understand, but mentioned that it was not clear when action was necessary to allow access. Two of the participants ($n = 2$) found it difficult to understand. Of these, one participant ($n = 1$) said he had concerns about what was shared as a result of granting a permission—specifically access to the SD card. He was not sure if his contacts or other information was stored on the SD card and thus shared as a result of granting the permission.

> *P6: "It was easy to understand what they were requesting access to... access to the SD card... but you know, what does that do? What all is held on the SD card, because I don't know where the information is held [like my] contacts. [...] I just really don't know what they are getting access to."*

The other participant ($n = 1$) said he was not able to understand how to do the task without additional help from the investigator.

Most participants ($n = 11$) did not think contextual access control was complicated. Some of them ($n = 5$) commented it was not complicated because it required little additional effort to use the prototype applications. A few participants ($n = 3$) said it was clear how to use contextual access control to grant permissions. Several participants mentioned that although the access control was simple to use, they did not like the repetition of interaction ($n = 2$) nor the extra time ($n = 2$) needed for managing access control.

Most participants ($n = 12$) felt like they had more control over their data using contextual access control. Some of these participants ($n = 5$) also liked the ability make a decision about whether or not they were willing to share during runtime. Similarly, some of these participants ($n = 5$) liked the increased awareness that resulted from the contextual access control.

> *P3:"Everything was asking for... like I had to save it. So I was aware of everything that was going on and what was being saved, it let me know... stuff popped up to let me know what was going on."*

One participant ($n = 1$) felt neutral about being in more or less control and commented how he wanted to see install time permissions but also liked being aware of when the application was requesting access using contextual runtime dialogs. Interestingly, one participant ($n = 1$) said he felt as if he was in less control when using contextual access control because he felt rushed to complete tasks.

P13:"You have to allow so much information and your really not... I know I wasn't paying attention to what I was supposed to be allowing. I was just rushing to get through it so I could complete the task. You just have to let go of some security in order to have access to the program."

Some participants ($n = 5$) found the runtime contextual access control to be distracting. The primary concern expressed ($n = 4$) was being distracted from completing the study task. The prototype applications using runtime contextual access control produced a dialog message automatically at any moment when a function was used that needed permission to access the phone. One participant explained how he felt distracted by the dialogs:

P9:"You are trying to do something, and you might be in the middle of doing something... and it just disrupts the flow of completing it."

Half of the participants ($n = 7$) disliked the extra effort and repetition involved in contextual access control. Some participants ($n = 5$) also found runtime contextual access control annoying.

P6:"It is good they are asking for permission but to allow every single access to all the different components is pretty annoying."

One participant ($n = 1$) also said that contextual access control was not user friendly. Other participants said they were not annoyed because they only had to grant access one time ($n = 2$), and they liked the additional awareness that resulted from contextual access control ($n = 2$). For example one participant explained:

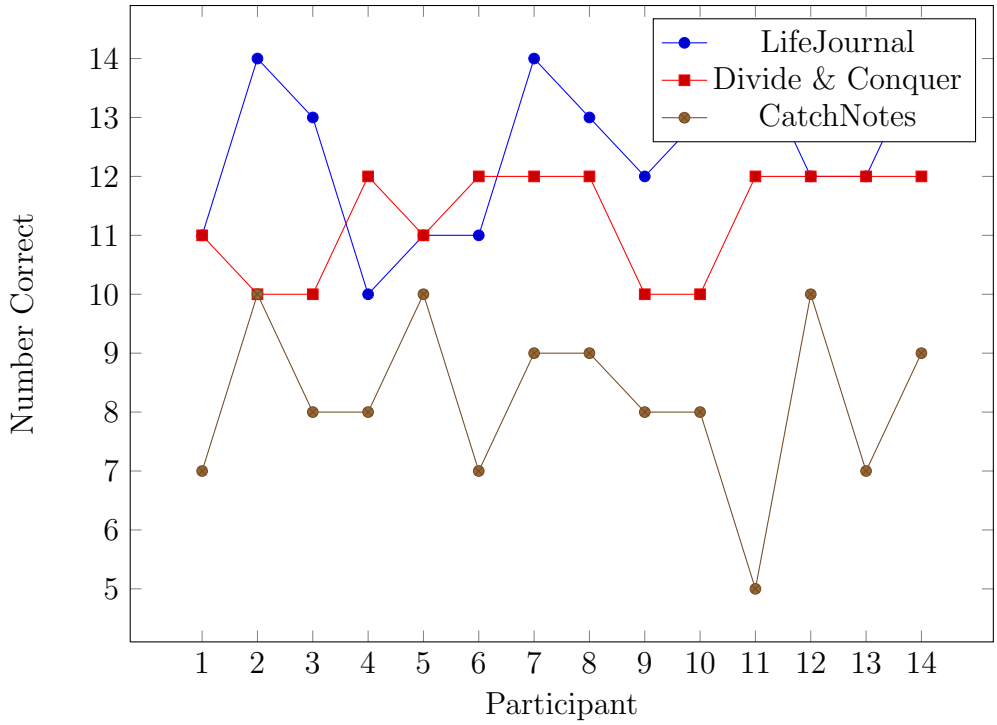

Figure 7: Memorability scores for two runtime contextual configuration applications vs an install time application Catch Notes.

> *P3:"It helps you understand what was going on and what your doing with the phone. I think it helps prevent confusion."*

Many participants in the study reported liking the increased awareness ($n = 7$) and enjoyed the increased level of control ($n = 8$). Yet, half of all participants ($n = 8$) complained of the effort, repetitive nature, and annoyance that they felt towards contextual access control. Ultimately, most participants ($n = 9$) thought they would want access control like this on more applications in the future. However, some participants ($n = 5$) said they would not want to see runtime contextual access control in the future citing concerns with the repetition of dialogs ($n = 2$), the annoying nature of it ($n = 2$), and that they had to deal with popups ($n = 2$).

I conducted a test to measure participants' memorability of what permissions were

requested by both types of access control techniques (runtime contextual and install time). The tests contained 16 possible permissions available to any of the applications. The same 16 items were asked for each of the three applications, however, each of the three applications asked for permission to different items. For each application, participants indicated whether that application did or did not ask for the permissions on the quiz. The total score was calculated as the total number of correct responses for each participant. A participant's answer was correct if he indicated an item was asked for and it was or an item was not asked for and it was not. Any other answer was marked as incorrect.

Participants scores per application are shown in Figure 7. Divide and Conquer and LifeJournal both used runtime contextual access control configuration, however, each application differed in its set of requested permissions. Catch Notes and LifeJournal were nearly identical in functionality but differed in access control techniques. Figure 8 shows histograms of scores for each application. The distribution of scores for both Divide and Conquer as well as LifeJournal are negatively skewed. The distribution of scores for Catch Notes, the install time application, is approximately normally distributed. In addition, Catch Notes scores appear to be similar to the binomial distribution. For this reason I tested each of the scores using the binomial test. I found that both Divide and Conquer and LifeJournal to be statistically significant with $p < .001$. However, Catch Notes was not statistically significant with $p = .791$. This means that participants observed scores were not significantly different than what could have been obtained by simply tossing a coin to determine the answer for each permission.

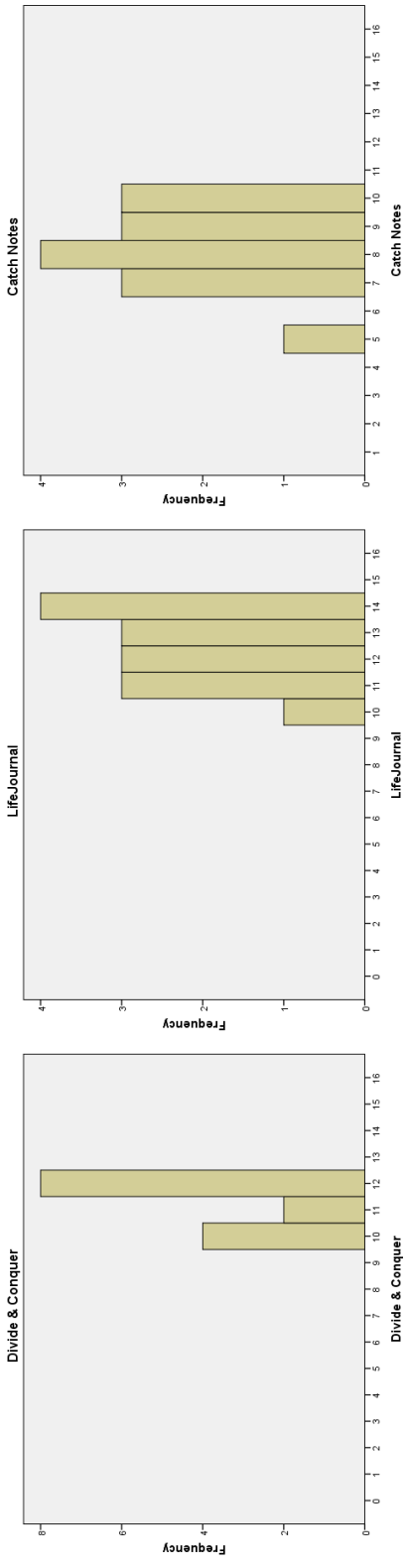

Figure 8: Histograms of memorability scores

Since the scores for all three applications were not normally distributed, I chose to use a Friedman test to determine if there were differences between the scores obtained for each of the three applications. After finding a significant difference with the Friedman test, I used Wilcoxon signed ranks tests to do all pairwise comparisons of applications. I calculated 6 tests (3 pairwise, 3 binomial), therefore, I adjusted my significance level to ($p < .008\overline{3}$) using a Bonferroni adjustment.

Using the Friedman test I found that the memorability scores were statistically significant with $\chi^2(2) = 23.098, p < .001$. Specifically, I found that participants scored higher $Z = -3.195, p < .001$ on Divide and Conquer ($M = 12$) than they did on Catch Notes ($M = 8$). Additionally, participants scored higher $Z = -3.330, p < .001$ on LifeJournal ($M = 12.5$) than they did on Catch Notes ($M = 8$). Effect sizes for Divide and Conquer and LifeJournal were both large, ($r = 0.85, r = 0.89$) respectively. There was no significant difference $Z = -2.172, p = .030$ between Divide and Conquer and Life Journal, both runtime contextual configuration applications. Effect sizes are reported as r as advocated by Rosenthal for analysis in social research [51].

5.1.4 Discussion

This study examined the differences between install time access control and runtime contextual access control. The results indicate that there may be tradeoffs between control, awareness, memorability, and usability that need to be addressed. I now highlight some of these findings.

- Despite liking awareness and additional control there are major issues experienced by half of the participants.

Participants in this study liked that they had control over access to information on the phone. Participants also gave positive comments on the awareness of access they gained using contextual access control. However, runtime contextual access control requires more interaction with the user while they are engaged with the primary task. As a result, half of the participants in this study disliked the extra effort and annoyances of runtime contextual access control. Participants did not seem to mind configuring one or two permissions, but felt it cumbersome to configure access many times. This problem of repetitive prompts surfaced throughout the interview discussion on various aspects of the user experience. It seemed to not be limited to one aspect of their experience. I also note this same issue in the next study presented in Section 5.2. Since this is a recurring problem, I more thoroughly explore the issue in Chapter 6.

Runtime contextual access control in these prototype Android applications could invoke permission configuration dialogs when using functionality that required permissions for the first time. After granting the permission, the application continued to operate with the expected functionality. It appears that the access control dialog may have been perceived as an unexpected response from the system. For example, when clicking "Tag a contact," the expectation is to see a representation of the contact on the phone. Instead, the application displayed a dialog asking for the contacts permission. Participants often referred to this dialog as a popup. Since the popup came as an unexpected response by the system, it seemed to distract many of the participants. In some cases, this distraction was described as annoying and future research on runtime contextual access control should explore how to capture

the increased awareness while decreasing or eliminating annoyances.

- It is not clear how well participants comprehended the permissions that applications were asking them to grant.

Nearly all the participants believed it was very easy to understand what permissions were requested by the applications using runtime contextual access control. Phone permissions are not very technical and are generally understood by most users, so it is not surprising participants' reported understanding of the access control. For example, several applications asked participants for permission to their contacts, the internet, or the phones vibrator. These types of permissions are not technical in nature and understood by most phone users. However, it is not clear users understand exactly where their data resides on a phone and what access is granted with some permissions. One of the participants pointed this issue out by asking what the granting of a certain permission really means. In particular, they questioned what it meant to grant the application permission to access their SD card. This participant introspectively thought about their contacts and photographs being stored on the SD card and therefore did not consider contacts as a separate permission. When using fine grained access control, presented in Section 4.1, understanding the scope of permission sharing was less of a problem. In that study, sample data was used to help establish context and convey details of the requested information for participants. Two participants in this study were confused with what would be actually shared by granting a permission. Despite this confusion, it illustrates that participants exhibited increased awareness by actively considering application permissions and the decision

to grant access to the application. Future research should consider combining these findings to see if presenting actual user data can further help understand the sharing scope and access control.

- Runtime contextual access control increased memorability of configured permissions and possibly awareness.

The results of this study support that runtime contextual access control increases memorability and awareness. Although participants were not directly asked about awareness in the interview, half indicated they were more aware that they were sharing data with the applications using runtime contextual access control. Quantitative results suggest improved memorability between the two prototype applications and the standard install time application. Participants' correct scores for remembering requested permissions increased by 25% between applications using the different access control techniques. Using prototype applications could have confounded these results with a novelty effect, however, many of the participants ($n = 8$) in this study reported never using an Android device. Thus, the novelty of Android's install time access control system should have been similar to the novelty of the runtime contextual access control for these participants. Additionally, I informed participants that the purpose of the study was to examine the differences between the two access control systems. This would have likely increased emphasis on attention to access control for both techniques.

Memorability and awareness are not equal. However, evidence suggesting increased memorability likely reflects some increased awareness of sharing. Increased awareness

can improve the capacity to make more informed decisions about what and when users share to applications and possible third parties. Unfortunately the positive results indicating an increase in accuracy of remembering requested permissions may still not be ideal. Recollection results for install time access control were poor and could also be represented by random guesses, therefore a significant increase in recollection may be good, but not represent enough awareness to truly inform users about information sharing.

Recollection of permissions at runtime using contextual access control was better, but the improved scores can still be considered poor. Participants were only able to recall permissions with 75% accuracy. Forgetting 25% of the granted access can possibly lead to significant unintended access because a single permission can indicate access to a large amount of information. The potential damage from such unintended access can still be widespread and serious. Improving and understanding the link between awareness and memorability is an effort that should be explored in future research. Further improvements will only help increase awareness and result in users making better sharing decisions.

5.1.5 Conclusion

This section compared runtime contextual access control to install time access control. Contextual access control consistently resulted in higher memorability of which permissions were requested by a phone application. Runtime contextual access control improved recall by 25% over the install time access control. Despite the positive change in the overall memorability, the results were still disappointing. Participants'

average score for install time memorability appeared to be no better than simply tossing a coin. Runtime contextual access control recall scores were on average 75% correct.

While 75% may appear to be a high number, the leftover 25% of sharing that was either unrecallable or incorrect is a concern and should not be dismissed. Future work should seek to further increase this memorability to provide users with better awareness of their data sharing. Additional research may also provide insights into how important memorability is to overall awareness with access control. In particular, discovering quantitative methods to judge awareness would be a pertinent contribution. Having such a quantitative tool may help to more completely compare different versions of access control configuration systems and eliminate variations that might result in unintentional sharing.

The prototype in Section 4.1.2.3 presented a concrete representation of what was being shared as a result of granting permissions. In contrast, the prototype in this section only provided the permission name and the ability to make the decision. As a result, two of the participants were unsure of the data sharing that they were engaging in. Future iterations of either type of access control should contain concrete representations of data being requested. Omission of this improved context can cause user decisions from incomplete information and therefore may expose users to a significant amount of risk from unintended sharing.

Finally, the prototype in this section presented an access control dialog whenever the application's function needed a permission not previously granted. As a result, participants encountered what they frequently called popups as they interacted with

the application. The repetitive nature of these popups came up throughout the study and manifested itself as an annoyance to the participants. This problem should be addressed in order for runtime contextual access control to be acceptable as a deployed technique. In the next section, Section 5.2, I attempt to address reducing this annoyance by changing the way dialogs are initiated within the application.

5.1.6 Acknowledgements

I would like to thank Ashely Anderson, a summer research experience for undergraduates student, who helped in the development of the two prototypes in this section.

5.2 Social Network Contextual Configuration

In Section 5.1, I explored contextual configuration on mobile devices with the Android operating system. In this section, I consider contextual configuration on a social network site, Facebook. Configuration of permissions on Facebook is slightly different than the configuration of permissions on mobile devices. On social network sites configuring a permission almost always results in data being shared with an actual third party. This is unlike mobile devices where execution of consumed data can take place directly on the device.

In the previous section, participants were aware of the nature of the study since they were comparing both methods of configuration. In this section, I study just runtime contextual access control on its own. Studying runtime contextual access control alone allowed me to examine decision making and the effects of its implementation. I also studied access control more subtly in this section by designing applications to

facilitate user interaction. Participants in this study were unaware that the access control was being studied.

5.2.1 Prototype Scrapbooking Apps

Originally, I developed an interactive game on Facebook that used contextual access control. Much like the mobile application Divide and Conquer discussed in Section 5.1, the icons were bound to features like high score profiles. In pilot testing, I determined that the game felt too much like creating a profile, similar to AudienceView [37]. For this reason I decided to create three scrapbooking applications that would consume user data and produce digital visualizations of their photos. Creating a scrapbook requires multiple steps such as looking through albums, selecting photographs, looking at friends albums, annotating images, and so on. It is therefore more interactive and less like unlocking elements of a single page. Each application had a different look and feel and requested its own set of permissions from the permission manager, later described. Permissions in these prototypes were still bound to the part of the interface most relevant to their context of use. Participants using the prototypes were led to believe these permission requests were from Facebook. All three applications are shown in Figure 9.

The first application AWScrapbook requested a total three permissions : *Albums*, *Basic Info*, and *Photographs*. The next, Photofire, requested four: *Albums*, *Basic Info*, *Friends*, and *Photographs*. Finally, Tiler, requested six: *Albums*, *Basic Info*, *Contact Info*, *Hometown*, *Location*, and *Photographs*. AWScrapbook was meant to be minimalistic in nature. The visualization it generated was a simple smattering of

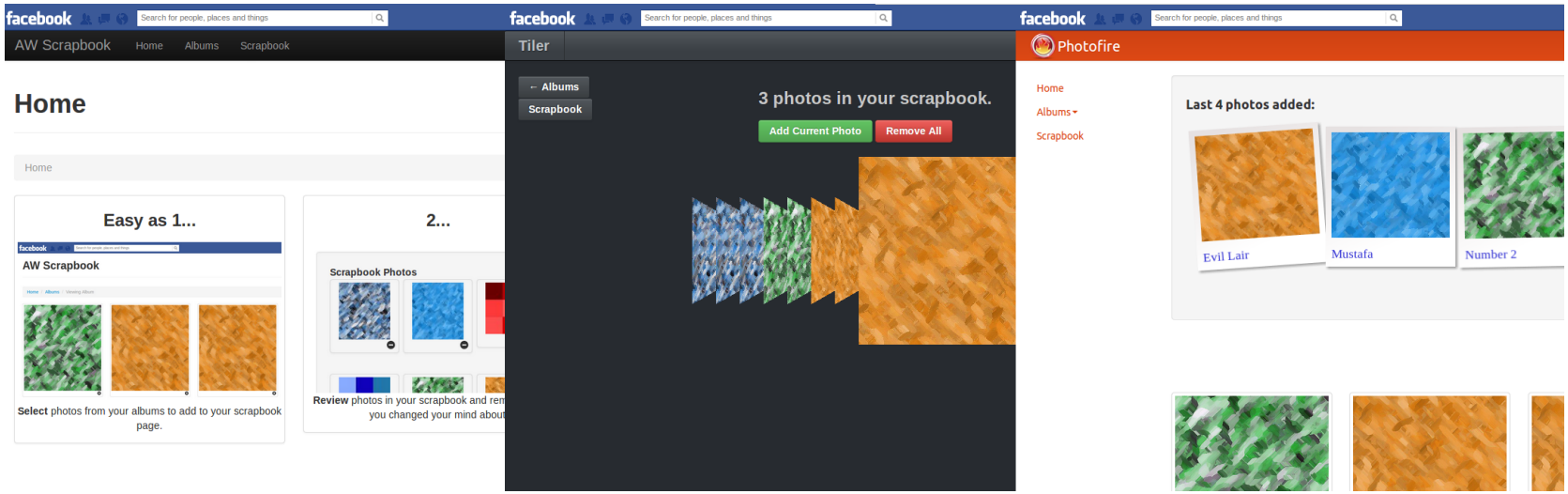

Figure 9: Three scrapbooking applications developed and used to assess runtime contextual access control. From left to right: AWScrapbook, Tiler, and Photofire.

the chosen pictures on an HTML5 canvas with the images slightly rotated randomly. All the permissions it asked for were meant to seem reasonable within the context of creating a photo scrapbook. Photofire was much more polished and included an advanced visualization that looked like a Polaroid photograph. The photographs in this visualization could also be annotated with custom text. Again, the permissions it asked for were meant to seem reasonable related to the scrapbook context. Tiler was the least polished using a carousel style selection. It offered little in the way of visual feedback and requested the most permissions from participants. While each of the applications differed in terms of functionality, visual aesthetics, and permissions requested, they all used the same interface for configuring permissions.

5.2.2 Configuration Interface

While interacting with the scrapbooking applications, participants granted permissions in order to utilize functionality of the application. These permissions were bound to the area of the interface most relevant to their context of use. Therefore, placement of permissions was considered in the design of applications. For the scrapbooking applications I noted that web applications rely heavily on div tags to encapsulate elements in containers, see Figure 10.

I exploited this use of div tags to provide binding locations for the configuration icons used in this study. For example, when displaying a list of participant albums there would be a div for the set of all albums. Inside that div would be another representing just a single row of albums. Similarly, inside that would be another for an individual album. If this section of the interface used the permissions *Albums*

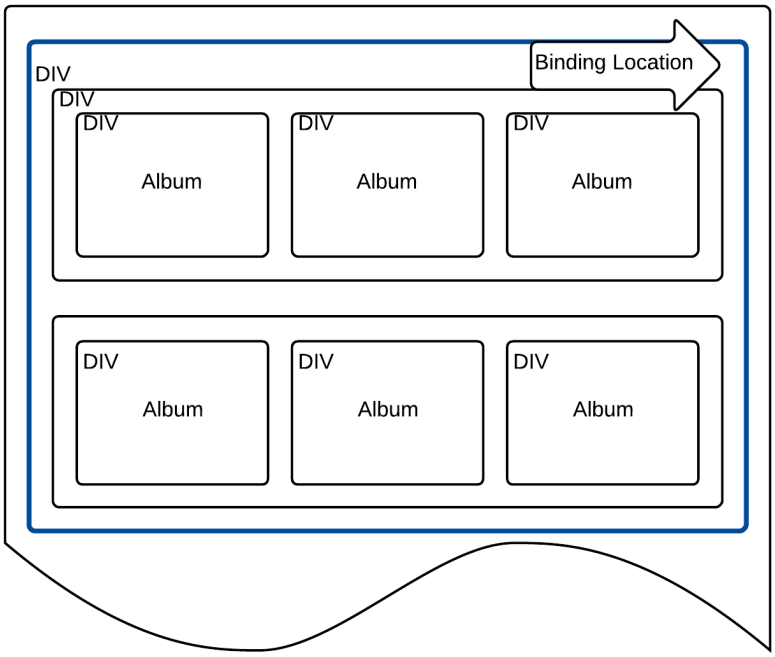

Figure 10: Nested divs and a binding location for permissions

and *Basic Info* then a natural binding location is the topmost div shown in bold in Figure 10. I used this div and absolutely right aligned a security icon like the one depicted Figure 11. Other domains not involving web applications may have different semantics for element placement. The most appropriate placement of configuration icons in this and other domains could be further investigated with additional research beyond the scope of this thesis.

As the participant interacted with the application, they begin to utilize functionality. If a participant tried to use functionality requiring a permission that had not yet been granted, two things would happen—both are depicted in Figure 11. First, the icon for the bound div will begin shaking from side to side for three seconds. Second, a growl sound is generated letting the user know which permissions are required to move on.

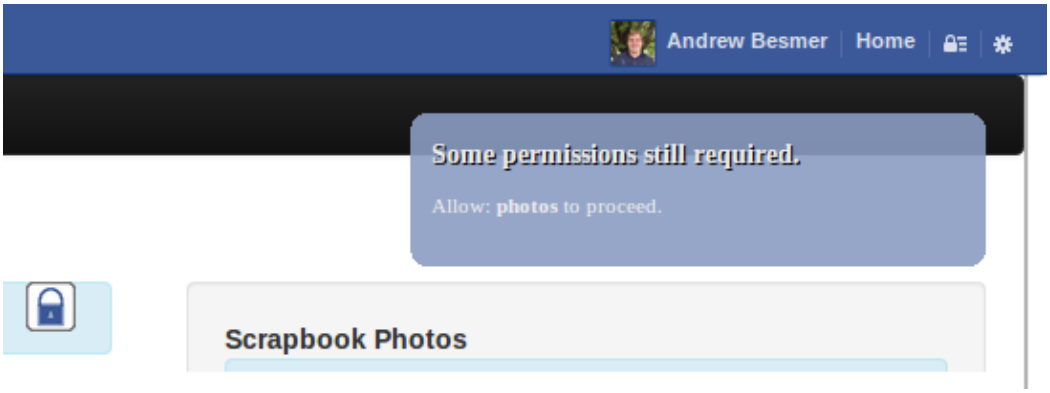

Figure 11: Growl like notification and configuration icon. The configuration icon shown is in motion.

In contrast to the Android implementation in Section 5.1, I do not automatically display a modal configuration dialog. During pilot testing, I determined this to feel less intrusive and hoped to reduce the frustration that accompanies modal popups. If the user hovers over the icon a short animation shows the user the portion of the interface that is bound to the configuration icon. If the user clicks on the icon, the configuration dialog is displayed by the permission manager. An example dialog is shown in Figure 12.

The dialog was designed to be simple to use and easy to understand. Each permission is displayed in its own container. That container contains several items of interest. First is an icon representing the permission that is being requested by the application. Icons lend themselves to being easily scannable. Next, a text description of the permission accompanies the icon, for example *Photographs*. Finally, a representation using the users actual user data is provided. The color of text emphasizes actual user data. In the case of the *Photographs* permission, a random photo is chosen and the string ", etc." is appended to it indicating that there is more in the set.

Figure 12: Runtime contextual configuration dialog with the *Basic Info* permission being dragged to allowing container.

By default, no permissions are granted and are located in the *You are not allowing:* container shown in Figure 12. As a user hovers over each permission, the background color changes to a slightly darker blue which is demonstrated in Figure 12 with the *Basic Info* permission. Additionally, the cursor changes from a pointer to a hand indicating that the permission can be clicked and dragged. Once the user starts dragging the permission, the target container background changes color to a slightly darker gray as shown in the *You are allowing:* container. This signals the user about which container the permission will be located if they stop dragging and release the permission. The notion of dragging from one container to another is meant to emphasize that the user is sharing their data to the application.

Once the permission is released, it is inserted into the target container as the last permission in that container. All the permissions that were below it in the source container move up to fill the space that was previously occupied by the permission.

Once the user is satisfied with the configured policy, they simply click "Okay" and the choices they made will be sent to the permission manager. The permission manager then calls a callback function and provides updates to the interface as necessary. This interaction is further explained in the next section.

5.2.3 Prototype Permission Manager

To facilitate the testing of runtime contextual access control and the configuration interface, I developed a reusable prototype permission manager. I used this permission manager in all three of the prototype scrapbooking applications previously described. Normally, permissions are managed and handled directly by the Facebook API. Since the applications I developed should not allow users to interact with Facebook's permission system, I created a permission manager that sits on top of Facebook's as shown in Figure 13. The permission manager requests the entire set of permissions P where $P \supseteq \{P_{App_1}, P_{App_2}, P_{App_3}, ...\}$. When each application App_j needs a single permission p it now interacts with the permission manager. Since the application will not interact with Facebook's permission system, I can create arbitrary p that does not exist in Facebook's API. I later use this to create finer grained permissions for my prototypes to use. However, since the permission manager is limited by the set of attributes S exposed by the social network (Section 4.1.1), the permission manager can offer no more functionality than the social network API itself provides.

The permission manager provides both client and server side functions for an application to check if p is granted. In both cases, the permission manager will respond indicating whether or not $p \in A_{App_{j,i}}$ where $A_{App_{j,i}}$ is the same application policy as

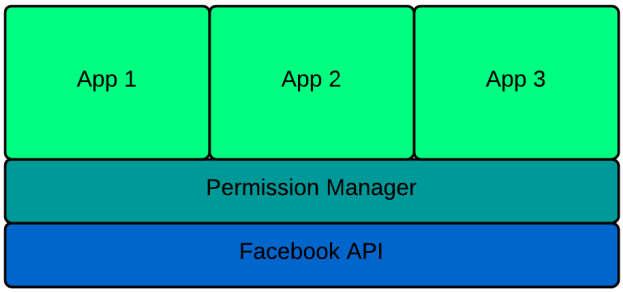

Figure 13: Example of permission manager that was placed on top of Facebooks.

described in Section 4.1.2.1 for a user i. The client or server can then make a decision about how best to proceed with application execution. Additionally, the permission manager also provides helper functions for the client side implementations to handle interface binding and visualizations depending on whether a given p is granted or not.

As the application loads on the client side, helper functions check to see if the user has granted the permissions the application is requesting for that section of the interface. For example:

```
var permRequest = new Permissions();
permRequest.albums = true;
permRequest.albumsRequired = true;
permRequest.basicinfo = true;

$.fn.checkPerms(permRequest, function(result){
        //Get and display albums or default content
}
```

In the above example, I require *Albums* because I want to display them, but I optionally request *Basic Info* so that I can provide a header with the user's name. The function checkPerms takes the requested permissions and checks them. It then calls back the anonymous inner function letting the application know the result. Develop-

ers do not need to concern themselves with alerting the user to missing permissions since the permission manager will do this for them, as described in Section 5.2.2. The only concern should be with the content for the bound div.

Finally, the permission manager and all three of the scrapbooking applications were instrumented to log all the actions taken by the participant. Each time they added or removed a photo, navigated to a page, granted or revoked permissions, that action was logged. Actual photographs were not logged, just the action of adding one. Logs were stored in a password protected MySQL database.

5.2.4 Study Methodology

In the Spring of 2013, I recruited 18 participants from Charlotte, NC and from the University of North Carolina at Charlotte (UNC Charotte) using flyers, tables, and active solicitation to participate in an hour long study. In exchange for participation, participants received a $15 Amazon Gift Card. Inclusion criteria for the study included the participant being 18 years of age or older, having an active Facebook account, and having photographs on Facebook. Participants scheduled an appointment to come to the Usability Lab at UNC Charlotte. Participants were not informed ahead of time about the security, privacy, or access control aspects of the study. I indicated to participants that they would be taking part in a student project to write a report on how to build better scrapbooking software. However, they were actually taking part in a deception based study on the runtime contextual access control.

Once participants arrived at the Usability Lab, I reintroduced myself to them and explained the scope of the study. Participants were told that there would be audio

and video recording and that the session would last about an hour. I then gave the participants an informed consent document and asked them to review it and provide me with any questions they might have. After the participants signed the informed consent, I gave them a brief explanation on how to use the study computer. The computer consisted of two screens, a primary screen where participants would perform the tasks assigned and a secondary screen off to the right containing the survey. I instructed the participants that after finishing each set of tasks, they would be asked to fill out a small survey on the secondary computer screen.

After participants indicated that they understood how to move between the two screens, I instructed them to fill out a brief demographic survey. The survey asked standard demographic questions and included no privacy or security related questions. Once this survey was completed, I handed participants a task sheet and asked them to read it over while I started the recording software and backup audio recorders. I started the recording but I also secretly installed the study software to the participant's Facebook account. This deception was necessary to avoid participants interacting with the Facebook API that uses an install time access control system. By installing the software without their knowledge, they would be interacting completely with the prototype runtime permission manager I developed. Again, the permission manager was meant to look as if it was from Facebook and not part of the disguised study.

The task sheet that participants read first informed them that I in no way endorsed or reviewed the three applications that they were about to interact with. It explained that the three applications were three random scrapbooking applications I

selected from Facebook. It then provided instructions for them to carry out normal scrapbooking tasks for each application. They were asked to add at least 5 photos for each application to the scrapbook, view the visualization, and add custom content to some of the photographs.

Once the participants finished reviewing the task sheet I let them know which application to begin with. Since there were three applications, each differing in functionality and permissions, the order participants used the applications was counterbalanced. I reminded them that after completing the tasks for that application they should fill out the survey on the right.

Once participants started using the applications they would encounter runtime contextual access control for the first time. If confused participants requested help from me I initially avoided helping them. I would simply ask them to show me what they had done. By having them show me what they did, participants would get a second chance of noticing the permission icons animation. If the participant remained confused and still required help, I would mention there is a lock present on the screen. Participants were left to use the application themselves while I sat at a neighbouring desk working on a non study related task. If participants asked if they should allow a permission I would respond by saying I was sorry but did not know anything about the applications.

The survey that accompanied each application was designed to measure several general usability issues: how hard or easy to use the application was, how frustrating or pleasing it was, how much effort it required, if there was a reason they would not want to use it, or if there was anything about it they would change. These questions

were all directed towards the scrapbooking application not the access control. I later separately ask about some of these items in a post interview. Finally, I asked participants to describe any problems they encountered completing the tasks. Each question included a small box asking participants to explain why they felt that way. I specifically instructed participants to be brief in their answer on the survey because I would be fully interviewing them after they completed all the tasks.

Once participants completed using all three applications I read a prepared deception debriefing statement. Participants were informed that the study had actually been about the decisions they had made and that I actually was studying the access control. Participants were then offered the chance to completely withdraw from the study, have all their data deleted, and also keep the $15 Amazon Gift Card for their participation. They were asked if they were willing to stay in the study and answer additional questions about the prompts they received.

Once the participants, confirmed they would remain in the study and answer additional questions, I gave them a short semi-structured interview. In the interview, I asked about the participant's understanding of the access control prompts, problems they encountered, and who they thought they were sharing data with. I also examined their overall experience using the access control and additionally assessed several qualities including: how hard or easy using it was, how complicated it was, how much control they had over their data, any distraction caused by the access control, and desire to have more applications use it. During the interview, the video recording was also played back for them and paused to observe as many configuration prompts as possible given the time remaining for the session. Each time the video was paused,

I asked them to discuss the reasoning for the access control decision they made at that moment. After the interview, I asked participants to complete the Westin privacy segmentation index [34] questions. I used their answers to identify participants as privacy fundamentalists, pragmatists, or privacy unconcerned. Finally, I thanked participants and provided them with the study inducement.

The audio files were transcribed for analysis and log files collected. The transcribed interviews from the recorded sessions were coded using Atlas.ti. Two researchers performed open coding on the transcribed data. We developed concepts based on the agreed coding to examine common responses and better understand behavior and then resolved any coding differences, resulting in inter-coder reliability of 84% agreement between two independent coders.

5.2.5 Results

A total of 18 participants participated in the study. One participant withdrew from the study collecting their gift card after the deception debriefing. I also encountered multiple problems with another participant during the study. That participant answered her phone and communicated via text messages during the interview process and did not focus on the interview questions. I asked the questions multiple times in order to solicit a response because the participant was distracted. I therefore discarded this participant from my analysis.

I analyzed the results for the remaining ($n = 16$) participants. Participants reported ages ranging from 18-43 with ($\bar{x} = 27.44$, $s = 9.19$) and ($M = 24$). Participants were divided fairly evenly between male ($n = 7$) and female ($n = 9$). Partici-

pants reported having a wide variety of ethnicities including Asian ($n = 4$), African American ($n = 5$), Hispanic ($n = 3$) and Caucasian ($n = 4$). Participants educational backgrounds varied with ($n = 7$) indicating they had at least some college, ($n = 7$) indicating they had a master's degree and ($n = 1$) indicating they obtained a Ph.D. Despite all having advanced education, very few ($n = 2$) strongly agreed that others would come to them for computer help, most ($n = 9$) were either neutral or did not agree others would come to them for help. Using Westin Privacy Segmentation Index, participants were mostly pragmatists ($n = 8$), followed by fundamentalists ($n = 7$), and then 1 unconcerned. The sample including so many university students is a limitation of this study and future studies may wish to examine a different sample.

Most participants ($n = 15$) reported having only one Facebook account, a single participant reported having two. All ($n = 16$) reported having been a member of Facebook for over 2 years, many ($n = 12$) over 4 years, and some ($n = 4$) over 6. In cases where participants were unsure we checked using their timeline for when they joined the site. Most participants ($n = 13$) used Facebook several times a day, although some ($n = 3$) used it once a day. So the sample was made up of fairly active Facebook users who were familiar with the site. Nearly all participants ($n = 13$) reported being deceived. Some participants ($n = 2$) said they suspected that the study was not just about scrapbooks and one participant ($n = 1$) said they knew that something else was being tested.

Participants added 339 photos to scrapbooks across all three applications. A breakdown per application is shown in Table 7. I used a Friedman test to determine if there were differences between the number of photos added per application and performed

Table 7: Photos added and dialogues requested per application.

	AWScrapbook	**Photofire**	**Tiler**
Photos Added	112	147	80
Dialogs Requested	56	57	76

pairwise comparisons with Bonferroni adjustment ($p < .0166$) because of multiple comparisons. The number of photographs added per application was statistically significant with $\chi^2(2) = 23.143, p < .001$. Specifically, participants added more photos to Photofire ($M = 8$) than AWscrapbook ($M = 5$), $Z = -2.947, p = .003$. Additionally, participants added more photos to Photofire than to Tiler ($M = 5$), $Z = -3.494, p < .001$. This result is not at all unexpected since the Photofire application included the ability to add photos from your friends albums which AWScrapbook and Tiler did not provide.

Participants in total were shown 189 dialogs, see Table 7, and requested ($M = 12$) dialogs each. The minimum number of dialogs required to complete the tasks for all applications was 7—this assumes participants added all required permissions the first time the dialog was initiated. Despite the appearance that participants initiated more dialogs for the Tiler application, I found no statistically significant difference $\chi^2(2) = 5.193, p = .075$ between the applications using a Friedman test. Although this was not statistically significant, it is approaching significance. Additionally, the effect size of the differences between Tiler/AWScrapbook ($r = .32$) and Tiler/Photofire ($r = .48$) was moderate. Given a larger sample it is very likely this difference would be significantly different and is therefore worth examining why Tiler had so many more dialogs.

Table 8: Final policies configured per application

	AWScrapbook	**Photofire**	**Tiler**
Albums	16	16	16
Photographs	16	15	15
Basic Info	6	6	4
Friends		13	
Contact Info			14
Hometown			3
Location			4

The Tiler application was designed to look crude and unrefined as well as request the most permission from the participants. In order to view photographs in an album, participants would have to grant the *Contact Info* permission. Many participants ($n = 13$) tried not to grant *Contact Info* to Tiler. Most ($n = 8$) only tried once to not allow their contact information to be shared with the application. In other words, they made a decision not to share it and then after seeing it was required went back and allowed it. Some ($n = 3$) tried a second time not to allow it, and one participant ($n = 1$) tried 6 separate times not to allow the permission before giving up and granting it. Finally, a participant ($n = 1$) tried once not to grant this permission and when the application did not respond by showing their photographs, they initiated the dialog twice in rapid succession and closed the dialog before it loaded. This triggered a bug in the application that allowed the photographs to be seen without the *Contact Info* permission, however, this participant had already granted access to *Photographs*.

Users configured a variety of policies during the course of the study session. Table 8 shows the final state of the policies participants had configured. The first thing to

note when looking at the table is that there are high rates of sharing for the *Albums* and *Photographs* permission. Given the nature of the applications participants are using this is expected. However, note the permission *Photographs* was not granted by every participant in the study. The reason for this is one participant revoked that permission for the Photofire application. When asked about it the participant explained that they were experimenting with the access control after completing the task to see what effect revoking it would have.

The permission *Contact Info* was also granted by nearly all ($n = 14$) participants. As previously discussed, this permission was required in order to view the photographs in an album and most participants initially attempted not to allow it, however, a couple of the participants ($n = 2$) did not allow permission. As a result, one participant ($n = 1$) failed to complete the task. After granting *Albums* she quit attempting to use the app and filled out the corresponding survey without asking me for help. Because study design required minimal interaction from the investigator, I was not aware of this until after the participant had been debriefed. The other participant ($n = 1$) I previously noted discovered a bug allowing her to see the photographs without actually granting *Contact Info*. Her initial decision was not to grant permission.

5.2.5.1 Decision Making With Contextual Access Control

Overall, participants granted permissions 148 times and revoked them 8 times. Participants did not always grant or revoke permissions for a single reason, therefore, it was possible that a participant had several reasons why they would grant or revoke a permission. Additionally, because of time limitations it was not possible to discuss

each and every decision. As a result there may be reasons participants granted or revoked permissions that went under-reported or even unreported. However, the majority of decisions were discussed with participants in the post interview.

When asking for permissions with contextual access control, I bound the permission requests closest to the context of use. Consequently, most participants ($n = 14$) cited granting a permission because it was necessary to unlock the features they needed/wanted to use. Most times the functionality they envisioned was real but other times it was imagined. For example, granting a hometown so that photos taken in the participants hometown would be easier to find. One participant explained why they granted *Basic Info* when picking out photographs:

> *P10:"[I allowed it] so that when it could publish the photographs, it would publish with my name."*

Similarly, most participants ($n = 11$) reported denying a permission because they felt it was unneeded for the current task. Most participants ($n = 11$) also reported denying permission because they were confused or unsure why it was needed for the functionality they were attempting to use for the task. Another participant discussed the same permission:

> *P16:"I really couldn't see why I would need to do that if I'm just making a picture scrapbook."*

Most participants ($n = 9$) reported granting a permission because they felt forced to allow it in order to continue the task. In fact, many participants ($n = 6$) even said they would grant almost anything in order to get the application to work.

Some participants ($n = 5$) engaged—at least partially—in managing privacy at the disclosure boundary. In other words, they manage what they share with social network by self censoring in order to ease the burden of making a privacy/security decision later. For example:

> *P14:"Sure, because the contact info I have posted on Facebook... I'm very careful with what info goes onto Facebook, so even if an advertiser or a identify thief got a hold of it, it wouldn't be enough for them to be able to ruin my life with. Anything on Facebook, I'm okay with sharing because I'm not putting anything on there that is going to be able to hurt me."*

As I note later, this participant became quickly desensitized to the dialog and granted every permission requested for every application.

Some participants ($n = 4$) explained that they were willing to grant the permission because it was the only permission being asked for in the dialog. Many participants ($n = 7$) also reported basing their decision on whatever they believed to be the minimal set of permissions to accomplish the task. This is different from being necessary to complete the task because in these situations participants were actively considering all the choices to chose the minimum possible policy to accomplish their goal. A few participants ($n = 3$) mentioned granting a permission just because they wanted to see what would happen if they added it.

> *P10:"I wanted to see, was it getting access to the next level by sharing only my location and not giving access to the photos? I wanted to see the response [from] the applications still."*

In essence they were testing to see how the permission they granted was affecting the application.

Other miscellaneous reasons participants reported granting permissions include: the application did not respond fast enough, having the ability to later revoke the permission, not disagreeing with request, the permission could possibly be useful in the future, not considering the data as personal to them, and the permission requested was safer than alternatives. Other miscellaneous reasons for denying a permission included: simply not wanting to share, friends already knew the requested data, the requested permission was unsafe to grant, and the permission being requested was unexpected.

Finally, participants revoked permissions 8 times. For some participants ($n = 2$), this was recorded as part of the decision making process. For example, the participant granted access to *Basic Info* and before clicking "Okay" dragged it back. One participant ($n = 1$) had decided to experiment with the access control as previously mentioned. After he finished the task, he went back and started revoking permissions to see what effect it would have on the application. Finally, a participant ($n = 1$) unintentionally granted a permission and when he discovered what happened, revoked it. This is a serious interface problem that I discuss later in Section 5.2.5.3.

5.2.5.2 Attitudes Towards Contextual Access Control

During the semi-structured interview participants discussed their feelings towards contextual access control. I discussed with participants various dimensions of those feelings including if they thought the access control was complicated and if they

reported being annoyed or distracted by the dialogs. In this section, I discuss how participants felt towards contextual access control and why they described feeling that way.

Many participants ($n = 7$) said their overall experience was positive although a few ($n = 4$) said it was negative. When asked about their overall experience, many participants' ($n = 6$) feelings were related to the applications and not necessarily the access control. Even after the deception was exposed, participants actually commented about the interactions with the prototype applications instead of the contextual access control. For example, one participant that said their overall experience was negative commented:

> *P12:"I think the only reason why I would say somewhat negative [is] because of the fact that it exposed my location and hometown which I found it kind of shocking."*

Similarly, half of the participants ($n = 8$) filling out the survey for the tasks performed after each application referred to the access control in determining their experience with that application. These surveys were taken before they had been debriefed. None of the questions in these surveys mentioned or asked about access control. During other questions unrelated to the applications, many participants ($n = 9$) had trouble separating their thoughts about the access control from thoughts about the applications. This may indicate there is an interdependent relationship between the access control and the application.

Many participants ($n = 8$) reported being disturbed by the applications requesting

permissions before being asked about their sharing decisions. For example, when I asked a participant if they knew the study was not about the scrapbooks, they responded:

> *P6:"No, [...] but I did notice that one of the things required me to add my contact information, that didn't bother me either because I only have an email address up there that I no longer use and then I noticed that it had my location up there but I didn't allow that."*

This indicates an increased awareness of sharing and concern of it when using runtime contextual access control.

Nearly half of the participants ($n = 7$) said they thought the access control was annoying. Of the participants who found it annoying, nearly all ($n = 6$) mentioned the repetitive nature of granting permissions as the reason for feeling that way. Interestingly, some of these participants ($n = 3$) also mentioned the permission being asked for as the reason they were annoyed with the access control:

> *P9:"... and then the fact that it was asking for what it was asking was annoying me. [...] One of the ones asked for access to my email stuff, like my main account information, and then it asked for my location. I didn't see the need for it to know where I was. [...] and then the last one I didn't like was the friend access scrapbook part of it. I didn't particularly care for that, because I don't know how the other person feels; their stuff is being added to something they didn't know."*

This provides further evidence that runtime contextual access control and the appli-

cations themselves have an interdependent relationship.

The remaining participants ($n = 8$) did not find the access control annoying. Participants cited various reasons they did not find it annoying including: it was easy to understand ($n = 1$), it offered more security ($n = 1$), it was part of a process to use the application ($n = 1$), and that they wanted contextual runtime access control ($n = 1$). A couple of the participants ($n = 2$) did not find the access control dialogs annoying but did mention the interactions seemed repetitive.

When asked about how complicated the access control was, most participants ($n = 12$) felt it was not complicated, although some ($n = 2$) found it complicated to use. A few participants ($n = 3$) discussed how their feelings were at least partially tied to the applications, again possibly pointing to an interdependent relationship between access control interaction and general interaction with the application. Some of the participants ($n = 4$) who did not find it complicated discussed some initial confusion they had with how to use the access control. All participants ($n = 16$) said it was easy to understand what information the applications requested with the runtime contextual access control dialogs.

Overall, participants did not find the access control to be distracting ($n = 11$). Participants who did not find it distracting explained: they considered it to be part of the process ($n = 4$), they like the ability to have control ($n = 2$), found the information given informative ($n = 1$), and gave them a secure feeling ($n = 1$). The remaining participants ($n = 5$) found it distracting. Most of those participants ($n = 3$) cited the repetitive nature of the permissions:

> *P9:"[...] and every time I would try to complete a task, I had to go to another permission, it would ask me for another permission."*

Some participants ($n = 2$) also said it was distracting because they were trying to understand why they were being asked for certain permissions. Additionally, a few participants ($n = 2$) suggested that some justification should be provided along with the request for permission.

Most participants ($n = 10$) reported feeling like they had more control over their data and two participants ($n = 2$) felt neutral about it. All participants who said they felt like they were in more control ($n = 10$) cited the ability to chose exactly what to share. Two participants ($n = 2$) also mentioned increased control because they had the ability to better understand the sharing that occurred. Surprisingly, some participants ($n = 4$) felt like they were in less control of their data. Participants ($n = 4$) cited feeling like they were in less control because they were unsure of what was done with their data after sharing. Another participant expressed an insecurity by describing the sharing as a possible hacking risk ($n = 2$):

> *P9:"[...] I think one of them asked about contact, it listed my email and something else behind it. I don't know what it's doing with that data itself. I was concerned that, 'Okay, is this going to hack into my page later on,' type of thing."*

When asked about what would happen if their account was hacked because of the sharing, they indicated that their friends would potentially be spammed from their account. They explained that their friends had experienced similar unpleasant hacking

events.

Most participants ($n = 10$) wanted to see access control like this associated with more future applications. Two participants ($n = 2$) did not care either way and four participants ($n = 4$) would not like it used with other applications. Participants who did not want to see it cited the time to configure policies ($n = 2$) and that it was annoying ($n = 2$). Interestingly, one of the participants who found it annoying also did not want it because of the information the applications had requested. They considered the decisions like granting photographs to a scrapbooking application to trivial to even be asked. Participants who wanted it used for future applications primarily liked the ability to choose what they were sharing ($n = 9$). Some also found it easy it to use ($n = 2$), thought it was intuitive ($n = 1$), and said it left them feeling secure ($n = 1$).

5.2.5.3 Problems With Contextual Access Control

The most common problem participants ($n = 11$) encountered was being initially confused about what was going on and how to initiate interaction with the prototype applications. Sometimes participants ($n = 4$) blamed themselves for the application not working. However, most participants ($n = 9$) were able to overcome this quickly and use the applications to complete the tasks. The remaining participants ($n = 7$) asked me for help when they encountered their first application that required them to interact with the contextual access control dialog. When participants asked for help I tried to provide little or no assistance as previously described. Most of the participants ($n = 4$) who required initial help simply needed to be shown the lock

icon. However, a few of them ($n = 3$) figured it out with me just looking over their shoulder and offering no assistance. After this initial help, participants completed tasks with the remaining applications without any assistance. A few participants ($n = 3$) suggested having additional instructions might avoid some of their initial confusion. As part of this study, no interstitial explaining the access control change was given to participants.

During the semi-structured interview, I asked participants who they thought they were sharing data with as they interacted with the contextual access control dialogs. While it can be harder to identify participants that fully understand the scope of sharing, it is easy to see when a participant does not understand it at all. A few participants ($n = 3$) indicated they did not understand the sharing scope. One participant ($n = 1$) indicated they thought they were sharing with their friends and several different locations. They said because they were asked to drag *Albums*, *Location*, and *Hometown* that they were sharing their albums with their location and hometown. This was an understandable, but incorrect notion of data sharing. This misconception of data sharing led to this participant considering only their friends as an audience when making decisions. Another participant ($n = 1$) thought they were sharing with themselves and their friends. Finally, one participant ($n = 1$) thought they were sharing only with Facebook, the company. They rationalized that because Facebook offered the application, Facebook was asking to share with themselves.

One interface issue led a participant ($n = 1$) to unintentionally grant a permission. When a permission is dragged and dropped, it becomes the last item in the box it was released to. All the other items that were below it move up to occupy the pre-

vious space. This participant in manipulating the *Photographs* permission had that permission moved to the last place in the box and *Basic Info* moved up to occupy its space. When the participant finally decided to grant *Photographs*, she accidently granted *Basic Info* instead of *Photographs* because the permissions had changed positions. When the application did not respond like she expected, she realized what she had done and then revoked access to *Basic Info* and granted *Photographs*.

A few participants ($n = 3$) became desensitized to the prompt and referenced that as a reason for their granting a permission.

> *P14:"Yes, I was like, okay, another one. Dink! [...] I was just on autopilot at this point, dragging."*

This participant received a total of 12 dialogs and used the scrapbooking applications for approximately 16 minutes. The second desensitized participant received 10 dialogs and used the applications for 27 minutes. The third user encountered 8 dialogs and used the applications for 15 minutes. Despite there being three participants who mentioned being desensitized, only one ($n = 1$) actually granted every permission they encountered. This was also the only participant in the study that granted every permission requested. Others mentioned that despite being in a desensitized "autopilot" state they still noticed things they were not comfortable sharing.

> *P13:"Other than the one that said hometown. I don't know why that caught my eye, but I didn't drag it over. [...] It didn't just say IOS photos or albums. It asked hometown. I thought, hmm, why did they want to know that? Then I didn't use that, I mean, drag it over there."*

Finally, some participants ($n = 3$) said the lock icon for the prompt was too small and needed to be made bigger and a few others ($n = 2$) suggested moving it because they did not notice it. These minor issues were not directly studied, so it is not possible to know how often they surfaced. Future studies might help to determine optimal placement of privacy control icons.

5.2.6 Discussion

In this study, I offered participants the ability to configure a smaller set of permissions that are contextually bound to areas of the interface where they will be used. As a result, most participants thought about the functionality that they used and determined what to grant based on that functionality. I observed an increased awareness of data sharing, increased decision making, and a general positive attitude towards contextual access control. However, I also observed several paradoxical results like engaging in decision making while desensitized, inability to distinguish between application and access control experience, and a feeling of less control that seemed to be the result of more awareness. This may be similar to other research which indicates that being more informed of data flows may lead to cognitive overload [54].

5.2.6.1 Desensitization

When all a user has to do is click "Okay" or "I Agree" to move on they frequently become accustomed to the prompt over time and stop carefully reviewing them. Earlier in Section 4.1.4, I examined offering fine grained access control at install time. The default policy in that case was to allow all requested permissions so that unmotivated users were not inconvenienced. In that study, nearly half of the participants

did not take the time to review policies and accepted the default policy. Since the default policy was to allow all requested permissions, I expected users would quickly become desensitized.

In this study however, the default policy was to allow nothing. In fact, participants needed to initiate an access control dialog to begin using the applications. They then would have to drag data they wanted to share from one section of the interface to another. Obviously, by using so many dialogs, I expected desensitization over time. However, I observed three participants who seemed desensitized almost immediately. The rapid desensitization and the additional effort introduced to manage permissions may reduce the efficacy of using runtime contextual access control. This may indicate that desensitization could also be a larger issue beyond these three participants.

Interestingly, two of the participants who reported being desensitized mentioned noticing a particular permission while using contextual access control during runtime. It may be possible that the small amount of time it takes to drag the permission combined with the enhanced context of actual user data helps them to more carefully consider the sharing decision. As previously mentioned, these desensitized participants changed the default and did not grant the application permission. If the sharing decisions for desensitized users are somewhat subconscious, using rich contextual cues such as real user data may make sharing more salient even with desensitization. This phenomenon warrants future research on possibly improving subconscious decision making with contextual access control. There would be some positive research implications if it is possible for desensitized users to bring sharing decisions closer to consciousness in moments where it subjectively matters to them.

5.2.6.2 Distinguishing Applications From Access Control

The dialog emphasized sharing with an application and forced participants to give their information to the application by dragging it. Participants were told they were testing random applications found on Facebook. Despite this emphasis on sharing with an application, some participants still did not understand who they were sharing with. Participants answers' appeared to be directly influenced by the contextual access control interface. Although this may be considered a problem, it may also be a positive result. By focusing on the parties involved in sharing, this problem could be lessened or eliminated. Of the three participants who did not understand with whom they were sharing, each of them had different answers.

One participant believed that because he was dragging location and hometown, he shared the scrapbook with those locations. The actual caption for the box used for granting permissions was labeled: "You are allowing:". It seems reasonable to consider an appropriate interpretation of that label is "You are allowing photographs with Charlotte, NC." presuming that Charlotte NC. was where his friends were located. The second participant thought he shared with himself. Similarly, the third participant thought he shared with Facebook, the company. While a first impression might seem that these are odd participant responses, it makes sense with the design of contextual access control and how applications are displayed on Facebook. Facebook applications enjoy a special privilege of appearing in the Facebook site itself, see Figure 9 and Figure 11. Immediately above the application, you can see the Facebook header which shows the users name and provides them quick access to

privacy settings, additional help, the logout feature and more. These features are all provided by Facebook and interact directly with Facebook. Thus, there is no visual separation between the application and the social network site itself and likely will lead to others being confused with sharing audience when granting permissions to Facebook applications.

When we add contextual access control, we further exacerbated confusion with the intended sharing recipient because the prompts, icons, and notifications all appear to be from Facebook. It becomes even harder to distinguish Facebook from a third party application. This is particularly true of well polished applications like Photofire. When a user encounters applications like scrapbooking applications, it may appear as though it is a new way to manage Facebook photos not yet activated. Once the user grants a permission like *Albums*, the user interface changes and updates displaying a list of their albums. It now seems entirely plausible to the user that he was activating a Facebook feature by moving their albums to this new visualization or even sharing with himself so he could see his own albums. Unfortunately, this could have negative and unintended consequences if more sensitive items are requested since both audiences may be trusted by the user. The user obviously trusts himself with his data and since he has already given it to Facebook he plausibly entrusts data to them as well. It is possible that as users use more applications with contextual access control, they would become accustomed to the correct sharing recipient of their data. However, this does not mitigate the damage that may already occur as a result of an incorrect mental model.

Other participants expressed attitudes that led me to believe they understood the

audience. However, they were no less susceptible to the blurred line between applications and the access control. As I stated previously, when discussing aspects of the access control many participants would comment about the applications. Their experience with those applications led them to feel a certain way towards the access control. Similarly, when discussing their experience with applications, they would also discuss the access control. This intermixing of concepts underscores the symbiotic nature of the applications and the access control. A poorly designed runtime contextual access control interface can reflect poorly on the user experience of well designed applications. Likewise, a poorly designed application can and seemed to affect the user experience of the access control interface. In this study, participants reported not liking the access control because of the data being requested by applications. This may extend even further into cross application problems. For example, the Tiler application affecting the experience of the Photofire application because of a common access control interface.

Since the line between applications and access control becomes more blurred and affects the user experience when adding context during runtime, it may be beneficial to consider preventing the system from encroaching on the applications and vice versa. One example is establishing a trusted path for the access control. Normally trusted paths are used to prevent users from being tricked by fake warning dialogs and engaging in risky behavior like installing an application on their computer. In this case, they can serve a similar purpose and also help to better highlight the real separation between the application and the access control for the platform. The increased separation may help to mitigate the damage to user experience that poorly

designed applications would have on the access control mechanism.

5.2.6.3 More Awareness and Less Control

This study provides evidence suggesting that participants were more aware of the sharing that was taking place with runtime contextual access control. On one hand, this is a success—encountering the runtime contextual access control interface raised awareness. Yet, that awareness produced some unexpected negative side effects. Many of the participants expressed concern about the sharing that took place before being asked about it. Many of those participants thought the access control was annoying because applications were asking for permissions they did not want to grant. Additionally, I observed that participants' increased awareness may have led to a feeling of less control.

In the previous study in Section 5.1.3, only one participant reported feeling like they were in less control. That participant reported feeling rushed and therefore could not make informed decisions resulting in less control. In this study 25% of the participants reported feeling like they were in less control, albeit for different reasons. The access control dialog in this study by default shared nothing and each permission had to be granted by the participant. Participants had full control over their sharing and nothing was shared without their explicit consent.

It could have been participants' increased awareness of their sharing that lead them to feel like they had less control. All the participants who felt this way did so because they felt unsure about what was being done with the data they gave the application. Additionally, two of them were worried that their sharing might lead to hacking or

inappropriate use. Their feelings are unfortunate because instead of feeling more empowered by the knowledge of what they were sharing, they instead feared it.

While it is possible that as participants increase interaction with access control, they may feel more empowered and less fearful of data sharing, their short-term behavior in this study suggests the opposite. Many of the participants felt obligated to share data out of context to get the application to work. Specifically, many participants did not want to grant access to *Contact Info* for a photo scrapbooking application. Nearly all participants tried at least once to not grant permission. One participant even tried six times before giving up on the task. Several of the participants reported only sharing because they were in the study and felt that I had probably reviewed the applications despite my contrary instructions. However, I have previously observed participants willing to trade data in order to access functionality. In Section 3.1.3.1, I found one of the reasons users added applications was because they were curious about bumper stickers that had been sent to them. Thus, participants may have ignored their better sensibilities because of the study, but in real life, it would likely be the interaction with others that they would trade for the same. As further evidence that this is a major problem, quite a few participants indicated that they would share almost anything in order to get the application to work.

One positive aspect to this problem may exist in that participants were far more aware that sharing data occurred. Sharing out of context became much more salient. While it is not ideal that they are engaging in unwanted sharing, they are aware of the decision to trade their data for functionality. Since uncomfortable sharing affected user experience, we may see developers in the future attempt to minimize out-of-

context sharing specifically to improve the user experience. Users tend to gravitate towards the best user experience over time and it may ironically be the developers who help protect user data.

5.2.6.4 Repetition and Contextual Access Control

An unfortunate but obvious side effect of having runtime contextual access control is that there is more interaction required between the access control system and the user. As the number of features of an application increases, so generally will the number of prompts that a user will encounter. As the number of prompts increase, so does the frustration of the user experience. In this short study, I observed many participants who said they liked the additional choice they were given and the majority of participants wanted to see more access control like this on their applications in the future. However, many of them do not like the repetitive nature of the configuration. This becomes an increasing problem because despite wanting and liking the control they are annoyed that they are forced to repeatedly interact with the system to accomplish a secondary task such as access control.

5.2.7 Conclusion

The goal of contextual access control is to present users with permissions when and where they are most relevant so that they can make more informed decisions. In this section, I evaluated a runtime contextual access control dialog on a social network site. I explored several aspects of the runtime contextual access control including decision making. I found that many participants would grant a permission if they thought it would unlock functionality that they desired to use. In cases where they

were confused or uncertain of the functionality, they would not initially grant the requested permission. Many participants eventually engaged in sharing they did not initially want to because the permission was required. While several participants attribute this to being in a study, several said they would grant most anything to get the application to work.

In this study, I found that participants were very aware of the sharing taking place. However, some also felt as though they were in less control as a result of the awareness. While it may not be desirable for participants to grant everything to unlock functionality, in this study, it represented a cognizant decision to trade private information for functionality. It does not seem that the problem with more awareness leading to a feeling of less control is isolated to runtime or install time access control. Instead, it would apply to any configuration interface that creates increased awareness of information sharing. Since this problem extends beyond a single interface and method of configuration, it could be a topic for further study.

I found three participants quickly became desensitized to the runtime contextual configuration prompts. One of those participants granted each and every permission asked of them. Granting everything required extra effort on the part of the participant since the default policy was not to grant any access to an application. As such, desensitization may be a problem for other runtime contextual access control interfaces. However, there is an interesting finding to the desensitization: two of the participants who reported being on "autopilot" noticed sharing they were uncomfortable with and did not grant those permissions. They also did not express unease with other permissions they did grant. These participants could not explain why they noticed

the items they did not want to share, only that it caught their eye. Future research needs to explore any subconscious decision making that might be taking place and how those results may address the desentization problem in this and other domains.

In this study, the runtime contextual access control further blurred the already uncertain line of where the platform stops and the application begins. As a result, some participants' understanding of the sharing audience was negatively affected. Additionally, many of the participants' experience with the access control was colored by their experience with applications and vice versa. Platforms need to be cautious not to create a negative user experience for other developers applications. Similarly, poorly designed applications can affect the experience of the application platforms. One potential solution to this problem lies in creating a trusted path. Trusted paths highlight the separation between what is part of the application and what is part of the underlying system. Traditionally, this has been desirable because it prevents users from being tricked by providing an understanding of that separation. In this case, highlighting that separation may help to reduce the interdependent user experience that could be beneficial to both the platform and applications.

Finally, it was clear from this study and also from Section 5.1 that repetition was a problem with runtime contextual access control. I needed to find ways to reduce the number of times users needed to interact with the access control dialogs. Alternatively, I found ways to reduce the perceived frequency or annoyance resulting from repetitive interaction with them. An example of this was abandoning the automatic dialog presented in Section 5.1 and having users initiate the dialog for themselves. Unfortunately, it was still not clear how much interaction was too much. This is an

issue I explore more thoroughly in Chapter 6 on notifying users.

CHAPTER 6: NOTIFYING USERS

Runtime contextual access control requires notifying users of permission requests. These notifications can help increase the awareness of data sharing in application platforms for users. In Chapter 5, I tested runtime contextual access control in two different domains. In Section 5.1, I examined runtime access control on the Android platform and in Section 5.2, I examined it for social network application platforms. In both cases, users had problems with the repetition of the access control indicating that the problem extends across both domains. In this chapter, I present a study that helps better understand the problems surrounding notifications and the acceptable usage limits.

The goal of the study in this chapter was to investigate the point at which participants would no longer be willing to be notified about data accessed on their phone during a two week long study. Ultimately, I was not able to use the drop off day as a metric as most participants stayed in the study the full two weeks. I did however learn some valuable lessons as well as observe some interesting qualitative cases which I describe in this chapter.

6.1 Notifying Users

It stands to reason that since there are a variety of ways to try and display information, some will by nature be better than others. For example, a popup box

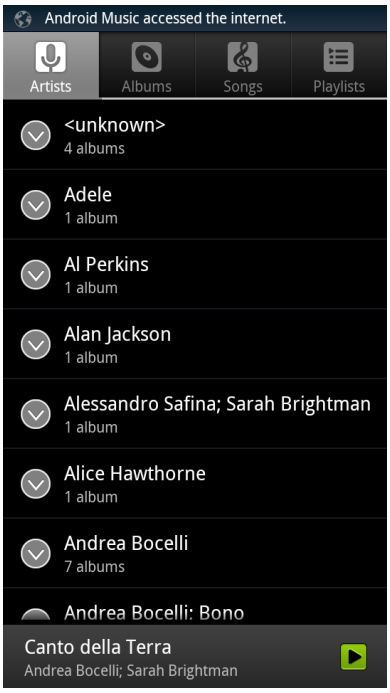

Figure 14: Android notification and ticker example

notifying the user of each and every data access would be quite annoying and have a low level of tolerance with users. In contrast, a ticker provides users with notices at the top of the screen. Users might accept this solution more frequently than the popup box and have a higher level of tolerance for it. The goal of this chapter is to design more informative notifications that are less frustrating by understanding which notification styles have higher levels of tolerance and usefulness. Additionally, we can better understand the repetition problem that recurs in Chapter 5. In this section, I test four notification styles on the Android platform: status bar notifications, icons, tickers, and toasts. I compare these methods to each other in terms of tolerance, usefulness, and understanding. In order to do that, I designed a mixed methods, between subjects deception study I describe in the next section. The main goal was to increase the notifications everyday and look for a drop off day.

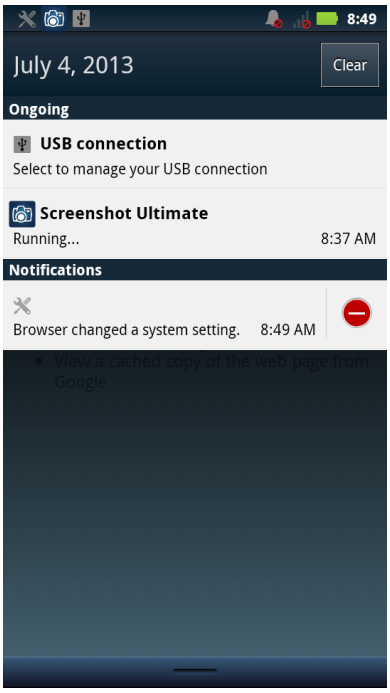

Figure 15: Android icon example

The first type of notification I tested is the status bar notification. This notification has a title and a message text inserted into the user's status bar. The application logo is shown in the status bar and is used to indicate that the application has a notification waiting for review. When the user dragged the status bar down, the notification's title, application logo, and message text together were displayed as seen in Figure 14.

The next type of notification, the icon was inserted into the top status bar representing the type of access that was just utilized. An example of this is shown in Figure 15. If the user dragged the status bar down, they saw the description of the icon as well as the application that generated the notification.

Next, the ticker style notification, was a spin off of the status bar notification. Android does not include a built-in ticker feature, however, it provides ticker-like func-

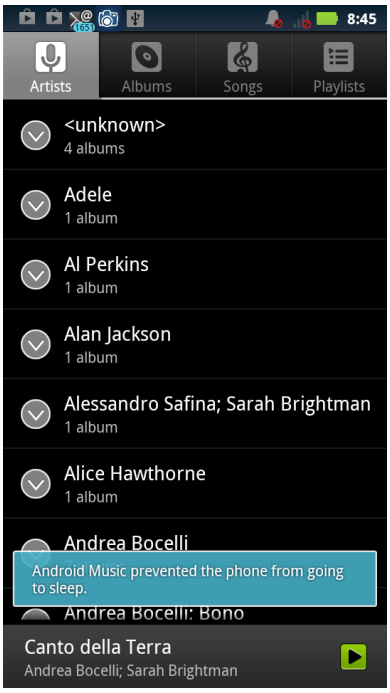

Figure 16: Android toast notification example

tionality. When displaying a new notification with ticker, the notifications message were shown line by line in the top status bar. Once the visualization was completed, the icon was added as previously described. In order to use only the ticker part of this functionality, the application I created issued a status bar notification, waited for it to provide the ticker visualization, then immediately removed the notification. Thus, the messages were ticked, but dragging down the bar showed no notifications. Finally, the toast notification as seen in Figure 16, displayed a short message in the lower portion of the screen. The message was center-aligned and appeared for a few short moments before disappearing.

6.1.1 Study Methodology

This study is designed to compare and contrast different notification styles on a mobile phone. To effectively test each notification style, participants were assigned to

one of four different groups. Each installed an application on their phone that would facilitate the generation of notifications, administration of surveys, and sending of log files to our lab server. Participants were recruited into this study by posting craigslist ads in major cities in North Carolina, South Carolina, Georgia, Tennessee, and Virginia.

Participants were informed they were participating in a study that examines notifying them about what information applications on their phones were able to access. Participants were informed the study would last up to two weeks, depending on how long they chose to stay in the study. Participants were told they could earn $1 for every day they stayed in the study and an additional $6 for not missing any daily surveys. They would also earn an additional $10 for a post study phone interview discussing the previous 14 days. Inducement was provided in the form of an electronic gift card for Amazon. Participants were informed that they needed to complete the final phone interview to receive the study inducement.

Participants first filled out a screener that asked if they had an Android phone, were willing to fill out daily surveys for up to 2 weeks, if they were above the age of 18, and the frequency they use applications other than the phone application and GPS. Participants who were under the age of 18, did not have an Android phone, were not willing to fill out surveys, or used phone apps less than once a day were not eligible to participate. Participants who did not frequently use their phone were not eligible because the data collected from them would not be meaningful.

Participants who met the criteria were formally invited into the study through an email further explaining the expectations of the study, a bulleted summary of

the informed consent as well as the entire informed consent text. Participants were asked to indicate if they still wished to participate in the study. Participants who affirmatively responded were assigned to one of four groups. Each group differed by the notification style they would be receiving in the study. Participants then received an email asking them to install software from the Google Play Store. The application is later described in Section 6.1.1.1. In order to utilize the software, they first had to accept the informed consent a second time, this time by clicking "I Accept." Participants then had to enter a code which would assign them to their respective group. Participants obtained the code by filling out a survey unique to their group. The link to their survey was sent in the same email along with the instructions to install the software.

The survey to receive the code asked participants for their age, gender, ethnicity, occupation, education, and a question used to indicate their level of comfort with technology. The survey also asked for identifiable information used to correlate the logs to participants so we could provide them with inducements, ensure the software was working as expected, and listen to audio comments before the phone interview upon exiting the study. These identifiers were deleted from the dataset after the conclusion of the study and inducement.

For the two weeks following the participants' installing and activating the software, they received notifications about data access by applications installed on their phone. Unfortunately, it was not possible to actually inform users of real data being accessed on the phone since that required modifying the phone's operating system. Projects like TaintDroid [20] could provide actual data for notifications, however, they would

require modifying the phone's existing operating system. In addition to voiding the participants' phone warranties, this would also have been logistically difficult. There would also be no guarantee of data collected that represented notification frequencies measured in the study design.

For these reasons, the notifications about application access to the phone was fabricated. However, the notices were actually based on plausible data access. For example, applications that requested permission to access the contacts would generate notifications regarding access to those contacts. If the application did not request the permission that allows access to contacts, I never generated a notification indicating such an event occurred. Thus, I only provided fake notifications to participants that were entirely plausible, but did not actually occur at the time of the notification.

Each day participants would receive a survey on their phone that allowed them to indicate usefulness, frustration, strange application usage, and general comments. To reduce the burden on participants, the survey could record audio comments instead of using text boxes. After they completed the daily survey, they would continue to receive notifications at the same frequency until the following day. The next day notifications would become more frequent than the day before. Again, at the end of the day, participants completed another daily survey. As they submitted the survey, they indicated whether or not they wished to remain in the study. All survey and usage data was stored on the phone and my application was designed to transmit all logs and audio files to the usability lab server at UNC Charlotte when the phone was connected to a WiFi network.

After the participant disabled my client software, I ensured that all the logs had

been transmitted successfully and scheduled a phone interview that required 24 hours notice. During that 24 hours, I listened to the audio comments and made notes about anything interesting that was useful to ask participants during the final interview. If the logs were not on the server, I asked participants to uninstall the study application and install a secondary application that transmitted the logs files in a different way.

During the phone interview, participants were first informed of the deception and asked if they had any questions about it. Participants were initially informed of the deception to ensure this understanding in the event they wanted to discontinue the interview. Participants were next asked if they were comfortable removing the study software themselves or if they wanted assistance to help remove it. After participants indicated they were comfortable with the removal of the study application, I began the final interview. Participants were asked to talk about the notifications they received and why they decided to withdraw from the study or continue for the full 14 days. While it might seem obvious that the reason was excessive notifications, it was also possible they withdrew due to concerns about what was happening on their phone or other reasons. I asked them to talk about their reactions to general application usage and how they reacted to the notifications generated by the study application. I asked about what concerns, if any, they had with notifications and if they considered them accurate. I then asked them if they wished they had more information about any notifications they received. Finally, I asked them any specific questions I generated from listening to their individual survey responses from both text responses and/or audio recordings.

Several tools were developed in order to facilitate, run, and analyze data from

the study. First, an application for the phone was developed and made available on the Google Play Store. Second, an alternative uploader application was developed to mitigate situations where log and audio files were not transmitted to the server. Next, a server application was developed that would accept, parse, and store log files in a database. The server would notify the client software of successful commits to the database so the client could appropriately mark the study file as transmitted. Finally, an analysis application was designed to pull data from the database, analyze it, and output it in a form useful for my analyses. In the next sections, I describe each of these tools in more detail.

6.1.1.1 Android Client Application

Participants installed an application on their Android phones which collected data for the study and provided notifications about application access to their personal device. This application could be found on the Google Play Store to make it easier for the participant to install the software. Once the participant installed the application, it would do nothing until the participant opened it and accepted the informed consent agreement. Once they accepted the informed consent, the application then prompted them for a code. The code was obtained by filling out the online demographic survey to begin the participation in the study. The codes' purpose was to assign users to one of four groups: toast, notification, ticker, and icon. The codes were simple words like water or pear and had no indication to the participant of anything related to the notification study.

Initially I planned on having the study server assign the group for the participants,

Table 9: Notification frequencies

Day	Frequency
1	1 every 10 mins
2	7.5 mins
3	5 mins
4	2.5 mins
5	1 min
6	50 sec
7	40 sec
8	30 sec
9	20 sec
10	10 sec
11	5 sec
12	2.5 sec
13	1 sec
14	.5 sec

but ultimately decided against this. I wanted to make every effort to ensure that participants did not incur any additional data charges as a result of participation in the study. For this reason the software was designed to communicate with the server only when the participant was on WiFi. Thus, by providing a code no wireless communication using their data plan would be necessary and they could start as soon as they entered the code.

After the participant entered the code into the application, it waited until the following day to start the participant on day one of the study. This delay was to ensure each day's treatment was consistent for each participant. The notification frequency was determined by the day of the study. Table 9 shows the frequencies the application provided notifications.

There were some exceptions to these frequencies. When the participant was on a phone call, notifications were not generated. Also when using maps or navigation,

notifications were not generated for safety reasons. To reduce battery usage and prevent notifications while the participant was not using the phone, the application only displayed notifications while the phone was being used. Additionally, the device settings was honored, so if the vibrator was set to silent it was not used.

To determine what notification was provided to the user, the client application queried the Android Package Manager for the current foreground application and retrieved a list of access permissions granted to that application. A permission was chosen at random and used to match with a pre approved permission list. The permission list contained acceptable permissions that were subjectively accessible to participants. For example, permissions involving mounting or unmounting file systems were not included in this list. If the randomly selected permission was in the list of acceptable permissions, the application generated a notification type based on the participant's study group. If the permission did not match the permission list, the application made several more attempts to randomly select a permission. If after several random tries no permission was suitable, the list of permissions was then iterated over to find a match. If no match was still possible, the application generated a notification that stated an unknown item was accessed by the foreground application. After a full day of treatment, a daily usage survey was generated at 6:00 PM and was available until 11:00 PM. An example of the survey is shown in Figure 17. The survey option was displayed by the study application when the participant used their phone during these hours. If the survey was not completed within that time, it was marked as missed. The survey asked a few questions about their experience with the notifications and allowed audio recordings for each question. The audio files and

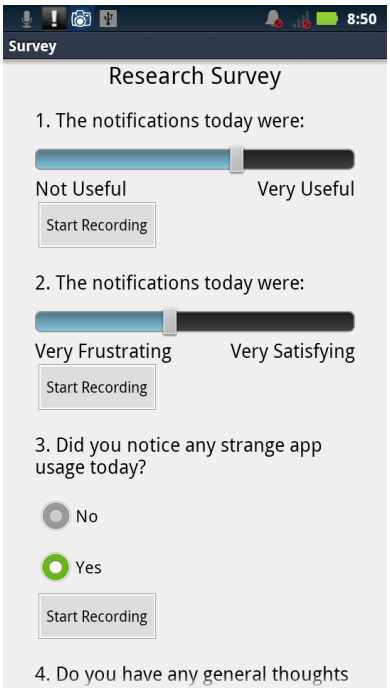

Figure 17: Android notification study survey

responses were logged on the phone. The participant had two options after filling out the survey: they could chose to stay in the study or they could chose to drop the study. If they chose to drop the study, a confirmation screen was shown to confirm their intent to quit participating. If the participant chose to continue in the study, the application would continue to work and stop providing notifications at 11:00 PM until 7:00 AM the next morning. When the participant dropped from the study the application stopped providing notices and the participant was informed that they would still earn credit for all 14 days. Since the goal of the study was to observe drop-off rates, this was considered the best possible and most fair practice. They were also informed to contact me after they quit the study to arrange an interview. A web link let them do this directly from their phone.

The application also logged all of the phone activity that occurred. In particular

we recorded:

- When the phone was started up or shut down
- When the phone screen was turned on or off
- The current running application and notification generated
- When the participant was alerted to a new survey
- Responses to the survey or an indication that it was missed
- If the participant dropped from the study through the survey or directly through the application

All events were notated with the study data and the exact date and time of the event. Periodically, the phone would check to see if it was connected to WiFi and attempt to upload the audio files and log files associated with the study participation. Based on the response from the server, the application marked the files as successfully uploaded or marked them for future upload attempts.

6.1.1.2 Server Application

On the server, a password protected mysql database was used to store the log entries of all participants. All communication from client to server was protected by Secured Socket Layer (SSL) communication to ensure participant confidentiality. The server software accepted the audio and log files and handled them appropriately. In the case of audio files, the files were moved to a non-web accessible location on the server for participant confidentiality. An indication of success was then returned to

the client so that the audio files could be appropriately marked on the participants' phone. The server software parsed uploaded log files to ensure they matched the required format for insertion into the database. It then started a SQL transaction to insert each of the log entries into the database. If for any reason any of the log entries were not inserted into the database, the whole upload batch was rolled back and a failure notice sent to the client. This ensured that only a complete set of logs was inserted into the log database.

6.1.1.3 Alternative Uploader

In some cases, participants did not have access to WiFi or their log files became corrupted and could not be successfully inserted into the study database. In these cases participants were asked to uninstall the study software and install an alternative uploader that was also available from the Google Play Store. The uploader would take each of the audio and textual log files and upload them to a different location on the target server. Instead of attempting to insert these files into the database, the files were sent to the non public file system directly. Again, success was indicated so that files were appropriately marked on the client side. I could then later look at the files and any exception traces to determine if there was a problem with the file or if there were any fixes that needed to be rolled out for the rest of the participants. These files were then inserted into the database with the rest of the log files.

6.1.1.4 Analysis Application

The amount of data collected was extensive. There were millions of log entries that were inserted into the study database. The data, while complete, does not lend

itself to simple SQL queries to extract content of interest. One example is when calculating the phone screen on/off data per day, there are anomalies that prevent simple analysis. The screen may be turned on and a participant may play a game so long that the battery is exhausted and the phone is shut off without the study application being able to log the event. This does not mean that the next phone off event, which could occur the next day, implied that the participant used their phone for 12+ hours. To accommodate intelligent mining of the data, I designed an application that parses the logs and compiles both per day and aggregate values for participant data. Since it is an application, it more easily tracked the state of the phone and made sure erroneous calculations were not included.

The analysis application calculates the average number of notifications per day, a list of applications notifications that were generated, statistics about the types of notifications provided to the participant, length of time a participant used the phone and so on. This data served as the basis for the quantitative analysis. With this data, I examined questions such as, did phone usage decrease as notifications increased? This was a self reported observation by some participants during the phone interview. Finally, the application generated output to an SPSS friendly format for statistical analysis of the observed data.

6.1.2 Results

A total of 160 people responded to the Craigslist ad indicating that they were interested in participating in the study, 111 were invited to participate in the study. Of those, 50 actually accepted and were assigned to one of the treatment groups in the

study. Of the 50, 43 actually filled out the survey associated with their group which gave them a code to activate the software. A few participants ($n = 4$) voluntarily withdrew from the study. One participant's phone broke at the beginning of the study. The cause was unrelated to the study. Another participant returned their phone and decided to buy an iPhone. One participant only had an iPhone, their Android phone was broken and thought they could complete the study on the iPhone. Finally, one participant contacted me to explain they were too busy and had to withdraw from the study. Of the 39 participants remaining only 22 actually finished the study. Those participants were assigned to each the four groups: notification ($n = 6$), toast ($n = 6$), ticker ($n = 4$), and icon ($n = 6$).

The 22 participants' ages ranged from 21-66 with ($M = 26.5$). Participants were primarily female ($n = 15$) and Caucasian ($n = 12$). Other reported ethnicities included African American ($n = 5$) and Hispanic ($n = 4$). Most participants reported having some college education but no degree ($n = 9$), ($n = 7$) indicated they had a bachelors degree, ($n = 2$) indicated they had an associates degree, ($n = 1$) had a masters degree, ($n = 1$) had a professional doctorate, and ($n = 1$) had less than a high school education. A few participants, ($n = 2$) indicated their occupation was in a technical related field and ($n = 5$) indicated they strongly agreed that others would ask them for computer help. Most participants ($n = 12$) were located in North Carolina. Other locations included: Georgia ($n = 3$), Florida ($n = 1$), South Carolina ($n = 3$), and Texas ($n = 2$). One participant chose not to disclose their ethnicity, education, state, and self reported computer help. One participant disabled the ability of the application to generate notifications on day 6 and continued to stay in the

study answering surveys everyday. For this reason, I did not include their data in the following analysis.

6.1.2.1 General Phone Use

During the study, participants used their phones (turned the screen on, then off) a total of 30,490 times and spent a combined 157.87 hours on their phones. On average participants spent 35.88 minutes on their phone per day. Each time participants used their phone they only used it for an average of 18.83 seconds. On average, 6 of the 21 participants shut their phone down at least once during the study period. I based this average on the phone shutdown event since participants in the interview reported that the notification application would at times crash their phones or that their phones ran out of battery. This event could only be triggered by a user shutdown, therefore it is a more accurate reflection for user-initiated shutdown than the phone startup event. In total, participants phones were shutdown 147 times or 7 times per participant. Obviously, the most frequent time to turn the phone off is between the hours of 8PM and Midnight. Given the crashing and battery problem with the study application, I did not determine the frequency of times participants powered on their phones. A comparison of the phone startup and shutdown events indicated that there were about 4 crashes per day across all participants making this a minor but important problem.

Participants' phone activity per hour is shown in Figure 18. The activity is reported in terms of the participants time zones. Therefore 11PM EST and 11PM CDT is reported simply as 11PM. Participants were least active between the hours of midnight

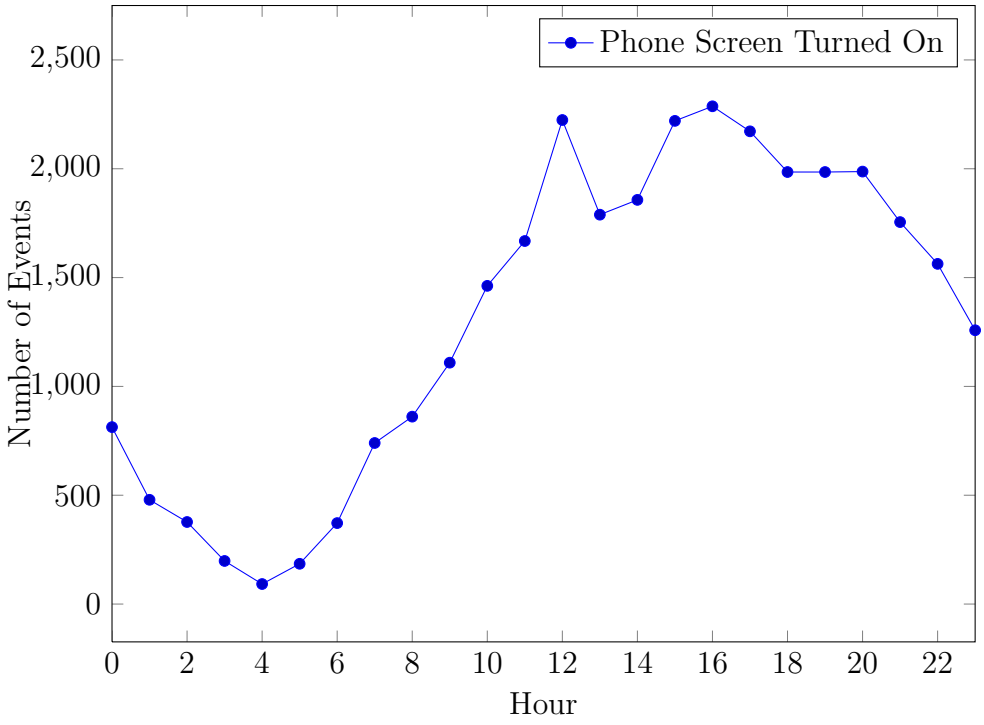

Figure 18: Phone activity per hour

and 9AM. Participants were moderately active between the hours of 9AM-11AM and most active between 11AM and midnight. There is a notable spike in activity at lunch time and again between 2-4PM.

6.1.2.2 Notifications

A total of 615,051 notifications were generated during the study. Notifications were generated for 338 different applications. On average each of those applications was installed by only 2 users highlighting the diversity of Android applications. There were several notable applications that were installed by at least half of the participants: Google Calendar, YouTube, MMS text messaging, Facebook, Google Maps, Browser, and Google Contacts. The top 10 notifications that were randomly generated and displayed to participants were: accessing the Internet, accessing the network state,

Table 10: Average notifications per day across all groups

Day	Average Notifications Shown
1	10
2	11
3	30
4	58
5	177
6	200
7	300
8	568
9	728
10	1285
11	2853
12	5178
13	11715
14	18652

accessing vibrator, accessing WiFi state, writing to the SD card, preventing the phone from sleeping, reading the phone identity, changing a phone setting, reading the phone contacts, and making a phone call—in that order.

As described in Section 6.1.1.1, the notifications were meant to become more frequent over time and cause participants to drop from the study. The frequency schedule is shown in Table 9 of Section 6.1.1.1. The average number of notifications per day resulting from that schedule is shown in Table 10. The number of notifications were modeled with the equation $5.33e^{.58x}$. The residuals $R^2 = 0.99$ indicate that this was a very good fit. If this sampled phone use was indicative of the population parameter this equation can be used to determine the average number of notifications users will encounter when given the frequencies in Table 9 as an input for x.

The purpose of making the notifications grow more frequent was to examine the drop off rate for each of the different notification types. By observing the drop

off rate I hoped to determine what frequencies of privacy related notification were acceptable for each notification type. I tied a $1 incentive to staying in the study each day. I thought that as the notifications grew more frequent, notifications would have become less useful, more frustrating, and participants would discontinue the study. Additionally, I planned to give the majority of the financial incentive at the initial point of agreement to participate. Unfortunately, I underestimated the power of a dollar. Nearly all the participants ($n = 14$) stayed in the study the full 14 days. This confounded my attempt to examine an actual drop-off day with the small sample size in this study. However, there were other metrics I looked at to examine cross and within notification differences.

One such measurement was the total time that participants used their devices. Daily measurements of participants' phone usage were calculated based on moment-to-moment recorded usage data. These daily measurements were combined into quarterly measurements (3 days) to observe changes over time that occurred as a result of the treatment conditions. Figure 19 shows the phone use in seconds per day for each quarter and for each treatment condition. One thing to notice is that the toast groups phone usage time is cut nearly in half from 2,939 seconds in first quarter to 1,550 seconds in the last. This is a decrease in phone use of slightly more than 23 minutes per day per user as a result of having the toast notifications. However, none of the differences shown, including the toast group, was significantly different from any other group or quarter. Such a large reduction in phone usage with a lack of statistical difference could likely be the result of a small sample size ($n = 5$) for this group.

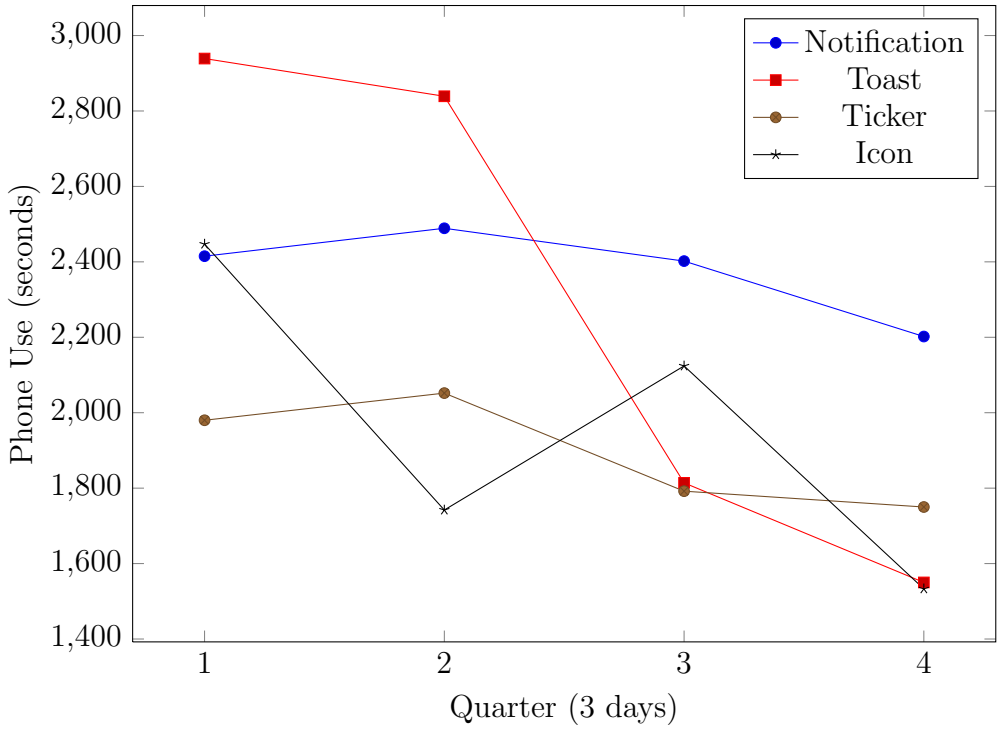

Figure 19: Phone use per group over time

Another measure that I examined was the sentiment scores generated by participants daily on dimensions of annoyance and usefulness. Each study day, participants were asked to adjust a sliding scale with values from 0 to 100 indicating whether the notifications were useful or not useful, and satisfying or annoying. This data only includes ($n = 18$) participants because I completely discarded two of the participants survey data. Both these participants marked the notifications very high in terms of satisfaction and usefulness while simultaneously also indicating they had seen no notifications that day for all study days. This seemed to indicate a clear pattern of not reading the survey. I still include these participants in the former analysis because they did received the treatment like others, they just failed to participate in the daily survey in a meaningful fashion. Additionally, their post-study interviews indicated they did receive and act on notifications, which contradicted their reported

daily survey results.

The two sentiment scores for satisfaction and usefulness sum to 200. In the case of a participant believing the notifications were extremely useful and very satisfying, the score would be 200. In cases where they thought the notifications were entirely useless and annoying, the score would be 0. Again, the values were averaged over a quarter (3 days) to examine difference over time and between groups. The results, shown in Figure 20, indicate a clear downward trend towards a more negative sentiment as notification frequency increases. The change in sentiment appeared nearly identical in every case except for ticker. Because of the outliers mentioned earlier, the ticker group only had ($n = 3$) data points. Again, none of the differences here are statistically significant despite a 20 point or more drop in sentiment. It is interesting to note that the only treatment condition that ever had any positive sentiment was the toast group. Also, this is the same group that caused participants to use their phones on average 23 minutes less per day over the study period.

6.1.2.3 Interview Results

I conducted a short interview after participants had either completed their 14 days or dropped out of the study. Interviews lasted between 10-15 minutes and were conducted after participants were debriefed and helped with uninstalling the study application. In this section, I examine some of the interesting participant cases that occurred during the study.

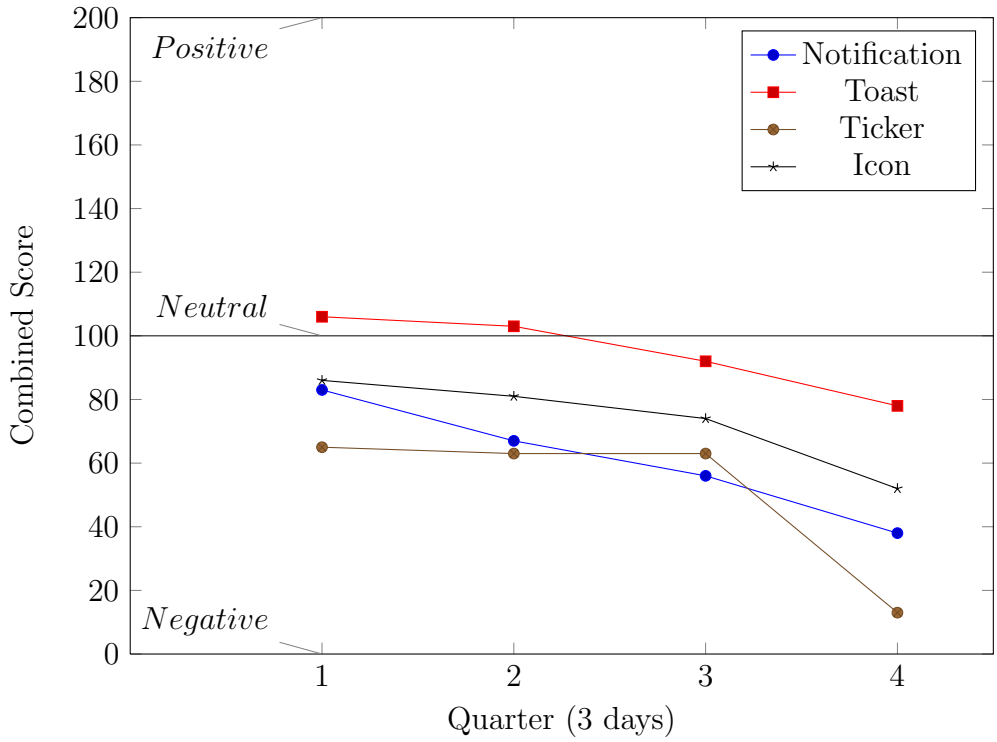

Figure 20: Combined scores of usefulness and annoyance

6.1.2.3.1 Cases 1 and 2

Perhaps two of the most interesting cases involved the participants' extremely strong desire to keep applications on their phones they liked despite the study notifications informing them of inappropriate and unwanted sharing. In both cases, participants had been informed of the same event. In both cases, they reacted to the notifications, but ultimately decided to keep applications installed on the phone despite being informed of the access event.

The first case, C1, involved several different applications. She noticed that Facebook and CNN had taken her photograph. After noticing this, she inspected and deleted many of her applications. The very next day she went and added them all back:

C1:"At first it freaked me out and I went and checked them out. Then I deleted them all... Like Facebook, I took it off, and I took CNN off, and I took a lot of the stuff off but then the next day I put a lot of that stuff back on because I need that stuff... you know?"

Instead of preventing the applications from taking her picture entirely she modified her behaviour with the applications to prevent the unwanted exposure.

C1:"And then what I would do is I would just put my thumb over the camera so they wouldn't get my photo, they would just get my thumbprint you know."

This participant said she trusted that the notifications were real. She explained that there was never a point where she doubted they were fabricated.

Similarly, in a second case, another participant had noticed when we informed her Facebook had taken her picture.

C2:"Facebook said it took a picture... and to be completely honest with you, I'm laying in my bed naked and Facebook takes a picture... and I'm like oh no, so where is this picture going?"

Again the participant modified her behaviour with the application:

C2:"I put a piece of electrical tape across the front camera of my phone."

When asked why she did not just uninstall the application she explained:

C2:"I like facebook... I don't use it that often but when I do I enjoy having it. I was also waiting to talk to you."

Ultimately, she later said she probably would get rid of her phone, yet she used it for nearly two weeks with a piece of electrical tape over it. Even this participant "absolutely" trusted the notifications until the last few days. Then she thought they were being duplicated or they were not as frequent as they should have been at the beginning. Thus, she still believed the notifications were accurate. Even when participants were aware and believed an unwanted event was regularly taking place, the desire to use an application outweighed the inconvenience of simply modifying their behavior.

6.1.2.3.2 Case 3

For case 3, a coincidence increased her belief that the study notifications were real. This participant mentioned that she noticed a notification that an application had called someone but could not find out who. The next day a friend called saying he had missed a call from the participant. As a result, the notifications increased the awareness that applications could access her phone.

> *C3:"My one app kept saying... I got a whole bunch of notifications about that one... about messaging and everything else. When I had downloaded it, the list of permissions was pretty lengthy and I hadn't looked through it. When I got like 20 notifications about it including my contacts, it's messaging, and making a phone call I decided it wasn't the safest app for me because it wasn't really doing anything anyway... so I didn't need, so I deleted that one."*

In this case, the desire to use the application was low so she uninstalled it permanently. She was also informed of access permissions she did not previously consider. She later explained she liked her increased awareness from the notifications:

> *C3:"They were great, especially when you don't understand a certain application or all the permissions they ask you for. It would be nice if there really were notifications like that because sometimes your phone really is doing something on your behalf and you don't know about it... "*

This case provided further evidence that the willingness to tolerate unwanted sharing seems to be balanced with the desire to use the application. This case, like the next, also shows how notifications provided an increased awareness.

6.1.2.3.3 Case 4

Case 4's awareness was also increased, but unlike the other cases, this participant felt powerless to do anything about it.

> *C4:"I was like wow, I didn't know my stuff had so much access to everything. Its like certain applications won't let your phone sleep, and I was like oh wow, everything has access to everything. [..] It kind of gets on my nerves because I wish certain apps can have access to certain things. Like, I don't think a game should have access to your contacts, now email I could understand that, but I just kind of wish every application doesn't have all that access."*

This participant had come to terms with application permissions and many have access to a large amount of her data and phone functionality. When asked if she took

any actions as a result of this she explained:

> *C4:"It was kind of hard to take action because everything had access to everything, and I use all my apps, so I couldn't change anything, or at least I don't think so."*

6.1.2.3.4 Case 5

This final case, although not unique, helps to explain some of the negligible changes in the toast group. Like the other cases, this participant believed the notifications they were receiving were accurate and useful.

> *C5:"At first it was relatively informative, like I said, I didn't know they were mock notifications but it was ok because I saw what was sending off or like you said, could possibly send off to other people that I didn't even know it was doing."*

However, as time went on the toast group experienced problems other groups did not as a result of that notification style.

> *C5:"Toward the end it became so frequent that the little space where the notifications come up was never free. There was a button behind that, I couldn't press it, I had to time it just right between notifications, that split second. If I was reading something online, I'd have to adjust my phone 3 or 4 ways to see what that was behind it."*

This helps to explain why participants in this group reduced their phone usage by nearly half at the end of the study period. It was difficult for them to make use of

features like the keyboard or read information on the screen because of the occlusion created by the notification style.

6.1.3 Study Lessons Learned

Findings in this chapter are limited by the small sample, unexpected user behavior, and lack of statistical significance. Additionally, these findings may not be relevant for other notification styles or types. For example, notifications containing ads for products are likely to be perceived differently than a notifications about data access. The design and purpose of the study was to investigate when notifications became unacceptable for a set of different notification styles. A better understanding of this could help address the repetition problem noted in several of the sections in this dissertation. However, several circumstances led to this not being possible for this particular analysis. First, a very high attrition rate resulted in a smaller sample size. Additionally, participants' behavior regarding the anticipated withdrawal from the study, reduced the ability to analyze notification overload. Most participants only stayed in the study to earn a few extra dollars despite that the majority of the inducement was provided at the beginning of the study. Although these results lack quantitative significant differences, additional participants may provide stronger evidence supporting some interesting patterns that emerged from the data.

Studies in the future need to look at alternative ways of finding the drop off day. There are a few other ways that I might have been able to go about finding the drop off day for notifications. First, instead of increasing the notifications daily I could make the frequency the between subjects independent variable. This would limit the

ability to test more than one notification style, but would allow for participants to more directly experience the large number of notifications instead of becoming slowly accustomed to them. Another option would be to remove the option to withdraw from the study and instead provide participants with a way to disable just the notifications aspect of the software. I would likely develop some other functionality that could be studied at the same time since the expectation would be that participants would continue in the study for several more days.

Studies in the future also need to plan for the extremely high attrition rates. In this study, I was able to understand that rate more directly by tracking a participants progress through the various stages. Others should seek to do the same. Since I was able to understand the rates of attrition, I could more easily over recruit into the study. Additionally, I set deadlines on the invitations since the funding was limited. This again allowed me to move on and invite others without receiving an explicit indication from a participant that they were no longer interested.

It still may be worthwhile to examine alternative ways of measuring drop off as a result of too many notifications. This might involve reducing the incentive per day, however, this may also reduce incentive to complete any portion of the study. In some ways, this highlights an interesting problem with studies using daily surveys and providing daily incentive to continue participation. For this reason, I would recommend designing a more elaborate study that allowed for the disabling of notifications but not withdrawal from the study.

6.1.4 Discussion and Conclusion

The toast group participants' total time using their phone decreased over time to nearly half as much at the end of the study period. Reports from these participants suggested this was a result of an occlusion that occurs on the interface. When the toast message frequency is low, the occlusion is only a minor problem because the toast is short-lived and does not significantly interrupt the interaction with other sections of the user interface. However, as frequency increases, that portion of the user interface never becomes completely visible and results in impeding phone interface usage.

Another interesting pattern was that the rate of sentiment change appears nearly equal across all groups except the ticker group. Because the ticker group only contained a ($n = 3$) after removing outliers, it is unclear if these participants might have exhibited same pattern. In any case, the expected result from the study was that some of the notification types would be more annoying than others. For example, I expected that toasts would cause a negative sentiment faster than an icon because of how prominent the message is on the user interface. However, I only observed a small change in phone usage and the change in user sentiment was largely unclear. A larger sample size might confirm this trend and would be quite interesting.

In the qualitative portion of this study, I noted several interesting cases. Cases 1 and 2 really challenge the notion that increased awareness of application permissions would effect a change of behavior. In these cases, the participants reported being disturbed by the application permission, yet continued to use the application, in once case actually reinstalling it. In some sense, the awareness did change behavior.

These participants covered their phones with tape and fingers to prevent the unwanted sharing with applications they desired to use. Despite any change in behavior, it still illustrates that the user's desire to use applications and be social leads to unwanted granting of permissions. This increases motivation for using runtime contextual access control. With runtime contextual access control, these participants would have been able to continue to use the Facebook phone application, however, they would lack the functionality to take pictures for Facebook until the application requested permission at a contextually appropriate moment.

In Case 3, I saw a similar circumstance to that of Cases 1 and 2. However, in Case 3, I observed that the desire to use the application was low and therefore resulted in its removal. Additionally, I observed evidence that the awareness created by notifications helped to address the desensitization issues of the install time access control. In Case 4 there is further evidence of this awareness, but with a different result. In this case, the participant felt as if there was nothing she could do to take action and so did nothing. Finally, Case 5 provided some explanation of the observed differences in the toast group.

CHAPTER 7: DISCUSSION AND CONCLUSION

In this dissertation, I sought to empower users to create more desirable policies. I presented several studies contributing towards that goal. First, I presented a study examining how users find, install and understand applications on social platforms like Facebook's application platform. I then added more concrete sharing context with community recommendations at install time to help users configure a desirable access control policy. From this study, I found that users had trouble making sharing decisions before they used the application, so I conducted two studies examining the effects of pushing sharing decisions into runtime. Each time I observed positive changes, but also causes for concern. Finally, I conducted a study examining the repetition problem observed in several of the studies conducted. In this chapter, I review the studies to bind the work together, discuss implications for system designs, and discuss directions for future research.

In the time the work for this dissertation was conducted, the Facebook application platform changed in size from 24,000 applications to over 9 million applications and websites. Additionally, Facebook grew to over one billion active users. With growth comes the necessity for change, therefore Facebook modified their access control to meet the demand of the increased usage while at the same time trying to provide users with more fine grained privacy choices. Access control permissions moved be-

yond simply controlling consumption of data; they now also extend to managing the creation and modification of user data. As a result, social network sites are creating new access control models that increase emphasis on when users are sharing their own data with applications in addition to sharing a friend's data. Thus, I now discuss the updates to the model previously presented in Section 4.1.1.

7.1 Updated Model

From a technical perspective, Facebook's access control still consists of users i who each maintain a user profile $Profile_i$. These profiles still contain user information such as interests and photos. Each user still maintains a default policy D_i that limits which attributes $s \in S$ are available to their friends who have installed applications. Recall that S is the set of attributes for consumption, not creation, which have been exposed by the social network site. Since Facebook now also allows modifying a $Profile_i$, we are not just concerned with $R_{App_j,v,t}$ which is the profile attributes for target user t accessible to application App_j through a viewer user v. We are now also concerned with $W_{App_j,t}$ which represents the ability of an App_j to write to target $Profile_t$. Just as R is upper bounded by S, W is bounded by X, the set of attributes exposed as modifiable by the social network site.

Facebook's introduction of fine grained access control permissions necessitates the creation of a user-to-application policy $A_{App_j,i}$. Note that user-to-application policy presented here differs from those created in previous sections of this dissertation. In this case, $A_{App_j,i}$ contains attributes from both sets S and X. However, when dealing with S, the access control makes a distinction for whether a single s has been granted

for a user himself or for a user's friends. In other words, it is now possible to allow your friends' birthdays to be consumed by an application while not allowing your own. Thus the user-to-application policy can now be defined as:

$$A_{App_j,i} = U_{App_j,i} \cup F_{App_j,i} \cup Y_{App_j,i}$$

Where:

- $U_{App_j,i}$ contains the s that i has granted App_j to consume for i himself
- $F_{App_j,i}$ contains the s that i has granted App_j to consume for i's friends
- $Y_{App_j,i}$ contains the x that i has granted App_j to modify of $Profile_i$

Thus, the set of information consumable by an application for a target t by a viewer v is defined by:

$$R_{App_j,v,t} = \begin{cases} P_{t,v} \cap U_{App_j,v} \cap S & : t = v \\ P_{t,v} \cap F_{App_j,v} \cap D_t \cap S & : t \neq v \end{cases}$$

and the set of information modifiable by an application for a target is defined by:

$$W_{App_j,v,t} = \begin{cases} P_{t,v} \cap Y_{App_j,t} \cap X & : t = v \\ \emptyset & : t \neq v \end{cases}$$

Note that in cases where $t = v$ the term $P_{t,v} = S \cup X$. It thus has no real effect in those cases but has been included in the above formalization for clarity. This formalization also does not pertain to notifications/wall posts since both are managed differently.

Facebook's current platform is a quasi-fine grained contextual access control system that permits users to have fine grained choice but only if developers allow. Facebook

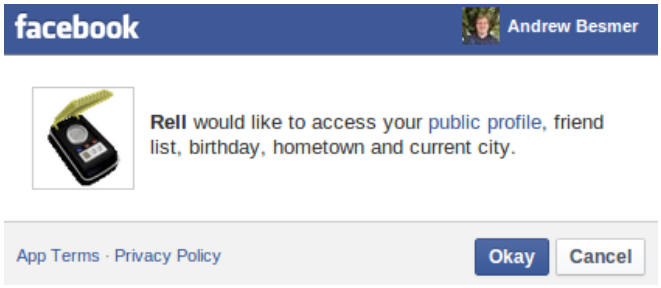

Figure 21: Facebook runtime dialog

also still supports the original style of install time configuration. However, it is now limited to only those permissions requested by the application. Optionally developers can chose to ask for those permissions at runtime. The dialogs used by Facebook at runtime have similarities to those in Section 5.1. In particular, Facebooks dialogs are presented to users when they encounter functionality that requires permissions not yet granted. In addition, the goal of the dialogs is to present just the permissions relevant to the functionality attempting to be used.

Unlike the dialog in Section 5.1, Facebook does not provide any on-screen icon for the user to activate the dialog and grant/revoke permissions. The dialog generated shown in Figure 21, only allows an all-or-nothing decision for the requested attributes, albeit at runtime. It provides very little context about the scope of the sharing, leaving users with only a sentence to reason about what is shared as a result of granting the requested permissions.

Facebook's access control interfaces are designed to facilitate an ever increasing amount of sharing. There is no way to revoke granted permissions at runtime and not all permissions are revocable once granted. To revoke a permission a user must visit the application privacy settings page shown in Figure 22. Even then only a subset

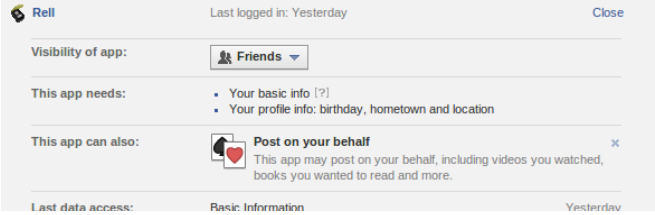

Figure 22: Facebook permission revocation

of the permissions are revocable. This is unfortunate since developers cannot specify a non-revocable permission as not required for interaction. Thus, if the user wishes to revoke any of the non-revocable permissions, they must remove the application entirely. So while the newer access control system is technically fine grained, the interface limits the user's choices.

The contextual access control I have presented supports users' need for increased context in making decisions. Since decisions can change over time, the interfaces designed supported the ability to easily review, and modify an access control policy is important. Facebook's platform could provide this functionality with small changes to their access control system which would better represent the desires of users to have control of their data. However, many of the runtime studies indicated an annoyance with repetition. Therefore, it may be desireable to provide this functionality seperately like in Figure 22 until these issues can be resolved through additional research.

7.2 Discussion

Users use mental models to help interpret and navigate the systems they use. In Chapter 3 the users' interaction with the site and social interaction with friends created false understandings of data sharing with the Facebook application platform.

In Chapter 4, a newer install time interface helped to address some of these problems. That interface helped to refocus users to the application and data they shared and accentuated actual data sharing practices. Using that interface, I observed that users attempted to configure appropriate privacy policies given the name of the application and perceived uses of it. However, even then social interaction was considered in the decision to allow an application or not. For example, I observed users who did not add applications because of the additional space it would take up on their profiles. Thus, the profile's outward appearance was more important than accessing the application.

In Section 5.2, I provided a runtime interface that allowed participants to configure policies as they interacted with applications and used functions that required permissions to access data. In that study a few participants' conceptions of sharing were incorrect as a result of the mental model they created when using the interface. For instance, one participant thought by putting the permissions photographs and location in the "You are allowing" box that the interface was implying "You are allowing your photographs to be shared with Charlotte NC." This misconception can be partially attributed to the interface, but also partially to their knowledge about the social interaction.

Thus, users' decisions about applications are influenced by the access control interface as well as perceptions of the social interaction created by the applications and the desire to engage in that type of social interaction. This is problematic when a user wants to use an application but is worried about the appropriateness of it for people within their social spheres. Facebook's newer interface now addresses this aspect of social application privacy. Users can now select to which audience they wish

to expose the details of an application's usage. Unfortunately, because of the problem described in Section 4.1.1, applications have the ability to gather more information about users.

While fine grained controls as well as strong default policies will help mitigate this property, it still very much exists. Even with the most careful use of applications, it can be hard to control the spread of activity generated within an application to a user's friends. For example, a user that uses an application but sets the applications' visibility to "Only me" still can have their activity in that application viewable by friends who use the application. Going beyond fine grained access control by providing generalization of personal attributes may be one solution to this problem. By perturbing the user identifier, the application will have a harder time accidently (or purposely) exposing user activity within the application.

An encouraging observation of participants in both the install and runtime interface studies was that their mental models were developed quickly in order to accommodate the change in the systems behavior. This means that any permanent understanding of the platform has not yet become deep-seated in users. It is therefore worthwhile to continue to investigate ways of providing awareness of the data sharing—not just for research—but also for actual adoption in production environments. Users simply do not wish to share everything and we should continue to try and provide them tools to make that possible. I presented evidence in this dissertation that increased context is one way to increase awareness of data sharing.

When users lacked context as studied in Chapter 3, they were largely unaware of data sharing with third parties. Participant responses to the surveys also indicated

they preferred to not share some requested information with the applications. When provided with an install time interface emphasizing the context of the application and data sharing, many tried to share appropriately. By providing additional context of the functionality during runtime time, many participants tried to make decisions based on the actual or imagined application functionality. Additionally, participants' understanding of audience was more accurate. This highlights that adding sharing context to runtime access control increased user awareness of data sharing to applications. Awareness is a distinguishing feature of a successful mechanism and I have shown how increasing context also increases awareness.

I observed quantitatively some increased awareness in Section 5.1.3 by demonstrating increased memorability of the permissions requested by platform applications. Increased awareness helps users to more fully consent to decisions to allow or deny sharing data with applications. However, increased awareness can be a double-edged sword. I observed that as users became more aware of sharing, they were also more aware of their discomfort by that same increase in awareness. In fact, 25% of the participants in Section 5.2, reported feeling in less control of their data. By design, they have more control of their data and the awareness of sharing led them to feel like they had less control. While some participants in that study attributed this to actually being in the study, I found in earlier studies that decisions are influenced by the social interaction of the site. Users have, and probably will continue to make decisions on that basis resulting in an uncomfortable awareness of sharing.

7.3 Optimizing User Experience

One of the ways I increased the sharing context for users was by providing concrete representations of requested user data. When not given concrete representations, Section 5.1, many participants reported being able to understand what was asked of them. When permissions are terms accessible to most people like: contacts, photographs, and so on, it is easy for users to think they understand the scope of sharing that is occurring. However, two participants introspection of their own understanding indicates that understanding is not as straightforward as it might seem. This is precisely why concrete representations of sharing are necessary. While the representations I provided users with were rudimentary, they were useful. It would be interesting to see if more user friendly representations of user data have an improved effect on decision making or memorability.

One of the design differences between the install time and runtime contextual interfaces for social network sites was including permission justifications. For the install time interface, I included the ability to read a justification about the requested permission. In that study, participants relied on the name of the application as context for sharing decisions and the justification feature was largely unnoticed by participants. Since participants did not benefit from the justification feature, I removed it when I designed the prototype runtime interfaces. However, when participants interacted with the runtime interfaces, they seemed to more carefully consider whether a permission was needed for the application to function. In cases where they did not understand why a permission was needed for functionality, they did not initially

grant the permission. Because of this uncertainty, two participants from the runtime configuration studies wanted the justification feature to be included. It is possible that the inclusion of the justification feature may help users trust applications, and therefore they would not use the feature. It is also possible that at runtime the increased decision making resulted in a desire to better understand the request. In either case, it is a feature that should be included in other access control interfaces and can improve the user experience.

The two contextual runtime configuration interfaces also differed in design. One of the differences was the choice in how to prompt the user. In the Android prototype, Section 5.1, when encountering functionality that required a permission a prompt was simply generated and shown to the user. As a result, some participants referred to it as a popup and found it distracting. In the Facebook prototype used in Section 5.2, participants still found it distracting, but for different reasons. In the Facebook prototypes, participants found it distracting because of the repetitive nature and that they took time to try and understand why the application needed permission for a particular functionality.

Many of the participants in the Facebook study using runtime contextual access control did not find it distracting and referred to the prompts as "part of the process" of using applications. The most likely reason for the difference between the two was that the Facebook prototypes did not automatically prompt the user with a popup, even though a popup was used. Instead of automatically prompting, the system responded by using visual indicators that additional permissions were needed to continue using the application. For both runtime systems, the permission was

necessary to continue with the task, however, with the runtime Facebook study, the user initiated the configuration dialog. While the difference is subtle, it appears to impact how users view configuration dialogs and the readiness to accept this type of interaction.

7.4 Limitations And Future Work

From a methodological point of view, most of the studies presented in this dissertation are limited by the sample I was able to obtain. While a goal is always to use participants from outside the university population, performing lab based usability studies requires participants to travel to the university. For this reason the participants from the surrounding community are rarely willing to participate. Other studies should be conducted to compare these results with more diverse or different sample populations. Additionally, small sample sizes limits the utility of the statistical and quantitative findings. Although, larger sample sizes would be ideal, the large effect sizes I reported would support significance with larger samples.

Beyond methodology, contextual access control has limitations worth discussing. One limitation of both install and runtime is with having empty context data for permissions. Providing users with more context of data being shared requires creating visualizations using that data. It is not clear how to create visualizations based on data that does not exist. For example, how would you meaningfully visualize a user's contacts when they have none? Even if they are able to rationalize a decision based on a visualization of empty data, they may make a preferred decision because they are not able to see the sharing impact. At a later time, the user may begin

to create contacts and engage in sharing based on the decision made with an empty contact list. As a result, privacy violations may occur. Future research should investigate alternative solutions such as preventing the authorization of empty data items or treating such authorizations as only symbolic in nature to see how well they are received. Another problem that affects runtime contextual access control is the limitations on the permission binding locations. Not all permissions are used in the foreground or visual parts of the interface. Thus, additional research is needed to understand where and how the user can initiate requests for such permissions.

There is also a real problem with increasing annoyance and repetitive interaction. This problem is not unique to install or contextual runtime configuration interfaces. It is however, exacerbated by the runtime contextual access control dialogs. Future research should examine ways to reduce annoyance and repetition of access control dialogs. The study I presented in Chapter 6 becomes a first step to understanding tolerances of access control messages and when the user experience is disrupted by different type notifications. Similarly, we must seek ways to mitigate the desensitization that occurred because of the repetitiveness. Desensitization was an issue in each of the studies testing access control. Even at runtime where the amount of effort to not think about dialogs and allow everything was large, participants still quickly became desensitized. While it is positive that two of the three desensitized users subconsciously noticed undesirable sharing, future research is needed to better understand and address this problem. Future research may even find that these two participants experiences are just anomalies.

Finally, in examining the repetition problem, user behavior and high attrition rates

of participants led to inconclusive findings regarding the acceptability of notifications. It does appear that there are changes in the time participants will use their phone for the toast method of notification, although this was not found to be significant. Rather unexpectedly it also appears that the rate at which sentiment changes is constant across the groups as frequency increases. Unfortunately, without additional participants it is not possible to say whether these changes are significant or confirm the pattern holds for the ticker group. Given the unexpected desire of participants to complete all 14 days the study design likely needs to be modified in order to accommodate better understanding the point in time that notifications are no longer useful.

7.5 Conclusion

In this dissertation, I argued for the need to increase context in access control decisions. I found that users did not understand the sharing that occurred with third party applications and that participants voiced social concerns about disclosure. In order to improve focus on the applications, I increased visibility of the sharing context at install time and runtime to help the user make better access control decisions. By providing context of data sharing through concrete representations of user data, I made decisions more clear and understandable. I also presented quantitative and qualitative evidence that runtime contextual access control increased memorability of requested permissions, illustrating a greater level of awareness over install time configuration methods. As a result, I found that users created more appropriate policies primarily determined by application functionality.

Despite the success of contextual access control, I found several concerns that should be addressed by future research. First, runtime contextual access control suffers from being repetitive in nature and thus can result in a negative or annoying impact on user experience. Additionally, contextual runtime access control may be subject to greater desensitization issues than those encountered on other platforms and interface types. In the case of runtime contextual access control the unexpected speed at which I observed users become desensitized is troubling and may hint at a larger problem with widespread adoption. While subconscious decision making seemed to take place in two of three desensitized users, it is impossible to say whether this would happen in other users and that question remains unanswered. It is a large problem that requires further research.

In each study, I increased the context given to users while making access control decisions. I argue that the policies configured by users at runtime when given the most context were the more appropriate because they better matched the user preference. This was a result of increased awareness provided by context combined with a more restrictive default policy. However, the increase in awareness came at a cost and some users felt like they were in less control over their data. This problem may be mitigated as users begin to feel more empowered with their data and increased control. However, I also presented evidence that the use of these applications is social in nature and so users may also continue to engage in sharing when they are uncomfortable because they desire interaction with both applications and others. In either case, runtime contextual access control can still be considered an improvement since users make more informed choices to exchange their data for functionality.

The scope of the problem I investigated is large and the evidence I presented in this dissertation deals with just two of the vast number of possible domains. While increasing context results in better decision making in these domains, it is not certain what effect, if any, it would have on others. Additionally, the permissions required for configuration in these studies were quite accessible to users. My dissertation only addresses a smaller scope of runtime contextual access control in two application platforms and shows a measure of improvement in sharing awareness with applications and others through application usage. Although future research may address further domains and scope, it is my hope that by providing more context to users to gain greater understanding about sharing decisions, they can begin to better control their digital lives.

REFERENCES

[1] AdMob. AdMob mobile metrics highlight. May 2010.

[2] I. Altman. *The Environment and Social Behavior: Privacy, Personal Space, Territory, and Crowding.* Brooks/Cole Publishing Company, Monterey, California 93940, 1975.

[3] R. Balebako, J. Jung, W. Lu, L. F. Cranor, and C. Nguyen. Little brothers watching you: Raising awareness of data leaks on smartphones. In *Symposium on Usable Privacy and Security*, SOUPS 2013, 2013.

[4] BBC. Identity 'at risk' on facebook. *BBC*, May 2008.

[5] A. Besmer, H. R. Lipford, M. Shehab, and G. Cheek. Social applications: exploring a more secure framework. In *Proceedings of the 5th Symposium on Usable Privacy and Security*, SOUPS 2009, New York, NY, USA, 2009. ACM.

[6] A. Besmer, J. Watson, and H. R. Lipford. The impact of social navigation on privacy policy configuration. In *Proceedings of the Sixth Symposium on Usable Privacy and Security*, SOUPS 2010, New York, NY, USA, 2010. ACM.

[7] D. Boyd and J. Heer. Profiles as conversation: Networked identity performance on friendster. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences, 2006. HICSS '06*, volume 3. IEEE, Jan. 2006.

[8] D. M. Boyd. Friendster and publicly articulated social networking. *Extended abstracts of the 2004 conference on Human factors and computing systems CHI 04*, (January), 2004.

[9] C. Bravo-Lillo, L. F. Cranor, J. S. Downs, and S. Komanduri. Bridging the gap in computer security warnings: A mental model approach. *Security & Privacy, IEEE*, 9(2):18–26, 2011.

[10] CNet. Facebook suspends app that permitted peephole. *CNET*, 2008.

[11] CNet. Google pulls more SMS fraud-related android apps. *CNET*, 2011.

[12] S. Consolvo, J. Jung, B. Greenstein, P. Powledge, G. Maganis, and D. Avrahami. The wi-fi privacy ticker: improving awareness & control of personal information exposure on wi-fi. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, pages 321–330. ACM, 2010.

[13] M. Conti, V. T. N. Nguyen, and B. Crispo. CRePE: context-related policy enforcement for android. In *Proceedings of the 13th International Conference on Information Security*, ISC'10, Berlin, Heidelberg, 2011. Springer-Verlag.

[14] V. C. Conzola and M. S. Wogalter. A Communication–Human information processing (C-HIP) approach to warning effectiveness in the workplace. *Journal of Risk Research*, 4(4):309–322, 2001.

[15] L. F. Cranor. What do they indicate?: evaluating security and privacy indicators. *Interactions*, 13(3):45–47, May 2006.

[16] R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 581–590, New York, NY, USA, 2006. ACM.

[17] P. Dourish, R. E. Grinter, J. Delgado de la Flor, and M. Joseph. Security in the wild: user strategies for managing security as an everyday, practical problem. *Personal and Ubiquitous Computing*, 8(6):391–401, Sept. 2004.

[18] S. Egelman, L. F. Cranor, and J. Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1065–1074, New York, NY, USA, 2008. ACM.

[19] S. Egelman, A. Oates, and S. Krishnamurthi. Oops, i did it again: Mitigating repeated access control errors on facebook. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2295–2304. ACM, 2011.

[20] W. Enck, P. Gilbert, B. Chun, L. Cox, J. Jung, P. McDaniel, and A. Sheth. TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones. In *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, pages 1–6, 2010.

[21] Facebook. Statistics. http://www.facebook.com/press/info.php?statistics.

[22] Facebook. Statistics. http://www.facebook.com/press/info.php?statistics, 2009.

[23] Facebook. Developer documentation. https://developers.facebook.com, 2013.

[24] A. Felt and D. Evans. Privacy protection for social networking APIs. 2008.

[25] A. P. Felt, K. Greenwood, and D. Wagner. The effectiveness of application permissions. In *Proceedings of the 2nd USENIX conference on Web application development*, WebApps'11, Berkeley, CA, USA, 2011. USENIX Association.

[26] B. Friedman, D. C. Howe, and E. Felten. Informed consent in the mozilla browser: implementing value-sensitive design. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences, 2002. HICSS.* IEEE, Jan. 2002.

[27] J. Goecks, W. K. Edwards, and E. D. Mynatt. Challenges in supporting end-user privacy and security management with social navigation. In *Proceedings of the 5th Symposium on Usable Privacy and Security*, SOUPS 2009, New York, NY, USA, 2009. ACM.

[28] N. S. Good, J. Grossklags, D. K. Mulligan, and J. A. Konstan. Noticing notice: a large-scale experiment on the timing of software license agreements. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '07, pages 607–616, New York, NY, USA, 2007. ACM.

[29] R. Gross and A. Acquisti. Information revelation and privacy in online social networks. In *Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, WPES '05, pages 71–80, New York, NY, USA, 2005. ACM.

[30] A. N. Joinson. Looking at, looking up or keeping up with people: motives and use of facebook. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1027–1036, New York, NY, USA, 2008. ACM.

[31] H. Jones and H. Soltren. Facebook : Threats to privacy. *Social Science Research*, December 1:1–76, 2005.

[32] P. G. Kelley, L. F. Cranor, and N. Sadeh. Privacy as part of the app decision-making process. In *Proceedings of the 2013 ACM annual conference on Human factors in computing systems*, pages 3393–3402. ACM, 2013.

[33] B. Kowitz and L. Cranor. Peripheral privacy notifications for wireless networks. In *Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, WPES '05, pages 90–96, New York, NY, USA, 2005. ACM.

[34] P. Kumaraguru and L. Cranor. *Privacy indexes: A survey of westin's studies.* Carnegie Mellon University, School of Computer Science, Institute for Software Research International, 2005.

[35] B. W. Lampson. Protection. *SIGOPS Oper. Syst. Rev.*, 8(1):18–24, Jan. 1974.

[36] S. Lederer, J. I. Hong, A. K. Dey, and J. A. Landay. Personal privacy through understanding and action: five pitfalls for designers. *Personal and Ubiquitous Computing*, 8:440–454, Sept. 2004.

[37] H. R. Lipford, A. Besmer, and J. Watson. Understanding privacy settings in facebook with an audience view. In *Proceedings of the 1st Conference on Usability, Psychology, and Security*, pages 2:1–2:8, Berkeley, CA, USA, 2008. USENIX Association.

[38] H. R. Lipford, J. Watson, M. Whitney, K. Froiland, and R. W. Reeder. Visual vs. compact: A comparison of privacy policy interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1111–1114. ACM, 2010.

[39] T. Mayfield, J. E. Roskos, S. R. Welke, J. M. Boone, and C. W. McDonald. Integrity in automated information systems. Technical report, DTIC Document, 1991.

[40] M. L. Mazurek, P. F. Klemperer, R. Shay, H. Takabi, L. Bauer, and L. F. Cranor. Exploring reactive access control. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 2085–2094, New York, NY, USA, 2011. ACM.

[41] M. Nauman, S. Khan, and X. Zhang. Apex: extending android permission model and enforcement with user-defined runtime constraints. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '10, pages 328–332, New York, NY, USA, 2010. ACM.

[42] H. Nissenbaum. Privacy as contextual integrity. *Washington Law Review*, 79:119, 2004.

[43] D. A. Norman. Design principles for human-computer interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, CHI '83, pages 1–10, New York, NY, USA, 1983. ACM.

[44] D. A. Norman. *The design of everyday things*. Basic Books, Sept. 2002.

[45] J. S. Olson, J. Grudin, and E. Horvitz. A study of preferences for sharing and privacy. In *CHI '05 extended abstracts on Human factors in computing systems*, CHI EA '05, pages 1985–1988, New York, NY, USA, 2005. ACM.

[46] M. Ongtang, S. McLaughlin, W. Enck, and P. McDaniel. Semantically rich Application-Centric security in android. In *Computer Security Applications Conference, 2009. ACSAC '09. Annual*, pages 340–349. IEEE, Dec. 2009.

[47] OPCC. Facebook agrees to address privacy commissioners concerns. http://www.priv.gc.ca/media/nr-c/2009/nr-c_090827_e.cfm, 2009.

[48] L. Palen and P. Dourish. Unpacking privacy for a networked world. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '03, pages 129–136, New York, NY, USA, 2003. ACM.

[49] A. Rabkin. Personal knowledge questions for fallback authentication: security questions in the era of facebook. In *Proceedings of the 4th symposium on Usable privacy and security*, SOUPS 2008, pages 13–23, New York, NY, USA, 2008. ACM.

[50] R. W. Reeder, L. Bauer, L. F. Cranor, M. K. Reiter, K. Bacon, K. How, and H. Strong. Expandable grids for visualizing and authoring computer security policies. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1473–1482, New York, NY, USA, 2008. ACM.

[51] R. Rosenthal. *Meta-analytic procedures for social research*, volume 6. Sage, 1991.

[52] J. H. Saltzer and M. D. Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278– 1308, Sept. 1975.

[53] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *Computer*, 29(2):38–47, Feb. 1996.

[54] B. Schwartz. The tyranny of choice. *Scientific American Mind*, 2004.

[55] M. Shehab, A. C. Squicciarini, and G. Ahn. Beyond User-to-User access control for online social networks. In *Proceedings of the 10th International Conference on Information and Communications Security*, ICICS '08, pages 174–189, Berlin, Heidelberg, 2008. Springer-Verlag.

[56] Sophos. Facebook ID probe shows 41% of users happy to reveal all to potential identity thieves. http://www.sophos.com/en-us/press-office/press-releases/2007/08/facebook.aspx, 2007.

[57] K. Strater and H. R. Lipford. Strategies and struggles with privacy in an online social networking community. In *Proceedings of the 22nd British HCI Group Annual Conference on People and Computers: Culture, Creativity, Interaction - Volume 1*, BCS-HCI '08, pages 111–119, Swinton, UK, UK, 2008. British Computer Society.

[58] F. Stutzman. An evaluation of identity-sharing behavior in social network communities. *Journal of the International Digital Media and Arts Association*, 3(1):10–18, 2006.

[59] F. Stutzman, R. Gross, and A. Acquisti. Silent listeners: The evolution of privacy and disclosure on facebook. *Journal of Privacy and Confidentiality*, 4(2):2, 2013.

[60] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. 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.

[61] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor. Crying wolf: An empirical study of ssl warning effectiveness. In *USENIX Security Symposium*, pages 399–416, 2009.

[62] T. Vidas, N. Christin, and L. Cranor. Curbing android permission creep. In *Proceedings of the 2011 Web 2.0 Security and Privacy Workshop W2SP 2011*, 2011.

[63] A. Whitten and J. D. Tygar. Why johnny can't encrypt: a usability evaluation of PGP 5.0. In *Proceedings of the 8th conference on USENIX Security Symposium - Volume 8*, pages 14–14, Berkeley, CA, USA, 1999. USENIX Association.

[64] M. Wu, R. C. Miller, and S. 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.

[65] ZDNet. Facebook timeline privacy concerns deepen as rollout begins. 2011.

APPENDIX A: SCRAPBOOKING STUDY TASK SHEET

General Information

You are going to use 3 applications obtained from the facebook app store. I do not endorse these apps in anyway they are just three that I chose. You will use the photos in your facebook albums to create a scrapbook in each of the applications. After you use each application you will answer a few questions about it. Try to remember features of each application that you liked and disliked. I will use that information to create a better scrapbooking application for facebook.

AWScrapbook

- Use the application to add 5 or more photos from your albums to the scrapbook.
- View the scrapbook.
- Fill out the associated survey.

Photofire

- Use the application to add 5 or more photos from your albums to the scrapbook.
- Use the application to add at least 2 photos from your friends albums to the scrapbook.
- Use the application to add at least 1 photo you're tagged in to the scrapbook.
- Add a caption to one of the images.
- View the scrapbook.

- Fill out the associated survey.

Tiler

- Use the application to add 5 or more photos from your albums to the scrapbook.
- View the scrapbook.
- Fill out the associated survey.

APPENDIX B: SCRAPBOOKING STUDY SURVEY

1. Participant ID:

2. Which app are you starting with?

    - AWScrapbook
    - Photofire
    - Tiler

3. What is your age?

4. What is your gender?

    - Female
    - Male
    - Prefer not to say

5. What is your ethnicity? (Please choose the best one that applies)

    - American Indian or Alaska Native
    - Asian
    - Black or African American
    - Hispanic
    - Native Hawaiian or Other Pacific Islander
    - White

- Other: Please Specify:

6. What is your occupation?

7. What is your highest level of education?

   - I have not graduated high school
   - GED
   - High school diploma
   - Some college
   - Associate Degree
   - Bachelor Degree
   - Masters Degree
   - Doctorate Degree
   - Other:

8. Are you a student at UNC Charlotte?

   - Yes
   - No

9. On a scale from 1-5, how would you rate the following statement: Others look to me for help when they have computer problems.

   - Strongly Disagree
   - Somewhat Disagree

- Neutral
- Somewhat Agree
- Strongly Agree

10. How many Facebook accounts do you have?

- 1
- 2
- 3
- If more than 3, please specify:

11. How long have you been a member of Facebook?

- I do not have a Facebook account
- Less than 6 months
- More than 6 months but less than one year
- Over one year but less than 2 years
- Over 2 years but less than 4 years
- Over 4 years but less than 6 years
- Over 6 years

12. How often do you use Facebook?

- Several times a day
- Once a day

- Once every few days
- Once a week
- Once a month
- Less than once a month

13. Application 1

- Which of the following best describes your use of the application?
  - 1 - Very Hard To Use
  - 2
  - 3
  - 4 - Neutral
  - 5
  - 6
  - 7 - Very Easy To Use
- Which of the following best describes your use of the application?
  - 1 - Very Frustrating To Use
  - 2
  - 3
  - 4 - Neutral
  - 5
  - 6

  - 7 - Very Pleasing To Use

- Which of the following best describes your use of the application?
  - 1 - Required To Much Effort
  - 2
  - 3
  - 4 - Neutral
  - 5
  - 6
  - 7 - Required Very Little Effort
- Any reason(s) you wouldn't want to use this application?
- Would you change anything about this application?
- If you had any difficulty or problems completing this task describe them here.

14. Application 2

- Which of the following best describes your use of the application?
  - 1 - Very Hard To Use
  - 2
  - 3
  - 4 - Neutral
  - 5

  - 6
  - 7 - Very Easy To Use

- Which of the following best describes your use of the application?
  - 1 - Very Frustrating To Use
  - 2
  - 3
  - 4 - Neutral
  - 5
  - 6
  - 7 - Very Pleasing To Use

- Which of the following best describes your use of the application?
  - 1 - Required To Much Effort
  - 2
  - 3
  - 4 - Neutral
  - 5
  - 6
  - 7 - Required Very Little Effort

- Any reason(s) you wouldn't want to use this application?

- Would you change anything about this application?

- If you had any difficulty or problems completing this task describe them here.

15. Application 3

    - Which of the following best describes your use of the application?
        - 1 - Very Hard To Use
        - 2
        - 3
        - 4 - Neutral
        - 5
        - 6
        - 7 - Very Easy To Use
    - Which of the following best describes your use of the application?
        - 1 - Very Frustrating To Use
        - 2
        - 3
        - 4 - Neutral
        - 5
        - 6
        - 7 - Very Pleasing To Use
    - Which of the following best describes your use of the application?

  - 1 - Required To Much Effort
  - 2
  - 3
  - 4 - Neutral
  - 5
  - 6
  - 7 - Required Very Little Effort

- Any reason(s) you wouldn't want to use this application?
- Would you change anything about this application?
- If you had any difficulty or problems completing this task describe them here.

16. Consumers have lost all control over how personal information is collected and used by companies.

    - Strongly Disagree
    - Somewhat Disagree
    - Somewhat Agree
    - Strongly Agree

17. Most businesses handle the personal information they collect about consumers in a proper and confidential way.

    - Strongly Disagree

- Somewhat Disagree
- Somewhat Agree
- Strongly Agree

18. Existing laws and organizational practices provide a reasonable level of protection for consumer privacy today.

    - Strongly Disagree
    - Somewhat Disagree
    - Somewhat Agree
    - Strongly Agree

APPENDIX C: NOTIFICATION STUDY SURVEY FOR CODE

1. What is your email address?

2. What is the phone number of the Android phone you will use for this study?

3. What is your age?

4. What state do you live in?

5. What is your gender?

    - Female
    - Male

6. What is your ethnicity? (Please choose the best one that applies)

    - American Indian or Alaska Native
    - Asian
    - Black or African American
    - Hispanic
    - Native Hawaiian or Other Pacific Islander
    - White
    - Other: Please Specify:

7. What is your occupation? i.e. Student, Sales Rep, Nurse, etc...

8. What is your highest level of education?

- I have not graduated high school
- GED
- High school diploma
- Some college
- Associate Degree
- Bachelor Degree
- Masters Degree
- Doctorate Degree
- Other:

9. How would you rate the following statement: Others look to me for help when they have computer problems.

- Strongly Disagree
- Somewhat Disagree
- Neutral
- Somewhat Agree
- Strongly Agree

APPENDIX D: NOTIFICATION STUDY DAILY SURVEY

1. The notifications today were: Not Useful → Very Useful (Slider)

2. The notifications today were: Very Frustrating → Very Satisfying (Slider)

3. Did you notice any strange app usage today?

    - No
    - Yes

4. Do you have any general thoughts you would like to share?

    - No
    - Yes

5. Did you notice any notifications at all today?

    - No
    - Yes

Note: Audio recording buttons were placed with each question.