# UniAuth: Building a Human-Centered Identity Management System

Eiji Hayashi

August 2015
CMU-HCII-15-102

Human-Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213 USA

Committee:
Jason I. Hong (Chair)
Lorrie F. Cranor
Anind K. Dey
Stuart Schechter (Microsoft Research)

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

*(This page is left blank for dramatic effect)*

# Abstract

Passwords are the most common form of user authentication today. When passwords were first introduced in the 1960s, computers were a scarce resource, and experts had at most a few passwords to manage. However, today, we are surrounded by many computers and services, and passwords are imposing a growing burden on users. As a way of coping, users choose insecure behaviors, such as writing down passwords, choosing weak passwords, or reusing passwords for multiple accounts. One result is that passwords are now a major source of vulnerabilities in computer systems.

To address this problem, I designed, implemented and evaluated the Unified Authentication Framework (UniAuth in short). The three core ideas behind UniAuth are 1) a user will have one smart device that manages all of their credentials, 2) the smart device can communicate with online services as well as physical devices via a standardized protocol to handle activities related to user authentication (such as authentication, account creation and password updates), and 3) the smart device can use its on-board sensors to improve the security and usability of user authentication to the device. With the UniAuth Framework, users only need to authenticate themselves to their smart devices a small number of times a day. Then, the smart device can communicate with online services and physical devices to perform tasks related to user authentication on behalf of users.

This work consists of three lines of research. The first explored how people use and manage their passwords in their daily life to confirm design of UniAuth. The second investigated how smartphones' onboard sensors could be utilized to adjust the security level of user authentication to the smartphones. Finally, the third involved the design, implementation, and evaluation of the UniAuth Framework through an expert review and a field study. These pieces of research demonstrated that UniAuth could realize secure and usable user authentication, which is one of the grand challenges in usable security, provide a smooth transitional path from password-based user authentication to a better user authentication, and open up new design space in user authentication research in the Internet of Things era.

# Acknowledgements

I would like to thank my advisor, Jason I. Hong, for his support and insightful feedback on this work. It has been a great pleasure working with and learning from his experience and insight for seven years. I also would like to show my deep appreciation to Queenie Kravitz for her unconditional support in my hard times.

I also appreciate all my collaborators, Hirokazu Sasamoto, Nicolas Christin, Rachna Dhamija, Adrian Perrig, Martina A Rau, Zhe Han Neo, Nastasha Tan, Sriram Ramasubramanian, Eric Paulos, Bryan Pendleton, Faith Kursat Ozenc, Oriana Riva, Karin Strauss, A.J. Bernheim Brush, Stuart Schechter, Shahriyar Amini, Vidya Setlur, Zhengxin Xi, Sauvik Das, Ian Oakley, and Manuel Maas.

Most of all, I would like to thank my family. My wife Eriko has provided encouragement through the hard times and shared joy during the good times. My daughters Anna and Sara always gave me their smiles that made me smile even in the hardest times.

Without tremendous supports from these great people, I believe I could not have reached the point I am now.

# Table of Contents

# List of Figures

# List of Tables

# 1. Introduction

Reliable authentication is an essential requirement for secure systems. Passwords are the most common form of authentication today; however, a great deal of past research has shown that people have difficulties in memorizing and managing their passwords in a reliable and secure manner (e.g. [54,65,88]). For instance, people tend to choose simple passwords, re-use passwords among multiple accounts, and fall for phishing attacks.

These problems are caused by the discrepancy between the context where password-based authentication was invented and the one where we are today. When passwords were introduced in the 1960s, computers were a scarce resource, and only computer scientists needed small numbers of passwords. Thus, it was reasonable to design an authentication system based on the assumption that users could bear the burden of managing their passwords. However, today, users have more passwords with more security constraints and requirements, and are facing new kinds of attacks. The original assumption from the 1960s does not hold anymore. Today's modern contexts have made passwords a major source of vulnerabilities.

The computing landscape is changing as well. More and more physical objects are equipped with computation, storage, sensing, and network capabilities. These smart objects are often connected to online services where users have their accounts. Thus, these objects have to perform some sort of user identification and/or authentication; nevertheless, many physical objects do not have input capabilities suitable for password-based authentication such as typing and pointing capabilities.

We are at an inflection point for authentication. The increasing burden on end-users, the growing number of security breaches, and the rise of the Internet of Things all point to the crucial need for better user authentication.

## 1.1 Not Just Passwords, But Password Management

Given these situations, there are those who claim that text-based passwords are fundamentally broken (e.g., [18]). However, I argue that text-based passwords are still good enough as an authentication scheme. Bonneau et al. [22] conducted a comprehensive analysis of password alternatives. In their analysis, they defined three important aspects of password alternatives: usability, security, and deployability. Text-based password schemes have their shortcomings in usability and security but have decent deployability. For instance, with regard to usability, users have to memorize passwords, which significantly limits scalability. With regard to security, text-based passwords are vulnerable to guessing attacks because users are likely to choose weak passwords. Password alternatives tried to address these challenges in usability and security by replacing and/or modifying text-based password authentication schemes; however, I argue that it is also possible to address these challenges without replacement or modification. If passwords are, for instance, long, high entropy, unique among multiple accounts, and users do not have to memorize them, we can address these challenges in the current text-based password authentication scheme. This can be achieved by adding a layer over text-based passwords while using them as backend of the authentication scheme to preserve its high deployability.

My position is that we need a better tool that undertakes the burden of account management including maintaining passwords appropriately. Furthermore, we can build a set of protocols that allows tools to manage accounts without user intervention. Recent changes in technology make this work possible. In the US, smartphone penetration broke 56.4% in 2013. Furthermore, considering recent introduction of smart watches and other wearable devices, it is very likely that in the future, everyone will have some kind of smart device with them. As such, the core insight of this work is to shift the burden of authentication to these smart devices, and to strengthen authentication with the devices using the rich suite of onboard sensors.

## 1.2 Authentication to Physical Objects

In addition to websites, we authenticate ourselves to physical devices. This is not limited to gadgets such as computers, tablets, or game consoles, but includes simple things such as office doors, bike locks, and copy machines. Besides, as represented by the concept of the Internet of Things (IoT), more and more physical objects are going to be connected to the Internet and presumably backend service running on servers. These services are likely to require some sort of user identification or authentication; however, these objects have limited input capabilities (e.g., no keyboard). Requiring user authentication for each of these objects significantly undermines user experiences in IoT environments. Thus, I designed protocols to support authentication to physical devices as well as online services.

## 1.3 Human-Centered Design

Past work has proposed numerous password alternatives; however, virtually none of them have been deployed. This implies that there are discrepancies between researchers' assumptions and realities of how people use passwords. To address this issue, I took a human-centered design approach throughout this work. This work first explored user authentication and account management practices in the wild, collecting data through different types of methodology. Based on these data, I made choices in the design of the Unified Authentication Framework (UniAuth in short). Furthermore, I iteratively evaluated the design with users in practical settings, and re-designed it to improve the framework from both security and usability perspectives.

## 1.4 Unified Authentication Framework

In this work I designed, implemented, and evaluated the Unified Authentication Framework. The core idea behind UniAuth is to have one's smart device handle all tasks related to credential management, such as account creation, authentication, password updates, and account termination, with minimum interaction by the user. The main strategy is to offer machine-readable interfaces that clients can use to communicate with services without human intervention. With UniAuth, users only need to authenticate themselves to their smart devices a small number of times a day

with the assistance of sensors. After the user authenticates themselves to their smart device, the device authenticates them to online services as well as physical devices.

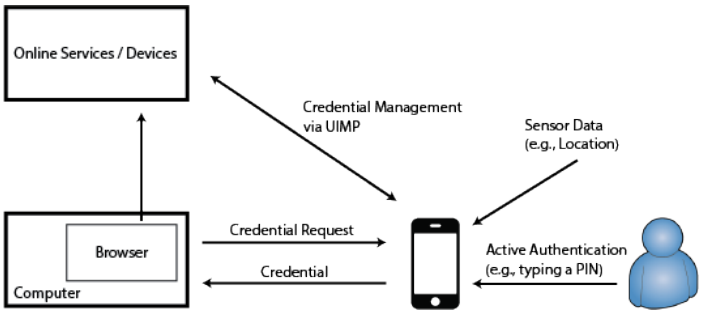



**Figure 1.1 Overview of the Unified Authentication Framework. User credentials are stored in a UniAuth client on a smartphone. A user authenticates themselves to the smartphone, and credentials are sent to server via browser on the user's computer.**

The Unified Authentication Framework consists of three technical components: a Universal Identity Management Protocol (UIMP), UniAuth clients, and a context-aware scalable authentication manager. The UIMP enables UniAuth clients to communicate with services to complete the tasks related to credential management. The observation here is that many aspects of authentication are only human-readable or require manual intervention. Examples include password composition policies, password reset mechanisms, and logins. The insight is to create a protocol that machines can understand and support. If websites and other devices implement UIMP, then any UniAuth client can interact with them through UIMP.

The UniAuth client communicates with services via UIMP and manages accounts on behalf of users including account creation, account management, and account logins. For instance, when users want to create an account, the client automatically configures one on behalf of users by generating a strong and unique password that complies with any required password policies, providing user information requested by the service (such as email addresses), and storing the credentials in a secure manner. When users want to log into web sites from their laptops, the client automatically provides the credentials to their laptops if the users are nearby. If users want to be authenticated to other devices (e.g. iPad), credentials are also sent to them from clients on their smart devices. Finally, in terms of credential management, the devices will provide features that existing password management tools do not

support well. For instance, the devices provide notifications when somebody accesses the account managed by the devices.

By having smart devices manage authentication, we can shift the burden of authentication from end-users to their devices. However, now one major risk is having ones' smart devices stolen. The goal of the Context-Aware Scalable Authentication manager (CASA) is to protect a UniAuth client using a set of passive multi-factor sensing from onboard sensors, modulating the level of active authentication needed based on the importance of the credentials and the situation at hand. For example, for places with reasonable physical security (e.g., physical locks) like workplaces and homes, quick active authentication schemes will be provided to access credentials stored in the smart devices, while for places that users rarely or never go to, or for situations that the system deems risky, highly reliable authentication schemes will be required. Similarly, the New York Times web site may only need a low level of assurance, whereas ones' bank accounts may want a high level of assurance that the legitimate users are nearby.

To demonstrate feasibility and usefulness of the proposed framework, I developed a prototype system that can streamline user authentication (Figure 1.1). The design of UniAuth evolved through multiple iterations of evaluations and redesigns grounded in empirical evidence. Each system component will be described later in this document. Through the development of UniAuth framework, this project made the following research contributions: first, I designed, implemented, and evaluated a Universal Identity Management Protocol that allowed communications between UniAuth clients and services; second, I developed and evaluated a UniAuth client that managed users' credentials using UIMP; and finally, I investigated context-aware authentication management that UniAuth clients used to authenticate users.

## 1.5 Thesis Statement

A Unified Authentication Framework which offers common services to facilitate all aspects of authentication can simplify the user experience of authentication while maintaining a good level of security.

## 1.6 Contributions

This dissertation makes several major contributions:

- Empirical data on usage of passwords and password management tools in practice (Chapter 3)
- A model of probabilistic user authentication based on mobile devices' onboard sensors and user inputs (Chapter 4)
- Empirical evaluations and iterative designs of probabilistic user authentication schemes (Chapter 4)
- Investigation of people's differentiation of necessary security levels for different applications (Chapter 5)
- Design of the Unified Authentication Framework that automates account management tasks (Chapter 6)
- Development and empirical evaluation of the UniAuth client (Chapter 6)

## 1.7 Outline

This thesis is organized as follows: Chapter 2 looks at related work and closely examines situations around user authentication; Chapter 3 reports on the results of two studies where I delved into deeper understanding of how people authenticate themselves, their needs for credential management, and how existing tools and technologies fit their needs; Chapter 4 examines Context-Aware Scalable Authentication, a probabilistic framework for user authentication to mobile devices utilizing data obtained through onboard sensors; Chapter 5 further investigates how people differentiate the importance of access privileges to different services; and finally, Chapter 6 reports on design, implementation, and evaluation of the Unified Authentication Framework, where UniAuth clients on smartphones manage all user authentication needs on behalf of users, and balance security levels of access controls to the credentials stored in the clients, based on sensor data and importance of the credentials.

# 2. Related Work

There have been numerous studies examining user authentication. I have organized this related work into three categories: studies of password usage in practice, alternative authentication schemes, and utilizing contexts for user authentication.

## 2.1 Password Usage in the Wild

Many studies have investigated how people use passwords. Adam and Sasse conducted a study investigating people's attitudes towards password authentication systems. They found that if the authentication systems did not mesh well with people's work practices, they tended to circumvent the authentication systems, which resulted in undermining security [19].

There have been other studies investigating password usage outside of organizations. Gaw and Felten interviewed 49 undergraduate students and found that the students had 7.8 accounts on average, with three or fewer passwords [42]. Florencio and Herley deployed a web browser extension to monitor user authentication for online services. They deployed the extension to roughly a half million computers over three months, and estimated that there were about 25 online accounts per client [39].

There has also been some past work looking at password sharing behaviors. Inglesant and Sasse found that people shared passwords to exchange files and to share web spaces in companies [54]. Singh et al. conducted a qualitative study about how people shared banking passwords with spouses or significant others [89]. Kaye conducted a survey with 122 participants investigating how they shared passwords.

He reported that people shared their passwords with family members, friends and colleagues, across many different kinds of services and devices [59].

One possible way to make users choose secure passwords is to enforce strict password composition policies. For instance, a service can require users to choose passwords that contain numbers, symbols, uppercase letters and lowercase letters. Klein analyzed 13,797 passwords and found that, using a dictionary consisting of 62,727 words, he could crack 25% of these accounts, indicating that many people chose simple passwords [62]. Inglesant and Sasse found that even if organizations enforced strict password policies on users, the policies did not guarantee security for certain attacks (e.g., key loggers) while frustrating users in the meantime [54]. Shay et al. conducted a survey consisting of 470 students to investigate how enforcing a strict password composition policy affected users' perceptions of security. The results showed that the students were annoyed when their university adopted a new password policy requiring more complex passwords, but at the same time, the students felt more secure [88]. Komanduri et al. investigated how various password composition policies affected the strength of passwords created under the policies. They found that simply requiring longer passwords made participants choose passwords with highest entropy among multiple password composition policies such as requiring them to include multiple character types [64]. These findings imply that choosing appropriate password policies is crucial to making password authentication systems more secure and usable. However, this works also showed that people tend to neglect password composition policies unless strictly enforced, or circumvent them by creating passwords that merely satisfy requirements, e.g., adding one symbol at the end to minimize complexities of their passwords when they create them. Additionally, some password cracking tools take common password policies into account to break passwords more efficiently.

## 2.2 Alternative Authentication Schemes

Existing user authentication schemes primarily depend on three types of mechanisms: *what you know*, *what you have* and *what you are*. Passwords are the most commonly used authentication scheme based on *what you know*. Passwords have advantages in their simplicity and convenience [53]. However, many studies have

found that passwords still place substantial burdens on users, resulting in users adopting insecure practices such as choosing weak passwords or reusing passwords [46,54,86].

### 2.2.1 What you know

There are systems that tried to make password management easier by reducing memorization. Systems such as *PwdHash* [80] and *PassPet* [102] decreased the numbers of passwords that users had to memorize by generating account-specific passwords based on a single master password. In these systems, users only needed to memorize one master password; however, these systems relied on one secret (i.e., a master password) to generate passwords for multiple websites. Thus attackers could try to obtain the master password from websites with lower security. For example, attackers could launch online attacks on insecure web sites, which do not restrict number of trials in a certain period to obtain a master password, and then compromised secure and important websites by generating passwords for them using the compromised master password.

Another typical approach to making passwords easier to memorize is to use a mnemonic password, which is a seemingly random sequence of letters generated from a phrase. For instance, a mnemonic password "Ilts@7S!" can be generated from a phrase "I love to ski at Seven Springs!" Yan et al. investigated the memorability and security of passwords composed using different approaches through a lab study with 288 university students. The results showed that mnemonic passwords improved security of passwords without undermining memorability [101]. However, because users are likely to choose common phrases as sources of mnemonic passwords, attackers can guess mnemonic passwords more easily than random passwords [65]. Furthermore, even using mnemonic passwords, users can forget which password is for which account, due to scaling issues and interference effects. Another solution to making passwords easy to memorize is graphical passwords [24,99,45]. Graphical passwords are based on the observations that people are better at memorizing (or recognizing) graphics than at memorizing text [66,79]. However, these graphical password authentication schemes have challenges in actual deployment because of uncertainty about their security and scalability [27,33,36].

One straightforward approach to making password management easier is to store user IDs and passwords in computers. There are numerous password management applications available. For example, all modern web browsers have built-in features for password management, for saving account information and for automatically filling in this information later on. There are also several password management applications which are standalone applications designed to manage login information [7,8,9]. Sometimes these applications store additional information such as screenshots of web pages to better manage users' accounts [1].

Although these password managers help users manage their passwords, users still have to "manage" their password managers. For instance, users have to manually update passwords periodically, and when creating a new account, users have to read password policies to create compliant passwords.

Another approach to mitigate the burden of password management is single sign-on architectures such as *OAuth* [14]. With OAuth, users can be authenticated to multiple services using a single credential registered at an OAuth provider (e.g., Google, Facebook, and Twitter). However, users have difficulty in understanding the security model of OAuth. For instance, when they used credentials for an OAuth provider (e.g., Facebook) to access third party applications, they misunderstood that this may allow the third parties to access their information stored in OAuth providers [73]. Furthermore, if an attacker can compromise a credential for an OAuth provider, he can access all third-party services using the credential.

### 2.2.2 What you have

Other authentication systems depend on *what you have*. *eToken* is a USB device which can be used as a "physical key" to log in [3]. A one-time password token is a device with an LCD, which shows numbers based on the current time and a key stored in the token. To authenticate, users can type the number shown on the device, with a server-side application checking whether that number was actually generated by that device. *RSA securID*, a variant of the one-time password token, depends on both *what you have* and *what you know*. In addition to a number shown on the securID, users have to type their personal identification number to be authenticated [16]. However, there are challenges in scaling this kind of approach across all of a person's accounts; since these tokens require server-side support, it would be impractical to carry a

custom token for each web site, and tokens are too costly. As such, these tokens tend to be used only for accounts with very high security requirements.

Finally, there are systems that use smartphones for authentication. *Phoolproof Phishing* is an authentication scheme designed to prevent phishing attacks, key loggers, and other kinds of malware [80]. A user selects what site to log in to on their mobile phone, which opens the website on a local computer. The mobile phone also checks the site's certificate to verify that the opened site is a legitimate site, at which point the user can log in normally. While Phoolproof Phishing prevents phishing attacks, it does not provide support for credential management.

*Clef* is a smart phone based authentication system [2]. When a user clicks "login with Clef" at a website supporting Clef, the website displays an animation pattern. The user captures the pattern with a camera on their smartphone. Then the Clef application on the smartphone logs into their account at the website using OAuth. Thus, to be able to log into an account with Clef, a website has to support both Clef and OAuth, and users have to have the Clef application installed on their smartphone and configured appropriately.

The FIDO alliance is a group of companies seeking to replace password-based authentication with public-private key pair based authentication [6]. To log into an online account, a user first authenticates to their FIDO device, typically using some biometric. Then the device completes authentication to the online account using a pre-shared public-private key pair. Although this approach could be more secure than password-based authentication, it requires server-side modifications and adoption of FIDO devices at the same time.

### 2.2.3 What you are

Finally, biometrics authenticates users based on *what you are*. Biometrics can be classified into two categories [57]. The first category leverages users' *physiological* properties, such as fingerprints, face, or voice (e.g., [28,74]). Guillaume et al. developed face authentication with a 2.4% equal error rate under controlled background and light conditions; however, the equal error rates became 13.49% under uncontrolled background and light conditions [52]. Wan et al. demonstrated that voice authentication could achieve a 4.03% equal error rate using GMM

LR/SVM with spherical normalization [97]. Schmidt et al. developed *HandsDown* that identifies users based on hand contours in a shared table-top surface [87].

The second biometric category is *behavioral* biometrics where users are authenticated based on their behavioral characteristics, such as key-typing pattern [82] and gait patterns [40]. Mäntyjärvi et al. achieved 7% to 19% EER for gait pattern authentication [67]. Generally, these approaches have higher usability compared to physiological biometrics because they do not require explicit user inputs. However, they are hardly adopted in practice since their security is not as high as physiological biometrics.

## 2.3 Utilizing Contexts for User Authentication

Some online services already modulate authentication level in specific circumstances. For example, many bank websites ask extra questions when users log in from new network IP addresses. Similarly, Facebook asks additional questions when accessed from unusual IP addresses [4]. In this technique, a user must identify several of their friends' photos before being allowed to login.

In mobile computing, several systems have used contextual information to improve security, for example, proximity has been used to authenticate users [22, 29, 53]. Similarly, Seifert et al. proposed *TreasurePhones*, a system that protects information on mobile phones based on users' locations as detected by near field communication technology [94]. Buthpitiya et al. demonstrated that a system could detect anomalous activities (e.g., a phone being stolen) by analyzing users' location histories using an n-gram model [26]. Riva et al. proposed *Progressive Authentication* that combined light, temperature/humidity sensors, touch screen events, login events, a microphone, and Bluetooth IDs using a SVM model to authenticate users [84]. Jakobsson et al. also proposed implicit authentication [55]. Their core idea was to utilize users' behavior patterns for authentication. They considered two behavioral features derived from a mobile device: time since the user last checked email, and GPS location. The two feature scores were combined through a weighted linear function to calculate an overall "authentication score", which was then compared to a pre-defined threshold to determine whether a user should be authenticated. These

works demonstrated that contextual information captured by smartphones' on-board sensors could be used for user authentication; however, no existing work has proposed a generalizable framework to use contextual information for user authentication.

# 3. Password Usage in the Wild

While past work has examined password usage on a specific computer [39], web site [62], or organization [54], there is little work examining overall password usage and password management practices in a wild. In this chapter, I report on two studies[1] investigating password usage and management practices in a wild. For the first study, I conducted a diary study where I asked participants to record contexts when they use passwords in their daily life. Through this study, I examined password usage in practice, and offer new findings based on quantitative analyses regarding how often people log in, where they log in, what password aids they were using, and how frequently people use foreign computers.

One interesting finding in the first study was that a relatively small segment of participants were using tools in managing their accounts. This finding led to the second study where I further investigated opportunities and challenges in password management tools. I conducted semi-structured interviews with 22 participants who were already using one or more password management tools. In my interviews, I looked at three issues. The first issue was what kinds of password management tools they use, as well as the range of behaviors around these tools. The second issue was which accounts they shared with others (or others shared with them), and why. Very little past research has examined this aspect of sharing [54,89,90], and our conjecture was that password sharing was more common and had a richer range of

[1] The results reported in this chapter appeared in [46] and [50].

behaviors than has been previously reported. The third issue was what kinds of features participants liked and disliked for password management applications. I had some surprising findings in terms of features, where some features that already existed were not highly desired (e.g. generating strong passwords), whereas other features that did not exist were highly ranked (e.g. notifications of when an account was used). These data helped me making appropriate design choices in developing UniAuth Framework.

## 3.1 Study 1 – Password Usage

This diary study investigated in what contexts people use passwords in their daily lives, examining password usage across all computers, services, and settings. My analysis also provided novel data, such as where people log into their accounts and how frequently they used computers owned by somebody else such as public computers or their friends' computers. Furthermore, the analysis confirmed or updated some password statistics, such as the number of accounts people had.

### 3.1.1 Method

In the diary study, I provided small diaries to participants and asked them to carry the diaries throughout their day. The participants were asked to record each *password event* in their diaries when they log into their accounts using desktop computers or laptops. Password events included typing passwords to log into online accounts and computers, unlocking screensavers, and logging into applications (e.g., email clients). Even if an authentication system automatically filled passwords, I instructed the participants to record the password event, as long as the participants had to click a button (e.g., login or OK). In contrast, if a system automatically logged into an account, I asked them not to record the password event since some participants would have difficulty distinguishing if they actually went through an authentication process. On the first day of the study, I asked participants to clear cookies in their browsers to log out from all of their online accounts.

On a password event, participants were asked to record their current context, including their location, the purpose of the password event, the type of computer they were using (e.g., personal or public computer), and whether they used a

password aid (e.g., a sheet of paper with a list of passwords, or a piece of software to help with passwords).

The study lasted for two weeks from July 1st, 2010. At the end of the study, I asked participants to complete a post-survey. The participants were compensated $20 USD for completion of the study.

### 3.1.2 Participants

I recruited 20 participants using a university recruitment web site. Nine participants were male and 11 participants were female. The participants consisted of 12 university students, three housewives, two university staffs, two self-employed, and one unemployed. Their ages ranged from 21 to 59 with a median age of 29. In the survey, I examined the participants' expertise levels by asking whether participants agree or disagree with various statements (see Table 3.1). In general, the participants were comfortable using computers and estimated their expertise as average. Nine participants used computers at their workplaces, 15 participants used computers at their home, and 11 participants carried around mobile computers regularly.

| Sentence | Mean (SD) |
|---|---|
| I'm comfortable with using email | 4.6 (0.49) |
| I'm comfortable with using web browsers | 4.6 (0.49) |
| I'm comfortable with purchasing products on-line | 3.8 (1.09) |
| I'm comfortable with configuring computers | 3.3 (1.00) |
| How do you estimate your computer expertise? | 3.1 (0.89) |

**Table 3.1 Self-reported level of computer expertise (one denotes strongly disagree and five denotes strongly agree). For the bottom row, one stands for novice and five stands for expert.**

Figure 3.1 shows the distribution of password events per participant observed in the study period. The numbers ranged from 11 to 169 password events, with a mean of 75 ($\sigma$=35.3). The most common purpose of these events was to log into online services (75.6%), followed by to log into computers (20.3%), to use applications on computers (7.4%), and to unlock screensavers (3.3%). The small ratio of "unlocking screensavers" implied that a small number of participants were using passwords to unlock screensavers. In my post-survey, I also found that only three participants had screensavers that required passwords to be unlocked. This data may imply that people do not need the screensavers with passwords because of their work practices

or environments. However, there seem to be opportunities to design better user authentication systems for screensavers to facilitate its adoption.

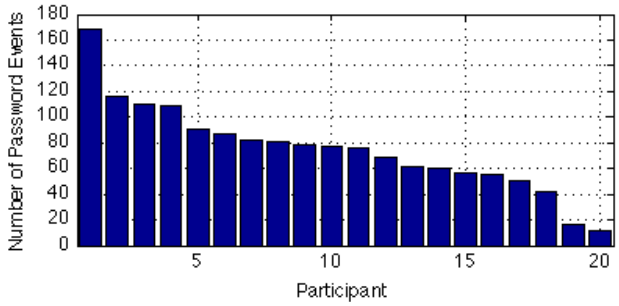



**Figure 3.1 Distribution of password events across 20 users, sorted from most events to least. Most users accessed their accounts 40 to 110 times over a two-week study period.**

### 3.1.3 Online Accounts

In the study period, I observed 172 online accounts in total. The numbers ranged from 3 to 16 with a mean of 8.6 accounts (see Figure 3.2).

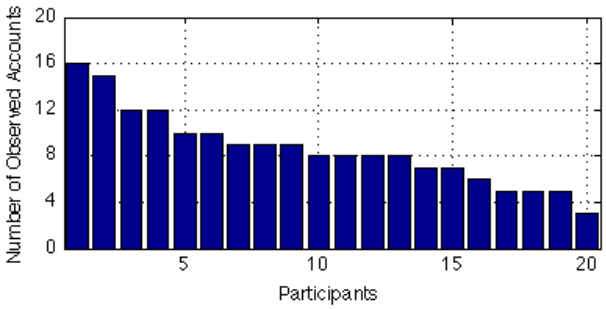



**Figure 3.2 Distribution of the number of the observed accounts for each participant. The participants were sorted according to the number of the accounts, so this x-axis does not correspond to the x-axis in Figure 3.1.**

Figure 3.3 shows the relationship between the number of days and the average number of accounts per participant observed. The dashed lines show one standard deviation. In the first two days, I observed five accounts. After that, the number steadily increased to 8.6. Florencio et al. [39] reported that it took 60 days for this number to be saturated. They also reported that they observed about 70% of the online accounts in the first 14 days. Thus, I estimated that participants had about 11.4 online accounts. This estimated number of online accounts is slightly larger than in Gaw et al.'s study conducted in 2006 (7.8 accounts per a user) [16].

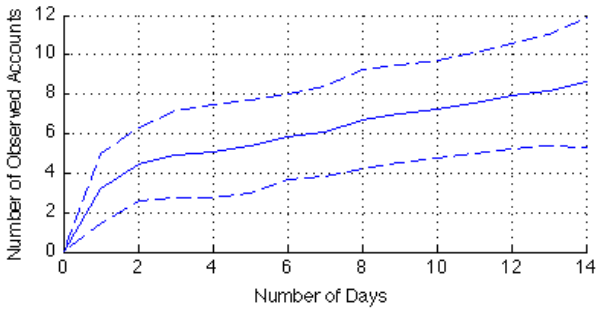



**Figure 3.3 Cumulative numbers of the online accounts that were observed by day. The dashed lines stand for one standard deviation for each day.**

One implication here is that, while password aids is better to scale for users with vastly more accounts, systems that can help people with about a dozen accounts would still be valuable. Another implication is that novel authentication systems should be tested for interference and memorability with roughly this number of accounts as an upper bound, rather than just for one (which is typical in many studies).

To facilitate analysis, I first categorized each online account collected in this study according to Google's Ad planner categorization. For the web sites not included in the Google's list, I manually categorized them using the same scheme. Then, I coded these categories into eight broader categories as shown in Table 3.2. "Email/Messaging" denotes webmail services, such as Gmail, or messaging services, such as Twitter. "Online Community" includes social networking sites or online forums. "University / Company" denotes web pages specific to universities or companies, such as online course registrations or work hour management system. "Portal" denotes pages such as MSN or Yahoo top pages. "Application" denotes online applications provided on web pages, such as Google Docs or Doodle.

Table 3.2 shows the number of password events in each category, as well as the number of accounts per category. Email and messaging had the largest number of password events, with 40% of all password events. Note that there were 33 accounts in this category, as some participants had multiple email/messaging accounts.

Email/messaging, online community, and university/company were the three most frequently used categories, consisting of 68.4% of the total number of the password events while covering 45.6% of the accounts.

| Category | # of Events | # of Accounts | Events/Account |
|---|---|---|---|
| Email/Messaging | 418 (40.4%) | 33 (19.1%) | 12.7 |
| Online Community | 165 (16.0%) | 29 (16.7%) | 5.7 |
| University/Company | 128 (12.4%) | 17 (9.8%) | 7.5 |
| E-commerce | 95 (9.2%) | 35 (20.2%) | 2.7 |
| Portals | 73 (7.1%) | 10 (5.8%) | 7.3 |
| Applications | 69 (6.7%) | 16 (9.2%) | 4.3 |
| Finance | 37 (3.6%) | 14 (8.1%) | 2.6 |
| Others | 49 (4.7%) | 19 (11.0%) | 2.6 |
| Total | 1034 | 173 | 6.0 |

**Table 3.2 This table shows the number of accounts, number of password events, and mean number of events per account for each category. While email/messaging consisted of 19% of the accounts, it consisted of 40% of the password events.**

### 3.1.4 Locations and Computers

I also asked participants to record their locations as well as the kind of computers used at password events. Table 3.3 shows the locations and the number of password events observed at those locations. 84.3% of the events were observed at either home or office. In contrast, only 6.9% of the events were observed in public places, such as libraries. Even if I include school as a public place, the total is 13.1%. Among the 20 participants, nine participants accessed their accounts only from home or office.

| Place | # of Events | Ratios |
|---|---|---|
| Home | 889 | 59.2% |
| Office | 377 | 25.1% |
| Public Places | 104 | 6.9% |
| School | 93 | 6.2% |
| Others | 37 | 2.4% |

**Table 3.3 Categorization of locations where participants accessed their accounts. 84.3% of the password events occurred either in home or office.**

| Computer | # of Events | Ratios |
|---|---|---|
| Personal | 996 | 66.4% |
| Work | 413 | 27.5% |
| Public | 60 | 4.0% |
| Friends' | 17 | 0.9% |
| Others | 14 | 0.9% |

**Table 3.4 Categorization of computers that the participant used at password events. 93.9% of the time, the participant used either their personal computers or work computers owned by them.**

Table 3.4 shows the type of computers that the participants used. I defined "personal computers" and "work computers" as computers primarily used by the participants

for personal purposes or work purposes respectively. Public computers were computers that anyone can access, such as those in libraries. Friends' computers were computers owned by participants' friends.

I observe that 93.9% of password events occurred on either personal or work computers. Although there were 91 (5.8%) accesses from foreign computers (a public, a friend's, or other computer), two participants accessed their accounts from foreign computers 45 times in total accounting for half of the data. In contrast, nine participants never accessed their accounts from foreign computers. Naturally, those participants overlapped with those who accessed their accounts only from home or office where they have personal or work computers.

Given that the vast majority of our participants only use their personal or work computers, and close to half of our participants do not login in public places at all, these findings suggest that if I can make the login process easier just for users' work and home computers, it can provide considerable benefit to a large number of users.

Furthermore, with the growing diffusion of location-based services, these findings suggest that I may be able to use one's current location at home or work as an additional factor in authentication. For instance, it is possible to build a screensaver that does not require a password to be unlocked when a laptop is at home or work, but would require a password or perhaps additional authentication in other places. Additionally, since many individuals access their accounts in similar contexts (e.g., the same locations, on the same computers, with the same printers and devices nearby), authentication systems could utilize these contexts to modulate the level of authentication required. These approaches could potentially improve the security of an authentication system without adding burden to users.

### 3.1.5 Password Aids

In the post-survey, I asked participants what password aids they used to manage the accounts observed over the study period. I also asked them to self evaluate how concerned they would be if someone obtained access to that account. Three accounts were missed due to the lack of data in the survey. Thus, 169 accounts were analyzed here.

| | 5 | 4 | 3 | 2 | 1 | Total |
|---|---|---|---|---|---|---|
| Not using password aids | 46 | 27 | 22 | 6 | 1 | 102 (60.3%) |
| Browsers' auto-fill features | 21 | 5 | 12 | 10 | 2 | 50 (30.0%) |
| Writing down on paper | 5 | 1 | 0 | 0 | 1 | 7 (4.1%) |
| Dedicated password manager | 0 | 0 | 0 | 0 | 0 | 0 (0.0%) |
| Others | 4 | 2 | 0 | 3 | 1 | 10 (5.9%) |
| Total | 76 | 35 | 34 | 19 | 5 | 169 |

**Table 3.5 The numbers denote the numbers of accounts in the categories. The rows denote types of password aids. The columns denote participants' self-evaluation of how concerned if someone obtains access to these account, 5 denotes *very concerned* and 1 denotes *not concerned at all*.**

Surprisingly, Table 3.5 shows that, for 60.3% of the accounts, participants did not use any password aids. This finding carries two implications. First, according to the survey, all participants except one reused their passwords for multiple accounts. Given that people chose not to use any passwords aids for important accounts, this suggests that people under-estimated the risk of reusing passwords compared to the risk of using password aids. Although there is no data about which passwords were reused, (i.e., important vs. not important accounts), educating users about these risks seems prudent. Second, the low rate of adoption of password aids suggests that there is still a lot of room for helping people, and that examining barriers to adoption of password aids may be a fruitful approach to improving security.

### 3.1.6 Limitations

One of the biggest limitations in our study was participants' demographics. Although our participants involved university staff and domestic residents, 60% of the participants were university students. Thus, our participants may not represent the general population.

Another limitation is that our study was based on participants' self-reports. Although I designed the diaries easy to record events to facilitate accurate reports, the participant may have under-reported.

Moreover, our study was limited to password events using computers. As other forms of computers, such as smart phones or tablet computers, people would have to use their passwords in wide variety of contexts. Further investigation would be necessary for password usages on these devices.

Finally, our study period could be short for some analyses. In the analysis of the number of accounts, the number did not saturate in the study period. Similarly, in the analysis of password aids, I may have observed larger number of infrequently used accounts, for which the participants might use different types of password aids if the study period is longer.

### 3.1.7 Summary

Through a diary study, I collected 1,500 password events, which illustrated how participants used passwords in their everyday lives. The analyses of the data provided several implications about user authentication systems.

- People log into their accounts mostly from their personal or work computers
- People log into their accounts at very limited number of places
- People use password aids for only a small portion of their accounts

## 3.2 Study 2 – Password Management Tools

Using password management tools is one of the promising approaches to mitigate the burden of passwords. These tools range from writing down passwords on a piece of paper to specialized applications that can manage account information. However, there has been little past work investigating how people manage their passwords using these kinds of tools in the wild. Also, my previous study showed that adoption of password management tools was limited. Understanding current practices of password management tools, their affordances, and their weaknesses could help us develop better password management tools that are more useful, usable, and desirable.

In this section, I report on the results of a study investigating how people manage their passwords using variety of password management tools. I conducted semi-structured interviews with of 22 participants who were already using one or more password management tools. In my interviews, I looked at three issues to better understand challenges and opportunities in designing password management tools.

- What kinds of password management tools they used, as well as the range of behaviors around these tools.

- Which accounts they shared with others (or others shared with them), and why.

- What kinds of features participants liked and disliked for password management applications

### 3.2.1 Method

To investigate issues around password management tools, I conduct semi-structured interviews. The previous study showed that a rather small segment of people used password management tools. Thus, to obtain rich data, I specifically recruited participants who managed at least five accounts using some kind of password management tool. I defined password management tools as systems that managed users' account information without depending solely on participants' memorization. Examples included using a sheet of paper, memos, sticky notes, browser auto-fill features, text files, and applications such as LastPass [9], KeePass [7] and KeyChain Access [8].

Participants were asked to bring their password management tools to the interview. I opted for this procedure so as to help prompting participants' memory, to help them describe their behaviors more accurately. I also asked participants to refer back to lists of accounts in their password management tools and to go through their accounts one by one in answering questions when necessary; however, I did not examine their password manager directly to avoid violating their privacy and security.

In my interviews, I focused on collecting qualitative data about password management behaviors and tools, to understand the range of behaviors as well as interesting ways that people used these password management tools. The interviews were comprised of three parts. First, I probed what password management tools participants were using. Second, I investigated what accounts people shared, with whom, and in what context. Third, I asked participants to evaluate features that could be supported by password management applications based on how they had been managing their accounts in practice.

To be consistent as much as possible, I showed questions on a display in the structured part of the interview; then, asked follow-up questions to obtain rich qualitative data about the participants' perceptions on password management tools. Each interview took about one hour. All the interviews were audio recorded and transcribed. Based on the participants' responses, I created an affinity diagram to extract underling themes in the password management domain.

### 3.2.2 Participants

I recruited participants via a participant recruitment website at Carnegie Mellon University as well as sending emails to local mailing lists. I recruited 22 participants, with age ranging from 18 to 62 years old with median age of 27. Three were undergraduate students, five were graduate students, 12 were full-time employed, one was self-employed, and one was unemployed. For the employed, they had a variety of professions, such as a photographer's assistant, a social worker, an office administrator, and a programmer. Six of them had technical majors. They also reported that, on average, they used computers 2 to 14 hours a day on weekdays with a median of 7.5 hours, and 1 to 14 hours a day on weekends with a median of 4.5.

### 3.2.3 Taxonomy of Password Management Tools

In the first part of our interviews, I asked about tools that participants were using to manage their login information including passwords. More specifically, I asked about:

- Tools that they used to manage their login information
- Accounts that were managed using the tools (as well as those that were not)
- Pros and cons of using their password management tools
- How they generated and updated passwords
- Their experiences in using their password management tools

Along with these issues, I asked many open-ended questions about their thoughts on these issues to deeply understand their behaviors and rationales behind the behaviors. Most of the participants' passwords were for online services. Thus a majority of responses were related to online accounts; however, they included ones for other types of accounts, such as login accounts for computers.

Through the interviews, I found that participants were using combinations of five types of password managers for different types of accounts (Table 3.6). Also, Table 3.7 shows the number of participants using each combination. For instance, the top-left cell shows that 12 participants used both text files and browsers to manage their passwords. There were four participants who used more than two tools, so the numbers do not add up to 22. Table 3.7 shows that using physical paper or text files with a web browser was a typical combination. From our interviews, I saw that participants often wrote down login information for important accounts on a piece of paper while using the browser to save login information for less important accounts. Our participants described a variety of rationales behind their choice of password managers. In this section, I first describe our findings related to each type of password management tool. Then, I describe themes that appear across multiple types of password management tools.

| Type | # | Examples |
|---|---|---|
| Text Files | 13 | Excel, standard text files, sticky note application, notepad on phones |
| Browsers | 20 | Chrome, Internet Explore, Safari, Firefox |
| Physical Paper | 7 | Memo, note, a sheet of paper |
| Applications | 5 | LastPass, KeePass, aWallet, Keyring, Keychain Access |
| E-mails | 2 | Gmail, Hotmail |

**Table 3.6 Password management tools that our participants used. Because most of them were using multiple tools, the sum of frequencies is more than the number of participants (N=22).**

| Tools | Browse | Paper | Apps | E-mails |
|---|---|---|---|---|
| Text File | 12 | 2 | 3 | 1 |
| Browser | | 7 | 3 | 2 |
| Paper | | | 0 | 0 |
| Applicati | | | | 1 |

**Table 3.7 Numbers of participants using each combination of password management tools. There were four participants using more than two tools.**

## Text Files

The most popular password management tool among the participants was text files stored on computers and phones. Here, I use the term "text" quite broadly to include not just standard text files, but also application that people can store texts such as Microsoft Excel files, sticky note applications, and notepad apps on smartphones. Four participants protected their text files using a password. Most of the participants stored both user IDs and passwords in those files.

***Simplicity*:** Almost all participants using text files as their password managers mentioned that simplicity was the biggest advantage. P21, who used a standard text file as her password manager, commented, "It's simple and easy to use. And I can edit it very conveniently." P5, who used an Excel file, said:

> "I think a text file is very simple. I don't need to install it [because it's already installed]. I can just copy the file from a computer to another one. It's simple and secure because the file has a password. I made the password [required to access the file] very difficult."

***Additional Protection*:** Similar to P5, many participants had additional security layers to protect their text-based files. Five participants used Excel files to manage their account information, and four of these had passwords on that file. Some participants only stored partial information about their accounts. For example, P5 reported that she often wrote down just half of her passwords:

> "For some passwords, I don't write down whole passwords. For instance, for a password, I'm using birthday [as a part of my passwords], and I only write down the month."

Similarly, some participants wrote down only user IDs or passwords to improve security, in case their text file was compromised. For example, P12 said:

> "I usually use a standard password. [So, I don't have to write it down.] But, user ID changes and the file has user IDs. […] I have four email addresses and use them for different websites."

P2 reported that she hid her text file in several ways: "I'm using a different extension for my text file. Also, I make it a hidden file." Instead of using a .txt file extension, she used .app, which stands for an application file on MacOS. She also added a hidden flag on the file so that the file was not visible in the graphical file system.

## Browsers

Modern web browsers offer features for storing user names and passwords, and auto-filling this information on the appropriate web sites. This functionality was widely used among my participants. Except for two participants (one using LastPass and one using a physical note to manage all login information), all of my participants

stored some of their account information in their web browser. Eighteen mentioned that these accounts were not important ones; however, there were seven participants who stored login information for their e-mail services.

***Convenience***: The majority of participants using browsers mentioned that convenience was the main reason why they save their login information there. As a result, login information for frequently accessed accounts is likely to be stored in web browsers. P12, who primarily used an Excel file to manage her passwords, said:

> "For these accounts that I access very frequently, I don't want to refer back to my file. Because, when I'm working, every minute counts."

Another participant mentioned that he stored passwords even when he remembered them, "I remember most of my passwords. But, I store the passwords in a browser to save my time."

***Importance of Accounts***: Many participants choose whether to save passwords in their browsers based on the perceived importance of a given account. P2 said, "I store passwords for the accounts that do not have my credit card information." Similarly, P11 said:

> "These are less important accounts that I don't know whether I'll access them again. I don't want to use my standard passwords [that I use for many of my accounts] because, if someone gets my passwords [from the website], potentially, he can access many of my accounts. But, I don't want to memorize the passwords or don't want to take time to write the passwords in my file. So, I create kind of random passwords and save it [in a browser]."

Interestingly, seven participants reported that they store login information for their email accounts (Gmail and Hotmail) in their browser. They commented that these accounts were important for them; however, they decided to store the login information because they accessed these accounts very frequently. One of the seven participants (P3) also reported that he stored login information for his online banking accounts in his browser. P3 explained that he did have login password on his laptop, and that he always carried his laptop with him. Thus, although he was worried his laptop being stolen, the perceived convenience of storing login information in the browser outweighed its risk.

## Physical Paper

Seven of my 22 participants used physical paper to store some of their passwords. Four participants used notepads to write down login information including service names, user IDs and passwords. They used the same notepads also to write down other information that they want to keep, such as dates and times of appointments, to-do lists, and something they found interesting. The other three participants used sheets of paper to manage their login information. In these cases, the participants wrote down login information on the sheets of paper. Only one participant (P3) used a small piece of paper, such as sticky notes, to write down passwords. P3 said:

> "I write down a password for [CMU's participant recruitment website] and tape it on my display. Because it's not an important account, I don't care. And I check the website pretty regularly."

***Securing memos***: All the participants using physical paper mentioned that they were concerned that an undesired person might gain access to their memos. Five of the seven participants reported that they always carried the memo with them. P7 noted, "I'm carrying my note with me. It's always in my backpack and it's always with me. So, I think it's safe." Furthermore, P22 explained that having the physical note gave him sense of security. He said:

> "I think it's the best way to have easy access to it. It's more personal. You can hide it. I think the feeling of touching it gives you sense of security much more than using something else."

The other two participants stored their memos in secure places. P12 noted, "I keep this [sheet of paper] on my desk, but I always lock my office when I leave." P20 said, "I hide this sheet of paper in a drawer. It has a lock and my office has a lock too."

***Memos as a backup***: The participants using physical memos also reported that they were not using random passwords even though they did not have to memorize the passwords. Rather than keeping login information on physical paper, they memorized passwords and wrote down passwords primarily as a backup. P22 commented, "I still need a sort of psychological security that, OK, I don't forget it." P9 also said, "I memorized most passwords, but, sometimes, I have to double

check." In these cases, the participants actually memorized their login information and used memo as a tool to retrieve their login information just in case they forgot it.

Some participants used physical paper as a backup for their digital files. P7 said, "I'm using a text file [on my computer] to manage my passwords. But, I also write them down on a piece of paper as a backup."

## Password Management Applications

Five participants reported that they used password management applications. The applications were LastPass (for PC), KeePass (for PC), aWallet (for Android phone), Keyring (for Linux), and Keychain Access (for MacOS). Among these participants, one participant (P6) used LastPass as his only tool for managing passwords. Other participants supplemented these applications with physical paper or text-based files to manage a small fraction of their accounts. Three participants described that there were no clear distinctions between the accounts managed by applications and ones managed by text-based files. One participant (P15) said, "The passwords in my Excel file are old ones. When I moved from the Excel file to KeePass, I didn't copy all of them because I don't access some of them." Another participant (P8) reported:

> "I'm managing my personal accounts using a text file. But, for the work related accounts, I want more security. So, I save the passwords in Keychain rather than the text file. Keychain requires a password to access. So, it's more secure."

One interesting finding was only one participant used randomly generated passwords, despite the fact that all of the applications (except Keyring) support password generation feature. P14 explained:

> "I don't like automatically generated passwords because all systems fail. If passwords are automatically generated, they are not my standard passwords. I would forget or cannot figure out the passwords [when they are lost]. If they are my standard passwords, I can try some variations of them [to figure out the passwords]."

This comment implies that participants do not want to completely rely on applications in managing their password, and want some sort of backup in cases where the application failed.

### Email

Two participants used email accounts to manage their passwords. One of them (P14) used Hotmail, and had one message that contained her user IDs and passwords for multiple services. She explained that when she created a new account, she simply replied to the message and added the new user ID and its password. She also had additional protection on her message. She said:

> "I actually have whole bunch of junk emails in the folder. This is the one that has my passwords. Its title seems like a spam. I'm trying to protect my passwords."

She also mentioned:

> "I'm using email [to manage my login information] because it's easy to access. I can access it anywhere."

Another participant (P16) used Gmail to manage his accounts. He described that whenever he created a new account, he sent an email to himself and then assigned a specific tag to the email with his login information, to make it easier to find these emails. He also had additional protection. He said:

> "I use service names as titles [of my emails containing login information], but I slightly modify them. Finding them is a little bit difficult."

He also mentioned that there were cases where he himself could not find the emails. He explained:

> "There were a couple of cases where I couldn't find emails [with login information]. I didn't use the accounts for a while and forgot the keywords to find them."

Interestingly, this challenge in re-finding one's passwords is not limited to email. In another case, one participant mentioned that he once hid a piece of paper with his passwords and later forgot where he hid it. In short, if users have information in multiple places, there is a risk of them forgetting where they stored the information.

## 3.2.4 Creating and Changing Passwords

In the first part of our interviews, I also asked how our participants created and changed passwords. I expected that our participants would use rather secure

passwords because they did not have to memorize password when using password management tools. However, the majority of participants reported that they created their passwords by simply combining words and numbers.

## Creating Passwords

Seventeen participants reported that they usually created their passwords using their own schemes, such as combining some information related to them (e.g., family members' names, pets' names, birthdays, street addresses, and phone numbers). P5 described:

> "I use birthdays. I have a mother, a son, many birthdays that I can use. In China, I have a lunar calendar. So, I can use [birthdays in] lunar calendar too [to create passwords]. I combine names and birthdays to create my passwords."

Three participants reported that they were reusing a small number of passwords for several accounts. P13 said, "I have one base password. I use it for many things." P19 commented:

> "I have three passwords. The easiest one is just numbers. The second one is characters plus numbers. The third one is the most complex one. It has long numbers and characters with symbols. I use the most complex one for my banks and other services related to finance. For the easiest one, I mostly use it for unimportant accounts. […] I also use the simplest one for Facebook and Gmail. I want to use an easy password because I type it frequently."

While memorability is well-known as an issue in passwords before, ease of typing is not as well-explored. Ease of typing also becomes more important if users do not want to store the passwords in browsers and have to access the account frequently.

Three participants reported that they created random passwords and let password management tools to save them. One participant was using the Chrome web browser and a physical note as his password management tools. Two participants were using password management applications (KeePass, and LastPass). P6 mentioned:

> "I like LastPass because it generates own passwords every time. So, getting one password isn't going [to let] someone get into all of my accounts."

I also asked whether he was comfortable with completely relying on LastPass. He answered, "I have a backup of the data. And nothing bad happens so far. So, I'm OK with it."

***Interference***: When websites have specific policies about password composition, those who manually created passwords added minimum modifications on their own password schemes to satisfy the requirements. However, these slight modifications made it difficult for them to remember the passwords, in part due to interference effects with other similar passwords. P21 said, "Because I have so many accounts and so many related passwords. I couldn't remember all of them." P14 also reported:

> "Some passwords require me to include at least one capital. That is one reason why I have to use password managers because my standard password does not include capitals. I typically forget which letter was capitalized. Now, I have to use password managers."

### Changing Passwords

Changing passwords periodically is recommended as a good practice in managing passwords. It is sometimes enforced in critical services, such as online banking. Seventeen participants noted that they had at least one account that required periodically updated passwords. However, letting users update passwords without enforcement is challenging. Among the 22 participants, 12 participants reported that they updated their password only when enforced. Three participants said that they never changed passwords. P11 said:

> "I only change my passwords when required. Creating passwords that are secure and easy to remember is difficult."

In contrast, seven participants changed passwords for important accounts regularly. P14 using an Excel file to manager her login information reported:

> "Sometimes, I go through these accounts. And, if I find some passwords are too old, I change them. I write when I created these passwords."

## 3.2.5 Sharing Accounts

In the second part of my interviews, I asked how my participants shared their accounts to deal with tasks in both personal and work contexts. Past work has looked

at how people use passwords, but very little work has examined how people share passwords [54,90].

Kaye conducted a survey consisting of 122 participants investigating how people share passwords. He reported that people shared their passwords with family members, friends and colleagues [59]. Although sharing passwords is not a good practice from security perspective, many people share their password for a variety of reasons in many different ways. As such, I wanted to explore the range of these sharing behaviors with our participants. Furthermore, I go one step further, compared to the existing works, to obtain deeper understandings sharing behaviors by revealing difficulties, concerns, and strategies that our participants had taking advantage of face-to-face interview. In this section, I describe my finds around sharing behaviors.

## Sharing Passwords

All participants except one reported that they shared passwords in the past. They shared their passwords with a small number of people, such as family members, friends and co-workers. Analyses of our participants' responses revealed that there were three different types of sharing behaviors: temporary access, repeated access and shared accounts.

***Temporary access***: There were cases where people wanted to give someone else temporary access to their accounts. In our interviews, I found that the most common case was a person asking a friend to access their accounts to check some information (e.g., email or Facebook). Eight participants reported that they either shared their passwords and asked somebody to access their accounts, or someone else asked them to do the same. Four of them said that they shared passwords at least once a month. P9 explained:

> "When I didn't have access to my email, I need information in my email. So, I called my friend and asked him to do it."

P9 did not use smartphones; thus, he did not have access to his email when he was away from his computer.

A slightly different form of temporary sharing is to let someone else use an account after the account owner has logged in. Twelve participants reported that they had

logged in and let another use their accounts or vice versa. For personal uses, this typically happened in the context of e-commerce. P20 reported, "It was my husband's Eddie Bauer account. I didn't want to set up a different account, and just used his account."

***Repeated access***: Eight participants reported that they shared passwords for repeated access. For most of the cases, they asked other people (or they were asked by other people) to help deal with tasks using their (or others') accounts.

In the context of personal use, P1 said that he shared his password for his school account with his parents to let them pay tuition. P18 reported that her daughter was sharing a password for the daughter's bank account with her. She commented, "I asked her to share the password. I want to check if there are any problems."

Interestingly, I found that sharing passwords for repeated access occurred more frequently in work contexts. P4 said:

> "I know some of my wife's accounts. She is teaching courses [at a university], and I'm helping her. To deal with things, I need to access her accounts."

Similarly, P14 helps take care of grant applications at a university. She reported:

> "I know a lot of passwords for the system. Just as a practical matter of being able to do the work, they have to assign me a special account. But they don't have time to do it. So, the way they manage it is to let me have all the user IDs and passwords to go into their accounts."

***Shared accounts***: I also found that people sometimes created shared accounts rather than sharing accesses to existing accounts. Sixteen participants reported that they have at least one account shared among multiple people. For instance, our participants shared bank accounts, Netflix accounts, accounts for paying utility bills, and Google accounts.

P12 described:

> "We have some shared accounts in Dropbox. When I work as a team, someone has to keep files consistent. In these cases, we shared a Dropbox account. Sharing one account is easier to manage for us [than configuring shared folder in

> Dropbox]. We have this type of account for all the projects I'm working on. After I finish projects, we just leave the accounts."

There are some accounts shared by a large number of people. P5 reported:

> "In my child's kindergarten, parents share one Gmail account. 20 to 30 people are sharing it. And we use the account to upload and download some files. […] We are using year and a class name as a password [for the account]."

I also found that our participants used very simple passwords for shared accounts. Because they knew that these accounts were shared before putting any data in them, these accounts were less likely to have sensitive information. Consequently, people used very simple passwords to make sharing passwords easier.

## Difficulties and Concerns

When participants described their experiences of sharing passwords, I asked them whether they had any difficulties and concerns when they shared passwords.

Eight participants described that sharing passwords was uncomfortable for them. P9 noted, "I don't really want to know their passwords. It's too personal." P10 also commented, "Sharing password really makes me uncomfortable." P16 said:

> "I don't like sharing passwords usually. […] Even if it's with somebody I trust, I don't want to share [passwords]. I'm using the same password for a long time. So, revealing one password potentially allows the person to access other accounts too."

Some participants touched on difficulties in sharing passwords. P4 mentioned, "When telling [their] passwords, people forget details of passwords, such as upper case vs. lower case." Similarly, P22 commented, "When I told my password to my girl friend [to let her access my account], she complained that my password was too complicated!" P14 commented:

> "I'm sharing many passwords [to access others' accounts]. But, they have to change the passwords periodically. So, whenever they change passwords, we have to share the password again. But, sometimes, they forget to tell me new passwords."

I also found that our participants adopted strategies to make sharing secure. Seven out of the eight participants described that they told their passwords to others (or vise versa) verbally over the phone. P11 commented:

> "I usually avoid sending my passwords via email or SMS because, if it's written, they can check it afterwards."

In most cases participants said that they trusted the persons whom they shared passwords with; however participants sometimes changed their passwords after sharing it. P8 said:

> "I changed my password after sharing it [with my friend] because I felt uncomfortable. Just to make sure that he can't access to my account later."

P7 observed that he had difficulty in changing a password:

> "I told my password for Instagram to my girlfriend. I wanted to show some photos. But, she went through all photos including very private ones. […] Then, I decided to change the passwords. I usually create my passwords using something related to me. And I thought that, wait, she knows this, she knows that. I worried that she may be able to guess my passwords."

Interestingly, P7 also reported that, despite experiencing this incident, he was still sharing some of his passwords with his friends.

Overall, the fact that our participants still share passwords despite these difficulties and concerns indicates that sharing passwords is necessary in many cases to deal with tasks efficiently in practice.

### 3.2.6 Features of Password Managers

In the last part of my interviews, I investigated how users perceived existing features of password management applications, as well as potential new features. Towards this end, I asked participants to evaluate 15 features that password management applications could support (Table 3.8). These features were generated based on an analysis of features in these applications as well as brainstorming by our research group members.

| Category | Feature | Description | Rating (median) | Top 5 Features |
|---|---|---|---|---|
| Password | Password Generation | Users can automatically generate a random sequence of letters, numbers, and symbols by clicking a button. | 2 | 6 |
| | Password Update | Users can configure some of their accounts' passwords to be periodically (e.g., every 60 days) updated automatically. | 3 | 3 |
| Login | Auto-fill | A password manager automatically fills password fields when users open pre-configured web pages | 4 | 3 |
| | Auto-login | When users click an account in a password manager, it opens its login page in a web browser and logs into the account on behalf of the users. | 4 | 4 |
| | Dedicated Browser | Users can login to their accounts using a web view implemented in the password manager without any browser extension or plug-ins to complete important tasks (e.g., online banking) | 4 | 11 |
| Security Lock | Master Password Lock | Users can lock password managers using a master password. | 5 | 15 |
| | Location Lock | Users can lock/unlock password managers based on devices' locations. | 3 | 2 |
| | Phone Lock | Users can lock/unlock password managers based on whether they have their phone nearby. | 3 | 3 |
| Management | Account Categorization | Users can organize their accounts into multiple categories (e.g., personal/work, or low/medium/high importance). | 4 | 6 |
| | Synchronization | Users can configure their account information to be synchronized among multiple devices. | 4 | 7 |
| | Data Backup | Users can configure password managers to take backups of their account information periodically to network storages (e.g., Dropbox). | 5 | 11 |
| | Information Management | Users can store additional information (e.g., insurance information, drivers' license information, address and/or credit card information) in password managers. | 4 | 8 |
| Detection | Notifications | Users receive notification on their phone when someone accesses their accounts using password managers. Users can configure which accounts send notifications. | 5 | 17 |
| Sharing | Temporal Sharing | Users can give another person who is using the same password manager one-time access permission to their accounts without revealing passwords. | 4 | 7 |
| | Repeated Sharing | Users can give another person who is using the same password manager access permission to their accounts without revealing passwords until they decide not to share the accounts anymore. | 3 | 2 |

**Table 3.8 Features that password management applications could support and users' evaluation of the features.**

I described these features one by one and asked participants to rate them using a 5-point Likert scale (1 being least important and 5 being most important). After participants rated all features, I asked participants to choose their five most important features, to compensate for any potential biases in ratings (such as giving 5 to all features). Furthermore, I asked participants to provide qualitative feedback about these features. Table 3.8 shows the descriptions of features, the median ratings, and the frequencies of being chosen as one of the top five features.

One very interesting result, which matches my findings of how people used password manager applications, was that users did not prefer *automatically generated passwords*. Participants mentioned that they understood that randomly generated passwords were more secure than manually generated passwords, and that they did not have to remember the passwords if they were using password management applications. However, they still mentioned that they wanted to have control over their passwords. P11, who saved some of his password in a browser, said:

> "I know that I don't have to memorize passwords [when using a password management application]. So, using randomly generated passwords shouldn't be a problem. But, I still feel a little bit uncomfortable [using randomly generated passwords]. I guess I'm worrying that there might be some cases where I can't use the password manager, but, still, I have to access my accounts. If I create passwords, I may be able to recover the passwords even if I don't remember them exactly."

P4 directly mentioned control: "I want to have control on my accounts. So, I don't like automatically generated passwords."

The most straightforward way to address these concerns is to let password managers have easy and reliable ways to recover users' account information when they fail, such as making a remote backup periodically (which was also a feature preferred by participants) or providing a mobile phone version that users can use to retrieve information from the remote backups.

Another interesting finding in this data is that our participants highly valued the *notification* feature, something that is not supported by existing password managers. P3 commented:

> "I'm constantly checking my bank accounts to see if there is something strange. […] I do want to have that feature for my bank account and for other accounts too."

When I asked about password management tools, almost all participants mentioned that they were concerned about cases where someone who obtains access to the applications would access to all accounts. A notification feature would let users detect suspicious activities and potentially mitigate damages (e.g., remote wipe login information to prevent further accesses).

Another feature preferred by participants and not supported by existing password management applications was a *dedicated browser*. In this feature, I assumed that a password management application has its own web view integrated with the application, and that the web view does not use extension or plug-ins that can be installed in browser applications. P11 commented:

> "I'm not sure what plug-ins and what extensions are running in my browser. In many cases, I just install plug-ins when webpages ask me to install one. Then, it becomes too complex to manage them. If there is a browser that I can use without worrying about them, that would be useful."

Finally, participants showed interests in the *temporary sharing* feature. As described in Section 4.3, our participants have to share their passwords in many different cases although they are concerning sharing their passwords. Thus, it would be natural for participants to prefer this feature. P6 commented:

> "I like this feature because it obfuscates my passwords even if I need to share my accounts. I don't have to change my password after sharing accounts."

Also, P14 described:

> "This feature helps us a lot. We often need to use somebody's account to do the work. Because I never used such feature, I'm not quite sure, but it seems useful."

Although there would be technical challenges in implementing this feature, letting people share accounts in a safer way would be more beneficial rather than prohibiting sharing accounts.

### 3.2.7 Issues around Password Managers

In the analyses of our data, I found several common themes in their behaviors surrounding their password management tools. In this section, I discuss these themes and their implications for password management application design.

#### Control and the Necessity of Password Managers

My participants indicated that they wanted to have control over many aspects of their accounts. One example I have already described is strong preferences on notifications of when an account was used. Another example that participants reported was that they tried to memorize their passwords even if they stored the passwords in password management applications. Perhaps the most surprising behavior I saw was that participants preferred creating passwords using their own schemes rather than using password generators provided by password management applications.

At the same time, people also observed that they needed password management tools. P20 noted, "It's not benefit. It's necessity." P21 echoed a similar sentiment:

> "The number of passwords that I have to manage increases every month. […] I can't handle them without [password management] tools. It's simply impossible to memorize all passwords."

P7, who used a text file, Chrome, and a notepad, observed:

> "I used to have the same password for all accounts, pretty much everything. But, I got hacked, and they got into different websites. So, I started changing my passwords for every different website. […] [If you use password management tools] you don't have to have exact same password for everything. You don't have to remember passwords."

Furthermore, three participants reported that they always went back to password management tools to check login information except for a few accounts that they used very frequently (e.g., email accounts or Facebook). P15, who used KeePass, said:

> "The only password that I don't use a password manager is the password to log into the password manager. There are some accounts that I access frequently and

> memorize passwords for. I generated passwords by myself for these accounts. But, occasionally, I forget the passwords. Then, I go back to password managers to retrieve the passwords."

As such, it may be useful to examine more ways of offering people a greater sense of control over their accounts, to facilitate the adoption of password managers. One possibility is to develop systems that let users access login information even when they do not have direct access to the applications, such as storing login information on their smartphones, or implementing a one-time master password that lets users ask somebody that they trust to check login information of a specified account without revealing those of other accounts.

## Password Managers as Backups

One theme that appeared across multiple types of password management tools was using password management tools as *backups*. This is typically true for physical paper. This usage could be because referring back to pieces of paper takes time and people try to memorize login information to save time. Nine participants using text-based files to manage their login information also commented that they were using the files as back up and did not refer back to their password management tools in most cases.

Even a participant (P4) who stored all of his account information including his online bank information in a browser explained:

> "I'm probably accurate [in remembering passwords] at 90% of the time. But, in the 10% I'm inaccurate. So, if there is a thing that makes sure that I'm not falling in the crack, it would be beneficial."

Similar to using password management tools as backups, five participants commented that the biggest benefit of using password management tools was that they knew one place where they could find all login information. P10 said:

> "I started using the sticky note [application] to manage my passwords, mainly because I started getting too many passwords that I couldn't memorize. And I wanted to make sure that I had one place I can reference back to."

Because participants regard password management tools as backups or a last resort, providing reliable backup and recovery becomes important. Existing password

management applications do have backup features; however, it is also important to provide straightforward way of recovering all login information from backups.

### Detection

Existing password management applications are focusing on *prevention*. Most of them provide random password generator that would prevent attacks directly targeting passwords, such as dictionary attacks or educated guess attacks. Some applications allow users to store URLs of websites to prevent phishing attacks. However, none of the password management applications focus on *detection*.

When discussing password management tools, almost all participants commented that they were concerning someone may access their login information stored in their password management tools. P5 said, "My only concern in using a password manager is, someone may have access to it." I believe that adding notification features will mitigate these concerns. P21 commented:

> "I like Gmail. It tells me when someone access to my account from a different IP address. It helps me find what's going on. It makes me feel comfortable."

As such, a detection feature could give users a stronger sense of having control, and makes them feel comfortable with using password management applications. Furthermore, providing notification features, which cannot be supported by physical paper or text files, could increase the benefit of using password management applications to attract more users and let them manage their login information in safer ways.

## 3.2.8 Limitations

Although I believe that this paper provides new kinds of insights about the range of behaviors in using password management tools, there are some limitations. The first limitation is that I focused on participants who were already using password management tools. As such, I have rich qualitative data about the way our participants used password management tools and difficulties in using them. However, it would be also important to investigate people who are not using password management tools to understand why they are not using them.

A related limitation is that I do not know how many people currently use various password management tools, and how well our population matches this distribution. As I have emphasized, our goal in this paper was not to quantitatively assess password management tools, but rather to qualitatively understand the range of behaviors surrounding password management tools.

Another limitation is that I relied on self-reports. In our interviews, I asked participants to bring their password management tools and to refer them when answering questions directly related to the tools. However, some of my questions, such as questions related to sharing, were based on self-report. Thus, participants' responses could be inaccurate.

Finally, my evaluation of password management features was limited by not having a working prototype. Although I explained the features in a consistent way, participants' interpretations of these features could vary. Although I still believe that the results of our interview provided useful insights for development of password management applications, further investigation with working prototypes would be necessary to fully understanding users' responses to these features.

### 3.2.9 Summary

In this section, I investigated how people manage their passwords with various tools through semi-structured interviews consisting of 22 participants. Because of relatively small number of participants, I do not claim that our finding can be generalized for general population. However, I still believe that our analyses of the interviews provided many insightful qualitative findings about the participants' behaviors and perceptions of password management tools.

- Participants showed strong preference to having control over their account information in many aspects when using password management tools.
- Most participants reported that the tools were essentially backup measures.
- The biggest advantage of having password management tools is that they had one place to refer back when they lost passwords.
- They still wanted to use passwords composed based on their own schemes (e.g., combining words and numbers related to them) rather

than using randomly generated passwords to have control over their passwords.

- Participants shared passwords as well as account in many different occasions such as temporal accesses, repeated accesses, and shared accounts.
- Our participants showed strong interest in a notification feature, which is not supported by existing password management applications.

## 3.3 Summary of Chapter

In this chapter, I described the results two studies investigating how people used and managed their account information including passwords. These results provided useful insights in designing new user authentication framework. The insights includes:

- People logged into their accounts in limited number of contexts (such as location and computers used).
- Despite high demand for password management tools, their adoption was limited
- People strongly preferred to have full control on their accounts even though they appreciated some automation that password management tools could provide (such as password auto-fill feature)
- People were reluctant to use randomly generate passwords worrying the cases where password management tools failed and they could not access their accounts
- To strengthen the control over accounts, people preferred to receive notifications when their accounts were accessed

# 4. Context-Aware Scalable Authentication

The investigations described in the previous chapter revealed that there were opportunities in developing a better password management framework. Participants also expressed a strong desire for high availability when discussing password management tools.

Mobile devices such as smartphones are a promising platform for password management applications. However, this approach leads to a new problem, namely protecting the mobile device from unauthorized access. As such, user authentication to these devices needs to be secure to protect credentials while also minimizing any burden.

The results in the first study also showed that people logged into their accounts in a limited number of contexts. I argue that this finding, coupled with the commoditization of sensor technologies, offers new opportunities for both simplifying and strengthening authentication. This insight is the basis for a probabilistic user authentication model that we call *context-aware scalable authentication*, or CASA. In this chapter, I report on iterative investigations of the model through a series of three user studies[2].

[2] The results reported in this chapter appeared in [49].

## 4.1 Combining Passive and Active Factors

CASA embodies two concepts. First, cheap digital sensors combined with models of people and places can yield multiple *passive* factors about users' identities. For my specific context, I define a factor as any data that provides information about users' identities. Passive factors are those that can be acquired without explicit interaction from end-users (e.g., users' locations or durations since last login). In contrast, active factors require explicit interactions (e.g., entering a PIN or scanning fingerprints).

Second, CASA is based on the idea that this passive multi-factor data can be used to modulate the strength of active authentication needed to achieve a given level of security. For example, with CASA, I want to enable quick and easy active factors in situations where passive factors indicate a high probability the user is a legitimate user (for instance, being located in home or work where only the user and a small number of trusted people can access). Conversely, I want active factors to be tough and reliable in situations where the passive factors indicate a low probability (such as being located in an unfamiliar place).

In this approach, CASA breaks the current underlying assumptions about authentication, by making authentication easier or harder based on passive factors rather than making it uniformly hard for all cases. I argue that today's authentication systems are designed to ensure security in extreme cases; consequently, they overlook common, mundane and ultimately average case scenarios that characterize most user authentications. Some people have argued that this conventional approach of always having more security actually leads to less compliance and less security overall (see for example, [19,88]). In particular, Norman argues that "[t]he more secure you make something, the less secure it becomes. Why? Because when security gets in the way, sensible, well-meaning, dedicated people develop hacks and workarounds that defeat the security"[17]. Norman's predictions appear to be well founded in statistics about mobile phone PINs usage. A survey in 2007 found that 61% of people had no PIN on their phones [11].

CASA targets this large population of users who do not secure their devices by attempting to derive solutions that offer them a more appropriate perceived balance between usability and security. By exploring solutions that provide easy access in commonplace everyday situations, such as whilst a user is at home, but require more

secure authentication in less common scenarios, CASA points the way towards how to lower the overall burden of having user authentication on mobile devices to increase the compliance rate.

## 4.2 Model for Active Factor Selection

In this section, I introduce my probabilistic framework for choosing an active factor given passive factors. My approach is to use a Naïve Bayes classifier to combine multiple factors, calculating a "risk assessment" value to determine the appropriate level of active authentication required given passive factors.

Most existing user authentication schemes can be considered binary classifiers, classifying a person as a legitimate user ($\hat{u} = 1$) or not ($\hat{u} = -1$). We can also model these schemes probabilistically as shown in Eq. (1) where $\hat{u}$ denotes the prediction (i.e., the result of the user authentication), $P(u = 1|s)$ denotes the probability the requester is the legitimate given the observation *s*, $P(u = -1|s)$ denotes the probability the person is not the legitimate user given the observation *s*, and $\alpha$ denotes the degree to which user authentication is conservative. The $\alpha$ parameter can be set based on one's comfort level with expected costs of false accepts and false rejects.

$$\hat{u} = \begin{cases} 1, & \alpha P(u = 1|s) > P(u = -1|s) \\ -1, & \alpha P(u = 1|s) \leq P(u = -1|s) \end{cases} \quad (1)$$

For instance, for PIN-based authentication, if the system observes that a requester enters the correct PIN, the probability that the requester is legitimate is much higher than the probability he is not. Thus, the system predicts $\hat{u} = 1$ and authenticates the user. Conversely, the system predicts the opposite if the requester enters a wrong PIN.

Many current authentication schemes focus on a single factor that has large differences between the probability distributions of $P(u = -1|s)$ and $P(u = 1|s)$ across the range of values of *s*. In contrast, CASA combines multiple factors that may or may not have as pronounced of a difference between the probability distributions of $P(u = -1|s)$ and $P(u = 1|s)$, but taken together offer benefits over a single factor approach.

In Eq. (2), I show the underlying probabilistic model of multi-factor authenticators such as CASA. Again, $u$ denotes whether a user is legitimate ($u$=1) or not ($u$=-1), and $s_i$ denotes the observation value for the $i$-th factor.

$$\hat{u} = \begin{cases} 1, \alpha P(u=1|s_1,\dots,s_n) > P(u=-1|s_1,\dots,s_n) \\ -1, \alpha P(u=1|s_1,\dots,s_n) \leq P(u=-1|s_1,\dots,s_n) \end{cases} \quad (2)$$

Eq. (2) can be reformulated into Eq. (3) using the sign function, which extracts the sign (positive or negative) of a real number.

$$\hat{u} = \text{sign}\left(\log \frac{\alpha P(u=1|s_1,\dots,s_n)}{P(u=-1|s_1,\dots,s_n)}\right) \quad (3)$$

Using Bayes' theorem, $\mathrm{P(u|s_1, s_2, \dots, s_n)}$ can be reformulated into Eq. (4). Eq. (4) has the term, $\mathrm{P(s_1, s_2, \dots, s_n|u)}$, that depends on all the factors simultaneously. In practice, estimating this term is challenging because the number of possible combinations of $\mathrm{(s_1, s_2, \dots, s_n)}$ increases exponentially when the number of signals increases. Therefore I simplify Eq. (4) as Eq. (5) by assuming conditional independence between each identifier. This is a standard transformation in building Naïve Bayes classifiers. This simplification allows us to deal with each signal separately. In Eq. (5), $P(u)$ denotes a prior probability of how likely a person is a legitimate user (or not) in general. $P(u)$ will be canceled in the following reformulations.

$$P(u|s_1, s_2, \dots, s_n) = \frac{P(s_1, s_2, \dots, s_n|u)P(u)}{P(s_1, s_2, \dots, s_n)} \quad (4)$$

$$= \frac{\prod_{i=1}^{n} P(s_i|u)\, P(u)}{P(s_1, s_2, \dots, s_n)} \quad (5)$$

Finally, by substituting $\mathrm{P(u|s_1, s_2, \dots, s_n)}$ in Eq. (3) with Eq. (5), we obtain a Naïve Bayes classifier (Eq. (6)). Intuitively, the parameter in the sign function increases with the probability that a requester is legitimate and vice versa.

$$\hat{u} = \text{sign}\left[\log\left(\alpha \frac{P(u=1)}{P(u=-1)}\right) + \sum_{i=1}^{n} \log \frac{P(s_i|u=1)}{P(s_i|u=-1)}\right] \quad (6)$$

Note that because each factor might not be conditionally independent, Eq. (6) may have approximation errors compared to Eq. (3). However, in practice, we believe the errors will be limited because we can choose largely independent factors (e.g. voice and PIN). Further, in Eq. (6), I can discuss each factor independently by estimating

$P(s_i|u = 1)/P(s_i|u = -1)$. Thus, I believe the benefit of the independence assumption outweighs its drawbacks.

## 4.3 Selecting an Active Factor

CASA uses this probabilistic model to select an active factor that provides enough evidence to authenticate a user, given a set of passive factors. The model allows us to compare the strength of the evidence using the terms in the sign function in Eq. (6).

I describe one example here to illustrate how we can utilize the framework in choosing active factors. Let's assume I want to choose an active identifier *S* that provides as much evidence when a user is at a café as compared to the user typing her correct PIN at her home. Assuming that location is the only passive factor, the condition that S should satisfy can be written as Eq. (7). The first term in Eq. (6) is canceled. $P_{s,1}(1)$ denotes the probability that the active factor *S* indicates that a person is the legitimate user when a person is actually a legitimate user. $P_{s,-1}(1)$ denotes the same when a person is not the legitimate user. $P_{L,1}(l)$ (or $P_{L,-1}(l)$) denotes the probability the person is at the location *l* when she is the legitimate user (or not). *H* and *C* denote *home* and *café* respectively.

$$\log\frac{P_{S,1}(1)}{P_{S,-1}(1)} + \log\frac{P_{L,1}(C)}{P_{L,-1}(C)} \geq \log\frac{P_{PIN,1}(1)}{P_{PIN,-1}(1)} + \log\frac{P_{L,1}(H)}{P_{L,-1}(H)} \tag{7}$$

Eq. (7) can be rewritten as Eq. (8), which quantifies the security criteria that an active factor *S* should satisfy to have the same level of security as the legitimate user typing her PIN at home, given that the active factor *S* authenticates the person at a café.

$$\begin{aligned}\log\frac{P_{S,1}(1)}{P_{S,-1}(1)} &\geq \log\frac{P_{PIN,1}(1)}{P_{PIN,-1}(1)} + \log\frac{P_{L,1}(H)}{P_{L,-1}(H)} - \log\frac{P_{L,1}(C)}{P_{L,-1}(C)} \\ &= \log\frac{P_{PIN,1}(1)}{P_{PIN,-1}(1)} + \log\frac{P_{L,1}(H)}{P_{L,1}(C)}\frac{P_{L,-1}(C)}{P_{L,-1}(H)}\end{aligned} \tag{8}$$

A legitimate user is more likely to be at her home than to be at café. Thus, $P_{L,1}(H)/P_{L,1}(C) > 1$. In contrast, someone else is much more likely to be at the café than to be at the user's home, i.e., $P_{L,-1}(C)/P_{L,-1}(H) \gg 1$. Thus, the second term on the right side is positive. Therefore, Eq. (8) indicates that the active factor should provide greater confidence than a standard PIN.

Furthermore, Eq. (8) offers a quantitative guideline for the strength of *S* given the user's location. My model can also include other passive factors, such as sensor data, time since last login, or number of times logged in at given places. I describe another example of selecting an active factor in my second field study.

## 4.4 Empirical Evaluations

To assess the feasibility of CASA, I conducted three different empirical evaluations. In my first evaluation, I investigated the potential of using location as a passive factor. Past work suggests that people spent most of their time in a few locations [20,46]. However, there is little empirical data on how frequently people used their smart phones at these locations. I collected this information to evaluate the usefulness of location information for CASA.

In my second study, I conducted a one-week field study of a prototype with 32 participants. This prototype modulated active factors based on their locations. This study helped me understand how well my ideas might work in practice, as well as to obtain feedback from participants.

In my third study, I iterated on both the system design and the study design based on the results of the second study. I conducted a 10-day field study with 18 participants. This prototype took into account location as well as whether the participants used their computers nearby recently.

### 4.4.1 Study 1: Mobility pattern analysis

In this study, I investigated people's mobility patterns along with their phone usage patterns, to evaluate the effectiveness of location information as a passive factor. I recruited multiple Android phone users through Craigslist and e-mails. Participants were asked to install our logging app from the Android Market. Participants were enrolled in a raffle for $50 Amazon gift cards as compensation. Over five months, I collected data from 128 participants. In this analysis, I focused on 36 participants with at least seven days of logs.

#### Data Collection

The app sampled location every three minutes regardless of whether participants were interacting with their phones. Location was obtained through standard

Android APIs using Wi-Fi and cell tower information. The standard API also provided the expected error for each location estimate. I discarded location data when the expected errors were greater than 200 meters. The app also logged the smartphone's running processes every 30 seconds when the smartphone was *not* in sleep mode. The timestamps of these logs let us infer when participants used their phones.

I analyzed location traces from 36 participants. The data collection periods varied from seven days to 140 days. The median length of the data collection was 26.5 days. I divided the latitude and longitude space into discrete 0.002 × 0.002 latitude/longitude grids (each cell was approximately 200 × 200 meters in/near North America) as previously done in [30]. The particular choice of discretization was based on practical considerations balancing the accuracy of Android's positioning system with granularity of the analysis.

## Identifying Phone Activation

To track phone use, our app ran a low-level thread that logged active processes every 30 seconds. When the phone was in sleep mode, the thread was automatically paused. Thus, by examining the timestamps of log entries, the phone state could be determined.

Theoretically, intervals between log entries that exceed 30 seconds signified a phone activation event after being in sleep mode once. However, initial trials of this log analysis identified two common sources of error. The first issue was the low priority of the logging thread leading to fluctuations in the sequentially logged times — variations typically in the region of 5 seconds. To deal with this, I considered valid differences between log time stamps to be in the range 30-35 seconds. The second issue was phone activations caused by push notifications (e.g. email arrival). We adopted a conservative approach to mitigate false positives relating to this issue. Essentially, phone activation events were counted only when there were two successive log timestamps after observing at least a 35 second gap. This filtered out short phone activations due to push notifications because the phone would quickly go back to sleep mode after an automatic activation. A consequence of these manipulations was that a certain proportion of valid user activations (e.g. very brief glances and interactions) would not be counted. However, despite this cost, I believe

that these manipulations ensured the validity of the study by counting only real user activations of their phones.

## Mobility Pattern Analysis

I identified 55840 phone activation events in our dataset. Participants activated their phones 27.4 times a day on average (SD=19.7). Table 4.1 shows the distribution of time spent and logins at the places where participants spent most of their time. I first calculated each participant's top five places based on the amount of time spent using location data alone (see the two columns under "Time"). Then, for each participant, I calculated the number of phone activations at each of these places using location data and process data (see the two columns under "Activations").

| Place | Time | | Activations | |
|---|---|---|---|---|
| | Mean [%] | SD [%] | Mean [%] | SD [%] |
| 1 (Home) | 38.9 | 20.2 | 31.9 | 15.6 |
| 2 (Workplace) | 18.7 | 12.6 | 28.9 | 18.1 |
| 3 | 9.9 | 8.4 | 18.5 | 13.7 |
| 4 | 5.5 | 4.8 | 10.8 | 8.5 |
| 5 | 4.3 | 4.7 | 5.2 | 4.7 |
| Other places | 22.6 | 13.1 | 4.5 | 4.6 |

**Table 4.1 The distribution of the time spent and the phone activation events at the places where participants spent most of their time. Place 1 to 5 denote the places where participants spent most time (1) to fifth most time (5).**

The results indicate that people spent 57.8% of their time at two locations, presumably homes and workplaces. This result is aligned with past work investigating people's mobility patterns e.g., [46]. However, before conducting this study, it was unclear to us how often people would use their smartphones at home and work, since there would be other devices with network connectivity and larger displays (e.g., desktop and/or laptop computers) at these locations. Nevertheless, my results showed that these top two places accounted for 60.8% of the total phone activation events on average (SD=14.5%). This data indicates that people activate their phones more frequently at their homes and workplaces than at other places.

This result provides supporting evidence that people exhibit strong patterns in where they use their smartphones, suggesting that location could be a very useful passive factor. This result also indicates that I can positively impact both usability and security if we adjust the active factor based on location data coupled with a very trivial model (home, workplace and other places). Again, this approach makes the

assumption that a person's home and workplace have reasonably good physical security, and that there are relatively few trusted people that can access those locations.

### 4.4.2 Study 2: Evaluation of CASA prototype

In this field study, I developed and deployed a prototype using CASA framework for Android smartphones. This prototype dynamically selected active factors based on participants' location (i.e., whether they are at home, workplace, or some other places). In this study, I investigated users' reactions to dynamically changing active factors. I also collected empirical data to estimate how much effort our participants could reduce in user authentication when using our prototype. These data help us to understand the design space opened by the concept of CASA, and to improve the prototype for the next design iteration to make it better fit users' needs.

#### Participants

I recruited 32 participants using a participant recruitment website at Carnegie Mellon University. Their age ranged from 18 to 40 years old with a mean age of 24. My participants consisted of 26 students, five full-employed and one non-employed. Twenty-three out of 32 participants were living with others in their homes. I compensated participants $40 for their participation in the study.

Participants were assigned to one of two conditions based on whether they used any security lock on their phones prior to this study. Nineteen participants *not* using a security lock (i.e., PIN or Android Pattern Lock) were assigned to the *PIN* condition. Thirteen participants already using a security lock were assigned to the *password* condition. None of the participants were using passwords to secure their phones. In essence, participants used the same authentication they already used at home and work, and had stronger active authentication at other places.

#### Procedure

In the first session, I installed our prototype on participants' Android phones. I asked participants in the PIN condition to choose a PIN. For participants in the password condition, I asked them to choose a password in addition to a PIN.

During the study period, when the participants turned on their phone displays, our prototype selected an active factor based on the participant's location (home, work,

and other) and condition (the PIN or password condition) (see Table 4.2). In the explanation, I explicitly defined "work" as a room or building where the participants spent most of time except home. For instance, for students, "work" means their offices or campus buildings. After participants authenticated, the prototype asked the participants to answer if they were at home, work, or other place (Figure 4.2 (c)). The answers were used to train the location classifier implemented in the prototype. This classifier is trained on the fly during the study using the ground truth. After one week, I had the second session where I asked participants to complete a post-survey, and conducted a follow up interview that lasted about 15 minutes.

## Prototype with Active Factor Selection

My prototype used location as a passive factor and selected an active factor from three options: no active factor, a PIN, and a password. First, I describe how I can use CASA in selecting active factors, using the password condition as an example. The participants in the password condition were using PIN or Android Pattern Lock to secure their phones prior to this study. Thus, for the participants in the password condition, I selected active factors so that they would provide the same level of evidence as typing a PIN at workplace, where risks are higher than home, but still lower than other places.

Because location is the only passive factor in our prototype, Eq. (6) can be simplified to Eq. (9) and (10). These equations denote the conditions that active factors should satisfy to provide no less evidence than being at home (Eq. (9)) or at a place other than home and workplace (Eq. (10)), where W, H and $O_i$ denotes *workplace*, *home,* and a place *other* than home and workplace respectively. Note that $O_i$ does not denote the aggregation of places other than home and workplace, but it denotes a single place. $f(l_1, l_2)$ and $g(S)$ are defined as shown in Eq. (11). Intuitively, $\log(f(l_1, l_2))$ means the likelihood that a person is a legitimate user when she is at $l_2$ compared to when she is at $l_1$. If it is less likely, $\log(f(l_1, l_2))$ becomes positive. Then, the evidence provided by the active factor (the term on the left side) should be greater than that of PIN. If it is more likely, $\log(f(l_1, l_2))$ becomes negative. Then, the active factor could be weaker than PIN. As $|\log(f(l_1, l_2))|$ increases, the user's location provides stronger evidence towards authentication. $g(S)$ denotes how strongly an active factor S indicates users' identities.

$$\log g(S) \geq \log g(PIN) + \log(f(W,H)) \quad (9)$$

$$\log g(S) \geq \log g(PIN) + \log(f(W,O_i)) \quad (10)$$

$$f(l_1,l_2) = \frac{P_{L,1}(l_1)}{P_{L,1}(l_2)}\frac{P_{L,-1}(l_2)}{P_{L,-1}(l_1)}, g(S) = \frac{P_{S,1}(1)}{P_{S,-1}(1)} \quad (11)$$

I estimated $g(S)$ based on the entropy of four-digits PINs (~9 bits) and passwords (~18 bits) according to the estimations by NIST [25]. Assuming that the authentication system allows three trials and that a legitimate user always types a PIN and a password correctly, then we have $P_{PIN,1}(1)=1$, $P_{PIN,-1}(1)=3/2^9$, $P_{Pwd,1}(1)=1$, $P_{Pwd,-1}(1)=3/2^{18}$ and $P_{None,1}(1) = P_{None,-1}(1) = 1$. Thus, $g(PIN) = 2^9/3$, $g(Pwd) = 2^{18}/3$ and $g(None) = 1$.

To calculate $\log(f(W,H))$ and $\log(f(W,O_i))$ accurately, further empirical data collection is needed. However, because our primary purpose in this study was to investigate participants' responses to our concept rather than applying CASA precisely, we approximated these values. I approximate the values in a way so that $|\log(f(l_1,l_2))|$ becomes smaller to avoid overestimating the strength of the evidence provided by location information. I discuss the data collection issue more in the discussion section.

For $P_{L,1}(H)$ and $P_{L,1}(W)$, I used 0.389 and 0.187 based on the results in the first study. For $P_{L,1}(O_i)$, I used 0.099, which was the highest probability among the places other than home and workplace in the first study (Table 4.1). When $P_{L,1}(O_i)$ becomes higher, the CASA model estimates the evidence provided by being at home and the workplace to be lower. Thus, I used the highest value for all $O_i$ to be conservative. Additionally, I assumed that $P_{L,-1}(l)$ was proportional to the number of people who can physically come into the location. Because I do not have empirical data about $P_{L,-1}(l)$, in the followings, I describe how the assumption affected the active factor selection.

Figure 4.1 is a graphical representation of Eq (9) and (10). The diagonal plots show the right sides of the Eq. (9) and (10), and the horizontal lines denotes $logg(S)$ for each factor (i.e., S=None, PIN or password). Intuitively, the X-axis denotes how many people can access certain locations (home for the red plot and other places for

the green plot) compared to the number of people who can access workplaces. The Y-axis denotes confidence about users' identities. When I only consider active factors, the confidences are not relevant to numbers of people who can access certain locations. Thus, the blue plots become horizontal. In contrast, when I consider locations as indicators of users' identities, the confidences become dependent on the likelihood. Thus, the plots become diagonal as shown by the red and green plots.

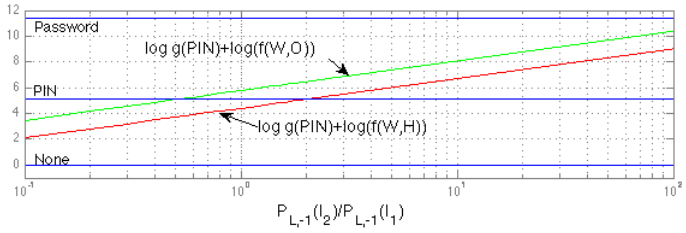



**Figure 4.1 Graphical representations of Eq. (9) and (10). Horizontal line denotes log(g(s)) for each S (None, PIN, or password).**

In Figure 4.1, satisfying Eq. (9) is equivalent to the condition that the red diagonal plot is below one of the horizontal lines in a given x-axis range of focus. Here, I make the first assumption about home:

> The number of people who can access home is less than that of workplace and more than 1/10 of that of workplace.

This assumption makes the range of focus to be $[10^{-1},10^{0}]$. The lower diagonal plot in the range is below the horizontal lines representing *PIN* and *Password*. Because I want to choose a most usable active factor that satisfies Eq. (9), I select PIN as an active factor for the case where a device is at home. Similarly, I make the second assumption about other places:

> The number of people who can access *other places* is more than that of *workplace* and less than 100 times of that of workplace.

This assumption makes the range of focus to be $[10^{0},10^{2}]$. The green diagonal plot representing Eq. (10) in the range is below the horizontal lines representing *Password*. Therefore, I select passwords as an active factor for the case where a device is at other places. I do not have emprical data to support these assumptions; however, these assumptions are safe to make considering they are considerably loose.

Additionally, my choice of active factors (Table 4.2) made active authentication more secure than that used by our participants prior to the study. My prototype required the same active factors as they used prior to this study at their homes and workplaces, and required more secure active factors at other places. Thus, I made the authentication more secure for our participants, compared to pre-study levels.

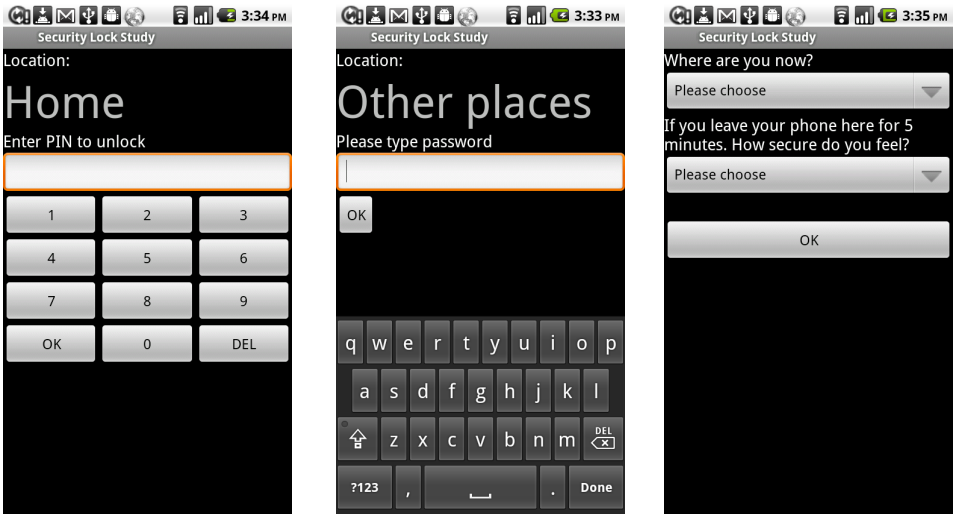
| Condition | Home | Workplace | Other places |
|---|---|---|---|
| PIN | None | None | PIN |
| Password | PIN | PIN | Password |

**Table 4.2 Active factors required at different locations in the second study. The prototype required the same active factors as participants were using at their homes and workplaces while required stronger active factors at other places.**

## User Interfaces

My prototype estimated the smartphone's location every 150 seconds using standard Android APIs (which uses WiFi access points and cell tower information). The positioning system returns latitude, longitude, and estimated error. I discarded the location if the error was greater than 200 meters.

When a participant turned on her display, our prototype took the latest location information and classified the location as home, workplace, or other, using a 5-nearest neighbors classifier. To minimize misclassifications, especially in areas where ground truth data is sparse, the classifier considered ground truth within a 100 meter radius. The prototype then requested an active authentication according to participants' locations and the experimental conditions (Table 4.2). After participants completed the active authentication, the prototype asked participants to confirm their semantic location (home, workplace or others) to use as additional ground truth data for the 5-nearest-neighbor classifier (Figure 4.2).




(a) PIN (b) Password (c) Questionnaire

**Figure 4.2 Prototype screenshot. Based on users' locations and the conditions (see Table 2), the prototype either skips authentication, (a) requests PIN, or (b) requests password. After authentication, the prototype showed a questionnaire to obtain ground truth of locations.**

## Results

### *Location Classification*

My prototype asked for the ground truth of locations after each authentication (Figure 4.2 (c)) and trained the 5-nearest neighbor classifier using all the ground truth collected up to the classification. The classification accuracy was 92%. Most of the misclassifications happened due to our location sampling rate. It would be therefore be possible to improve the classification accuracy by increasing the sampling rate when the accelerometers on the mobile device detect that it is moving.

### *User Authentication*

Participants activated their phones 33.8 times a day on average. Figure 3 shows the distribution of phone activations per day. The black and gray bars represent participants in the PIN condition and in the password condition respectively. The participants in the PIN and the password condition activated their phone a mean of 23.9 times and 47.3 times a day respectively. The difference between the means was statistically significant with Welch's t-test ($t(14)=2.78$, $p<0.05$). This result might be because those who use their phones more frequently are more likely to have sensitive data on their phone. Table 4.3 shows that participants in the PIN condition activated their phones 68% of the time at home or work, and participants in the

password condition did the same 55% of the time. This indicates that they mostly activated their phones at homes or workplaces.

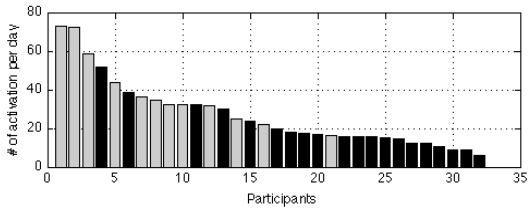



**Figure 4.3 The number of phone activations per day. Gray and black bars denote participants in the PIN-password condition and none-PIN condition respectively.**

| Condition | Home | Workplace | Other places |
|---|---|---|---|
| PIN | 13.1 (1.4) | 2.5 (0.4) | 8.1 (1.1) |
| Password | 24.5 (3.2) | 7.1 (1.0) | 15.7 (2.0) |

**Table 4.3 The means of the phone activation frequency per day at each location. The numbers in parentheses denote standard deviations. Both the PIN and the password condition activated phones more than 50% of time at homes or workplaces.**

One possible concern with CASA's approach is that users may be more likely to forget their PINs or passwords because they are used less frequently. However, in this study, we found that participants still typed PINs and/or passwords frequently enough to retain them. As shown in Table 4.3, participants typed PINs 8.1 times a day on average in the PIN condition. Similarly, in the password condition, participants typed PINs more than 31.6 times a day and typed passwords 15.7 times a day on average. Furthermore, I found that the participants typed correct PINs 96.5% of the time, out of 1034 total authentications using PINs. Additionally, no participant typed the wrong PIN three times successively. For passwords, there were two cases out of 1193 authentications using passwords where participants typed wrong passwords three times successively. However, in both cases they retrieved passwords in the next authentication. These data indicate that, although the frequency of typing PINs and passwords decreased, the memorability of PINs and passwords remained high.

### *Participants' Receptiveness*

In a post-survey, I asked participants about their perceptions of our prototype using a 5-point Likert scale (higher scores being more positive). Participants in both conditions were very receptive to our prototype. Below, the number in the parentheses denotes the median of ratings.

Participants in the PIN condition reported that not requiring a PIN at home and work while requiring a PIN at other places was useful (4) and very easy to understand (5). They also reported that they felt our prototype was secure (4) compared to not having any security lock on their phone. They were neutral (3.5) to using our prototype if it were available on their phones.

Similarly, participants in the password condition reported that requiring a PIN at home or work while requiring a password was neither useful nor useless (3) and easy to understand (4). They also reported that they felt the prototype was more secure (4), as easy to use as requiring a PIN at all the places (4). However, they were neutral (3) to using our prototype.

I further asked the participants in the password condition about the configuration that we used for the PIN condition. (i.e., not requiring PINs at homes or workplaces and requiring PINs at other places). The participants reported that the configuration would be easy to use (4) and as secure as a requiring a PIN at all places (3.5), and they somewhat agreed (4) that they would use the system if it were available on their phones.

As these results indicate, participants thought our prototype useful. Although participants were neutral to using our prototype on average, our participants rated the none-PIN configuration as easier to use than the security lock that they used prior to my study, and more or equally secure to the security lock. I further iterated on the system design in our third field study to make it fit better to users' needs.

## 4.5 Security Analysis

In this section, I discuss the security implications of CASA with respect to our results from the second study. Through this discussion, I identify potential security risks and possible improvements to the system that I tested in the third study. Table 4.4 divides possible attackers into four groups based on whether they have information about their target (*informed* or *uninformed* attackers), and whether the attackers have knowledge about CASA as well as information security in general (*novice* or *expert* attackers).

### 4.5.1 Uninformed Novices and Experts

An example scenario where an uninformed novice might attack CASA is the case where a legitimate user loses her phone outside of home or work, and some stranger picks it up. In this case, CASA is almost as secure as a system that requires a PIN all the time. The only situation where it would be weaker is if the user loses her phone right next to her home or work, or if an attacker breaks into a person's home or work (again, I assume these places have reasonably good physical security; thus, it is less likely). An expert attacker could try to activate the phone at different places to find the user's home or workplace, in hopes of putting CASA into its simpler mode of authentication. However, CASA can also be configured to always require a PIN after a certain number of trials, making this kind of attack infeasible.

| | | Knowledge about target users | |
|---|---|---|---|
| | | Uninformed | Informed |
| Technical expertise | Novice | Uninformed Novice | Informed Novice |
| | Expert | Uninformed Expert | Informed Expert |

**Table 4.4 We categorized attackers into a 2×2 table based on knowledge about target users and technical expertise.**

### 4.5.2 Informed Novice

Informed novices would be people who know a lot about an individual but not a great deal of technical expertise. Those who living with users, such as family members, could be informed novices. However, my survey results showed that the participants trusted people who they are living with. Furthermore, even if the phone is protected by PIN, existing work has reported that people frequently share their PIN among these people [34]. Thus, even if CASA does not work well when the family members are not trustable, it does not increase the risk significantly. Alternatively, the system can allow people to configure where it does and does not require PIN.

Friends or co-workers could be informed novices as well. They could visit users' homes or workplaces to access the users' phones. The system in study 2 did not have protection for this threat model. In designing the prototype used in the second study, I assumed that homes and workplaces would have reasonable level of physical security and that people would put more weight on ease of access than security in these places. The results of the second user study suggest that these assumptions hold for homes but not for workplaces. Thus, we improved the system to mitigate the risk at workplaces and tested in the field study 3.

### 4.5.3 Informed Experts

Informed experts are the most capable attackers against CASA, who can dedicate time and resources to breaking the system. In practice, this group likely represents a small and exceptional case, one that goes outside of the average case that I am focused on, but also a case that CASA should offer some kind of protection against. At the same time, too much emphasis on security may lead people to not have any security, as exemplified by the number of people not using security locks on their phones.

Given the difficulties of defending against a dedicated attacker, the relative rarity of attacks, plus the goal of balancing security and usability, I opted to focus less on prevention mechanisms and more on detection, making it easier for phone owners to see if others were making use of their devices. I implemented and evaluated one possible detection mechanism in the field study 3 in the form of a notification mechanism indicating when and where the smartphone was used.

## 4.6 CASA Design Iteration

Based on the previous field study and security analysis, I iterated on the system design of our prototype, to make it more acceptable for users and secure against some of the potential attacks. The results from the previous field study clearly showed that participants with security locks on their phone found requiring passwords on mobile devices too high a burden. Thus, we configured our system to require a PIN at places outside of one's home.

Participants also showed concerns about not requiring user authentication at workplaces. Our survey results showed that, while 68% of participants strongly trusted people who could access their home, only 18% strongly trusted those who could access their workplace. This implies that the approximation that I made in the active factor selection in the system design was not accurate enough because it did not take the level of trust into account. Thus, in this iteration, I assumed that being at workplaces does not provide enough evidence to change the active factor from PIN to none. Instead, we added one more passive factor at workplaces, having smartphones check whether users were using their laptop (or desktop) computers nearby. If the computers were being used, the probability that users were near their smartphones was quite high, assuming that the computers required passwords to be

accessed. Thus, the smartphones required no active factor when the computers were used nearby recently.

My smartphone prototype communicated via Bluetooth with an application installed on the users' computers every 60 seconds to check the last time the keyboard or mouse was used. If the users interacted with their computer within the past 180 seconds, the smartphone prototype did not require a PIN. This modification could address the cases where, for instance, the users leave their smartphones at their workplace unattended.

In addition to improving prevention mechanisms (i.e., user authentication), I also added a notification mechanism. When a user's smartphone was activated, a popup message would appear on their computers. Clicking on the message would show the geo-location of the smartphone on a map. The message disappears after five seconds automatically. Although this approach does not prevent illegitimate accesses, it makes detection easier and could prevent further access to sensitive data.

Finally, I modified CASA to always require a PIN when someone turned on the phones' display more than five times without typing a PIN, to prevent attackers from trying to systematically find a user's home or workplace.

### 4.6.1 Study 3: Iterative Evaluation

In addition to using the modified prototype described in the previous section, I also modified our study protocol to improve ecological validity. In the second study, participants were always asked their locations after they activated their phones to train the location classifier. However, for a real version of CASA, this data collection should happen only for a short period when users start using the system. Furthermore, in field study 2, some participants commented that answering where they were located every time they typed in a PIN was tedious. This may have negatively impact participants' perception of the prototype. To address this problem, I divided the study period into a training period and an evaluation period.

On the first day, I had the first session where I explained our system and installed my prototype on the participants' Android phones and laptops. The first five days was a training period where the system asked for participants' semantic locations (home, workplace or other places) as well as requiring PINs if necessary, as I did in the previous study. After the training period, I had five to nine days of the evaluation

period, where the system stopped asking the questions. After the evaluation period, I had the second session, where I asked participants to fill our survey and conducted interviews for 20 minutes.

I recruited 18 participants using the university's participant recruitment website. I recruited participants who had Android phones and laptops with Bluetooth, which they used at their workplaces. None had participated in previous studies. Their age ranged from 21 to 40 years old with a mean age of 26.3. The participants consisted of 12 students and six fully-employed. Sixteen of 18 participants were living with others in their homes. Seven were using security locks on their Android phones prior to our study. All the participants used passwords to log into their computers and to unlock screensaver on their computers. I compensated participants $60 for their participation.

## Results

For the basic analyses, I found results similar to those from field study 2. Thus, I will describe the analysis of the modified parts.

### *Logins at Workplaces*

Figure 4.4 shows the distribution of the average numbers of the phone activations at workplaces per day for each user. The black parts denote the number of cases where their computers were active and the phones did not require a PIN to be activated, and the gray parts denote the number of cases where the phones required PIN. On average, the participants activated their phone 5.5 times a day at their workplaces. Out of the 5.5 times, the phones did not require PINs 2.9 times, while they did 2.6 times.

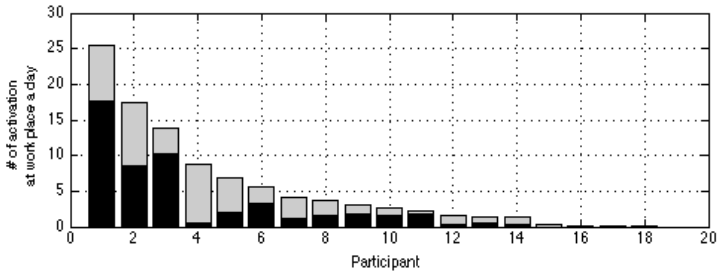



**Figure 4.4 The average numbers of activations at workplace per day. The black portion denotes the cases where the system did not require a PIN because the user's computer was being used. The gray portion denotes the cases where the system required a PIN.**

### *User Perceptions*

In the post-survey, I asked our participants to rate the three features in the system both in Likert scales and freeform responses. In the followings, the numbers in parentheses denote medians of Likert scale responses where 1 denotes very negative and 5 denotes very positive.

The participants were generally positive about our system. They reported that changing the authentication method based on location was useful (4.5), and that changing it based on computer usage was useful (4) as well. They also answered that changing authentication method based on location and computer usage was easy to understand (5 and 4.5 respectively). They also reported that, compared to always requiring a PIN to unlock the phone, it was not less secure not to require a PIN at home (4) or based on computer usage (3.5). Furthermore, they reported that they wanted to use the features if the features were available on their phones (4 for the location-based modification and 3.5 for the computer-usage-based modification). These ratings were better than the ratings in the field study 2, which implies that the modifications in this iteration made the system a better fit to users' needs.

One interesting finding was that participants who were not using a security lock prior to this study also reported that they wanted to use CASA (4). P17 commented, "It is annoying to use security locks all the time, but whereas if I had such a system which requires pin only at unsecure places its usefulness adds more value when compared to the annoyance caused by it. So, I will definitely use it." The ratings and this comment indicate the that current all-or-nothing approach where users have to either enable security lock all the time or disable the lock completely, does not meet users' needs well. Furthermore, the users indicated that they would adopt the system

even if it would undermine usability so long as they see an appropriate balance between usability and security.

Participants were also very positive about the notification feature. They rated the feature useful (4), and stated that they wanted to use the notification feature if it was available on their phones and computers (4). P13 commented, "I think it's a great way to help with privacy. I use both my computer and phone a lot and it would be very useful to have security." On the other hand, P2 was concerned about distractions, saying, "The notification system is very useful […]. But at the same time, if you just unlock your phone and quickly get back to work, the notifications on the screen can be annoying at times." Interestingly, despite the P2's comment, users are unlikely to see the notifications when they activate their phones by themselves. Users would be looking at their phones when they activate their phones; thus, they are less likely to see the notification on their computers' displays because it disappears after five seconds. Therefore, although there are some cases where the notifications distract users as pointed by P2, we believe that the distraction would be minimal.

## 4.7 Discussion

In this chapter, I proposed a generic framework for active factor selection based on passive factors. Then, I investigated one possible implementation in this design space by building, evaluating, and iterating on a prototype that made use of location data as the passive factor. I believe that the prototypes demonstrated the feasibility and usefulness of our framework. Investigating other points in this large design space will be beneficial in developing authentication systems that provide good security while putting minimum burden on users.

As exemplified in our iterative design process, we can start with a reasonably good system design based on the CASA framework using approximations. Then, we can iterate on the system design, improving the approximation based on data obtained in user studies.

One line of future work is to evaluate other passive factors and user models. Prior work has investigated the security of some passive factors, such as behavioral biometrics. However, the security of other passive factors is not clear, especially when malicious attackers try to impersonate legitimate users. Furthermore, in this

chapter, I used a very simple model (two passive factors modeling three locations and computer usage). The model had the benefit of being simple to implement and simple to understand. It is clearly possible to build more sophisticated models, combining more passive factors (e.g. last login time, number of times logged in at a given location). However, this approach raises new questions about how well users can understand what the system is doing, and could lead to frustration in case of poor prediction.

Lastly, I believe it is worth investigating new "good enough" forms of active authentication. For example, most active authentication schemes today are designed for high accuracy in differentiating between legitimate and illegitimate users. By leveraging multiple passive factors, it is possible to relax this constraint, requiring only "good enough" accuracy.

## 4.8 Limitations

This chapter proposed probabilistic approach in user authentication for mobile devices. The results obtained in the three iterative studies showed that the approach would improve both usability and security of the authentication. Nevertheless, this work has several limitations. For example, for each factor, CASA needs estimates of $P(s|u = -1)$, the probability that a person trying to be authenticated is not legitimate. Most active factors, such as passwords, have both theoretical and empirical estimates of this probability. In contrast, passive factors, which have not been investigated in the context of user authentication, have limited data. More investigation of passive factors is necessary to rigorously understand this space.

Another limitation of our studies is the treatment of workplace. In the prototypes, I assumed that there would be reasonable physical security at workplaces. This assumption is appropriate for many office workers, but may not be for those who do not have offices or other dedicated space at their workplaces. A possible solution for this issue is to ask users to configure places where they think they have reasonable physical security. It may also be possible to estimate the level of security of a place based on analysis from publicly available sources, such as foursquare.

## 4.9 Summary of Chapter

In this chapter, I introduced Context-Aware Scalable Authentication (CASA). CASA is a generic probabilistic model that enables the selection of appropriate active authentication factors given a set of passive authentication factors. I also developed prototypes exploring this design space, investigating the feasibility and effectiveness of the proposed framework. The results of three field studies demonstrated that the prototypes could select active authentication factors based on passive factors while balancing security and usability of user authentication. These results also provided useful insights:

- Participants were positive about adjusting the security level of a user authentication scheme based on contextual information.
- Location was a promising piece of context for adjusting the security level.
- Adjusting the security level improved usability by leading to fewer requests for user inputs (such as a PIN).
- Adjusting the security level also improved security by facilitating adoption of security locks on smartphones.
- Participants were comfortable with making user authentication less strict at trusted locations to lessen the burden.
- Participants were very positive about notification feature that informed them when their phones were unlocked.

# 5. One Security does NOT Fit All

Context-Aware Scalable Authentication demonstrated that it could reduce the number of explicit user inputs in user authentication to smartphones by utilizing contextual information captured by on board sensors. However, it did not take purposes of phone activations into account. For instance, it did not make distinction between the case where a person wanted to check weather forecast and the case where he wanted to use an email client, which typically did not require an additional authentication to access email accounts. More generally, existing access control mechanisms on smartphones (such as PIN or CASA) restrict access to nearly all functionality when the smartphones are locked regardless of the importance of each feature; then, when the smartphones are unlocked, the mechanisms allow access to all functionality on the phones except a few hard-coded exceptions such as answering incoming calls, making emergency calls, or taking photographs. In this chapter, I report on a study that investigated how well (or badly) this *all-or-nothing* locks met the access control needs of 20 smart phone and tablet users, and how receptive they might be to alternative access control policies and authentication mechanisms[3].

First, we examined how users would configure their phones if given the opportunity to make additional functionality available when the phone is in the locked state. We asked participants to identify their 20 most important applications. For each

[3] The results reported in this chapter appeared in [48].

application, we asked whether they would want some, or all, of the application's functionality to be available when the device was locked. All participants wanted at least one of their applications protected by a security lock. On average, our participants wanted roughly half of the applications available even in the locked state and half of the applications only available in the unlocked state. This means that our participants must currently opt for an access control model that is either "too hard", putting all applications behind the lock, or "too soft", using no lock at all. A device that was "just right" would allow them to lock roughly half their phone's functionality and make the other half available when the device is locked.

I also investigated solutions to the challenges users faced when trying to share their devices under an all-or-nothing access control model, which I (and others) had observed in prior work [21]. I created paper prototypes of two alternative access mechanisms that could support safer sharing: group accounts and an activity lock. Configuring a *group account* to a device enabled a device's owner to grant others access to a limited set of applications. A group-specific PIN unlocked the phone to login to the group account; alternatively, the owner could transition the phone from an unlocked state to the group account state without further authentication. Another access control mechanism to facilitate device sharing, the *activity lock,* required no configuration but was activated by the device owner before handing the device to another user. As its name implies, the activity lock restricted the available functionality of the device to that associated with a specific activity (*e.g.*, playing a game). Both of these sharing controls appealed to a significant fraction of participants. In particular, I found several parents of young children to be quite interested in enabling safe sharing of devices with their children. However, when presented in the context of devices that allowed selected applications to be made available when locked, nearly a third of participants deemed these additional sharing mechanisms unnecessary.

While knowledge-based authentication methods (*e.g.*, passwords and PINs) are most commonly used on mobile devices, proponents of biometric authentication methods have argued that these technologies may provide a faster and more convenient way to unlock a device. However, it is not clear how users react to biometrics, especially when exposed to possible false rejects and false accepts. Thus, we investigated participants' reactions to the use of biometric authentication to unlock their devices.

Participants tried five different authentication mechanisms. Three control mechanisms chosen from technologies that are ubiquitous today, *i.e.*, numeric PINs, passwords, and security questions (*a.k.a.* challenge questions). Two biometric authentication mechanisms, face recognition and a combination of voice and face recognition, were presented to participants as if they were working features, but were actually simulated using a Wizard-of-Oz approach. The researcher remotely (and discreetly) unlocked the device when the participant tried to authenticate under some conditions, but did not unlock the device when we dimmed the lights or introduced noise to illustrate the limitations of these forms of authentication to each participant. Participants were also warned that biometric authentication might falsely allow imposters who looked or sounded like them to access the device. Despite the disclosure of these limitations, and the potential privacy-invasiveness of biometric authentication, participants were surprisingly receptive to the technology as simulated.

When combined, these findings move us closer to a future in which devices require authentication less often, can be shared more safely, and offer additional choices for how to authenticate.

## 5.1 Application Classification Study

I conducted structured interviews in our lab to investigate users' perceptions about access control mechanisms. Each interview lasted 90 minutes. In the interviews, I asked for participants' opinions and preferences as well as their reasoning behind their choices. Although the lab study has its limitation in terms of the ecological validity, the interview format allowed us to investigate a wide variety of options in access control mechanisms before more costly implementations and field deployments take place.

I recruited 20 participants (9M, 11F) who owned both smart phones and tablets, using Microsoft's recruiting service to access a diverse population in the Seattle region. To ensure participants could speak to the problem of access control when sharing their phones, I required that participants live with others and had shared their smart phone at least once in the last month. Prior work (*e.g.*, [21]) suggested that sharing mobile devices was common practice among friends and/or family

members. I similarly found sharing was common and this requirement did not overly constrain our recruitment. To gather diverse opinions, I recruited both participants who currently use a security lock on their smart phone (11) and those who do not (9). Participants received a choice of Microsoft software and hardware gratuities (Max value $600 USD).

Our participants ranged in age between 23 to 54 years old (mean: 34). Participants used a variety of phone operating systems: iPhone (9), Android (8), Windows Phone (1), Symbian (1) and Palm Pre (1). Most participants had an iPad (17), with two participants having Android tablets and one a webOS tablet. Their occupations were diverse and included baristas, stay-at-home parents, engineers, wedding planers, business owners, and mechanics. None of the participants were Microsoft employees.

During the interviews, I first asked participants about how they would like to control access to their most important phone applications. Next, they tried multiple authentication methods including biometric authentication and then gave feedback on two mechanisms for limiting access while sharing: activity lock and group accounts. The study first focused on their phones, and then I repeated the same questions for their tablets. I now describe each section of the interview in more detail.

### 5.1.1 All-or-Nothing Access to Applications

I asked participants to select from their installed applications the 20 that they would be least willing to give up. I then asked participants how frequently they used each application and how often they shared it with others. Next, I asked them to place each application into one of the following three categories (see Figure 5.1):

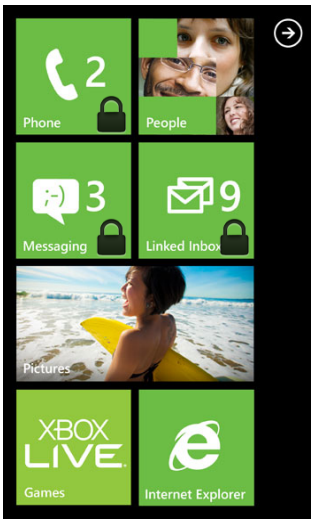
**Always Available**: Applications that would be available regardless of whether the phone was locked or unlocked.

**After Unlock**: Applications that could only be accessed when the phone was in the unlocked state.

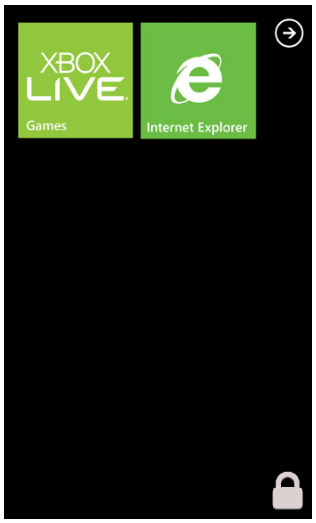
**Split:** Applications to partition such that some functionality would be accessible when the phone was locked, and other functionality would be available only when

the phone was unlocked. The example I gave of such an application was splitting the phone application into making local calls and making international calls.

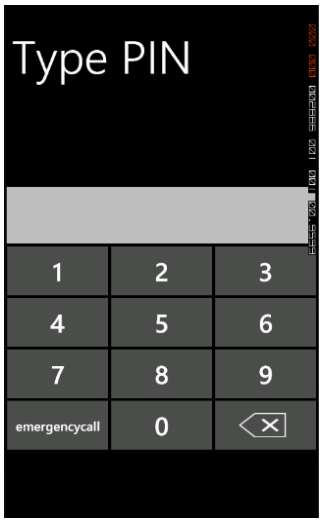

**Figure 5.1 Participants categorized their applications by whether they wanted the application *always available*, available only *after unlocking,* or if the application's functionality should be *split* between those two categories.**




**(a) Tiles with padlock icons show applications that are inaccessible when device is locked**




**(b) Inaccessible applications are hidden when device is locked**




**(c) Baseline: all applications are inaccessible and hidden**

**Figure 5.2 Prototype UI designs for navigating a phone with some applications available when the device is locked.**

After users categorized the applications, I interviewed them in more depth about their choices, and, in particular, about the applications in the Split category. I then

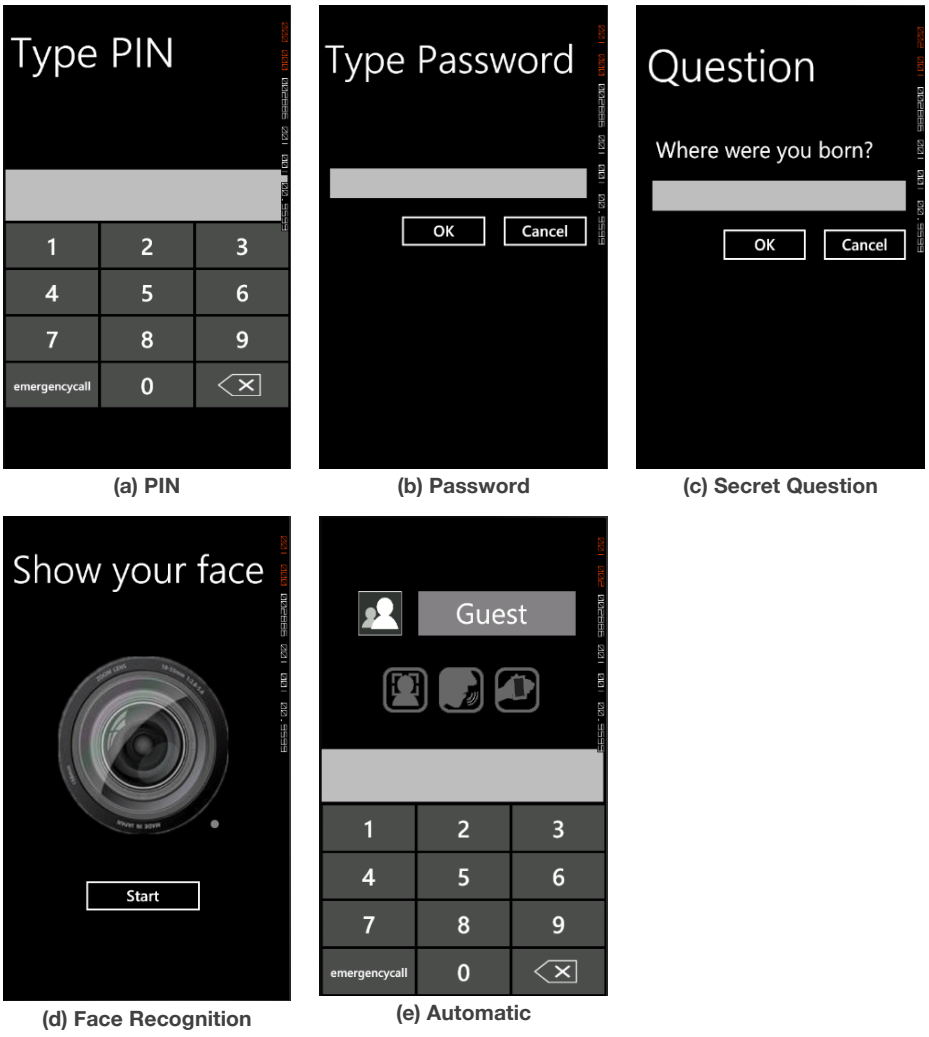
asked them how they would prefer to manage the access to their applications and the visibility of which applications are not accessible when the device is in the locked state. To illustrate the options, I used the paper prototypes as shown in Figure 5.2. I offered them the choice of showing all applications with padlock icons indicating which applications are inaccessible when the device is locked (Figure 5.2 (a)) or hiding applications that are inaccessible when the device is locked while showing all accessible applications (Figure 5.2 (b)). I also included a baseline case which is common practice on most mobile devices. In the baseline condition, none of the available applications are shown and all applications are accessible only upon authentication (Figure 5.2 (c)). I counter-balanced the presentation of the three designs to avoid the ordering effect.

### 5.1.2 Biometric Authentication Methods

To gauge participants' reactions to different forms of authentication, particularly biometric authentication, I had them try and rate five authentication mechanisms. I used a Samsung Focus Windows Phone 7 device, shown in Figure 5.3. I augmented the phone with additional Gadgeteer sensors [12] including a front camera and touch sensor on right side because there was no Windows phone that supported these sensors when I conducted the study.


**Figure 5.3 The prototype used to test the authentication methods. The casing on the rear side of the phone contained a Gadgeteer and a touch sensor.**

I tested three baseline authentication mechanisms that I expected any smart phone user would be familiar with: PINs, passwords, and secret questions (*aka* challenge question). I used a Wizard-of-Oz simulation to test two biometric authentication mechanisms: one using face recognition and one using a combination of face and voice recognition, which I called *automatic* authentication (see Figure 5.4). To describe this last mechanism, participants were told that they could use their voice, their face, or both to perform automatic authentication, so long as the biometric evidence was strong enough from whichever sensors were available.




(a) PIN (b) Password (c) Secret Question

(d) Face Recognition (e) Automatic

**Figure 5.4 The screens presented in each of the five authentication methods we tested.**

Furthermore, as long as the touch sensor on the phone sensed that they were holding their phone, the phone did not lock itself. I demonstrated that the two biometric authentication mechanisms could fail when there was noise (for voice recognition) or insufficient light (for face recognition). Participants were told to try to unlock the

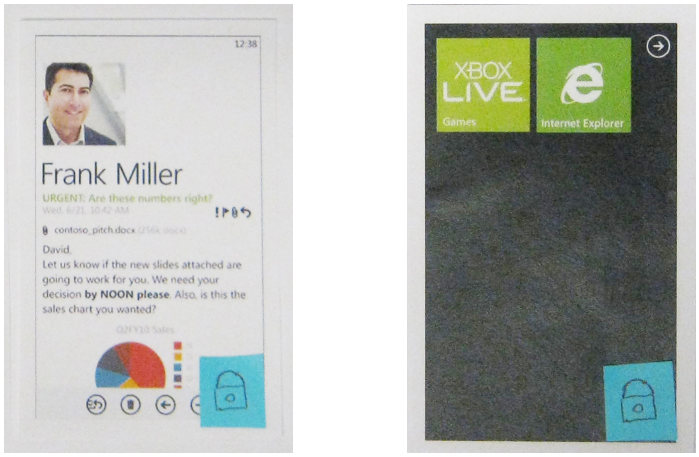
phone while I added noise and dimmed lights to confound participants' attempts to login via a biometric method. In these cases, participants were allowed to fall back to using the device PIN. For instance, the user interface in Figure 5.4 (e) shows the case in which automatic authentication failed because none of the three signals (face, voice or touch) was present or detected with sufficient accuracy.

Participants locked and unlocked the phone five times using each method and I counter-balanced the order in which they were presented using the Latin square method. When participants were "using" the two biometric methods, the researcher remotely unlocked the phone as appropriate to simulate working biometrics. Participants were oblivious to this deception.

The primary purpose of showing multiple authentication schemes was to elicit our participants' qualitative opinions about using different authentication schemes. Testing the specific user interface designs was out of scope for this study. Hence, I adopted straightforward user interfaces for the authentication mechanisms.

### 5.1.3 Limiting Access when Sharing the Device

Finally, I exposed our participants to two optional add-on mechanisms for sharing their devices without granting access to all functionality: *activity lock* and *group accounts* (see Figure 5.5). I told our participants that the activity lock restricted the phone such that the recipient would be limited to accessing only functionality available in the locked state and functionality associated with the device's current activity. I said that when an unlocked phone had its activity-lock activated (via a button press), only the current application would remain unlocked; for example, applying an activity lock when running the e-mail application would allow this application to remain accessible. Pressing the activity lock button one more time would lock the phone to a specific functionality *within* the application, such as reading a specific e-mail. Unlocking the activity lock would require the same authentication mechanism as the device lock.




(a) Activity Lock (b) Group Account

**Figure 5.5 Paper prototypes of the two proposed sharing mechanisms. The blue padlock icon in (a) indicates that the access is limited to a specific e-mail and in (b) that is limited to a specific set of applications.**

Group accounts are similar to guest accounts in desktop operating systems and to the restricted mode approaches proposed by prior work [21,93]. One or more group accounts would provide access to some of the functionality that is normally available only when the phone is unlocked. Group accounts would be accessed via a group-specific PIN or, when the phone was unlocked, it could be put into group-restricted mode without a PIN.

I asked participants to think about how they shared their phone and whether they would prefer a simple lock/unlock mechanism (*i.e.*, current state-of-the-art), an activity lock, or group accounts. The phrasing of our question encouraged a single preference, but I allowed participants to choose more than one mechanism if they asked to do so.

## 5.2 Results

In the following I present our main findings for the three parts of the study described above. To facilitate the comparison, I present results for phones and tablets together.

### 5.2.1 All-or-Nothing Access

Participants had between 12 to 103 applications on their phones (mean: 58, median: 51), and 20 to 167 applications on their tablets (mean: 72, median: 63). Out of these, each participant was asked to choose the 20 most important applications for her

phone and tablet and then categorize them into the three categories of Always Available, After Unlock, and Split. Overall, our participants categorized 378 phone applications (five participants had fewer than 20 applications, and one participant chose more than 20 applications as he did not initially include phone-feature applications, such as calling) and 399 tablet applications (three participants had fewer than 20, and four participants categorized a few more than 20).

I found a surprisingly even division between the number of applications that participants wanted to protect with the lock and the number they wanted available. Participants wanted 35% (median) of applications on their phones to be Always Available, 45% (median) of applications to be available only After Unlock, and 20% (median) of four applications to be Split such that some functionality would be always available while other functionality was protected by the lock. The allocation of phone applications to categories for each participant is shown in Table 5.1(a).

For our 20 participants all-or-nothing access to applications appears to be a poor fit for every one of them, regardless of whether they currently use or do not use security locks on their devices. Overall, our participants put at most 71% of their categorized applications into a single category. The best case, as shown in Table 1, is that of P17 who could have 15 (71%) of her applications in the category she desired for them (Always Available) by deactivating her phone's security lock. Yet, even this is not a perfect solution—she currently uses the lock on her phone to protect six of the applications she did not want always available. Thus current locking mechanisms that force users to choose between using the phone's lock for all applications or for none do not match how participants told us they wanted to manage access control to their applications.

Simply allowing users to configure whether each application is available or unavailable when their phones are locked enables users to manage the applications in Always Available and in After Unlock according to their preferences. Although this simple modification does not help the applications in Split, it makes the access control system significantly closer to what users want compared to the current all-or-nothing approach.

| Phone Lock? | Partici-pant ID | Always Available | Split | After Unlock |
|---|---|---|---|---|
| Yes | 17 | 15 (71%) | 3 (14%) | 3 (14%) |
| | 20 | 11 (55%) | 3 (15%) | 6 (30%) |
| | 18 | 9 (47%) | 4 (21%) | 6 (32%) |
| | 6 | 8 (40%) | 4 (20%) | 8 (40%) |
| | 19 | 8 (40%) | 3 (15%) | 9 (45%) |
| | 14 | 7 (35%) | 5 (25%) | 8 (40%) |
| | 3 | 7 (35%) | 0 ( 0%) | 13 (65%) |
| | 1 | 5 (25%) | 5 (25%) | 10 (50%) |
| | 12 | 5 (26%) | 5 (26%) | 9 (47%) |
| | 16 | 4 (20%) | 6 (30%) | 10 (50%) |
| | 8 | 4 (20%) | 4 (20%) | 12 (60%) |
| | *Subtotal* | 83 (38%) | 42 (19%) | 93 (43%) |
| | *Median* | 35% | 20% | 45% |
| No | 9 | 12 (60%) | 1 ( 5%) | 7 (35%) |
| | 2 | 10 (50%) | 7 (35%) | 3 (15%) |
| | 10 | 9 (39%) | 6 (26%) | 8 (35%) |
| | 5 | 9 (45%) | 3 (15%) | 8 (40%) |
| | 7 | 9 (45%) | 2 (10%) | 9 (45%) |
| | 15 | 6 (35%) | 3 (18%) | 8 (47%) |
| | 13 | 5 (25%) | 7 (35%) | 8 (40%) |
| | 4 | 5 (42%) | 2 (17%) | 5 (42%) |
| | 11 | 2 (29%) | 4 (57%) | 1 (14%) |
| | *Subtotal* | 67 (41%) | 35 (22%) | 60 (37%) |
| | *Median* | 41% | 19% | 40% |
| *Total* | | 150 (40%) | 77 (20%) | 151 (40%) |
| *Median* | | 39.5% | 20% | 40% |

**(a) Phones**

| Tablet Lock? | Partici-pant ID | Always Available | Split | After Unlock |
|---|---|---|---|---|
| Yes | 20 | 13 (59%) | 4 (18%) | 5 (23%) |
| | 6 | 9 (45%) | 2 (10%) | 9 (45%) |
| | 15 | 8 (36%) | 5 (23%) | 9 (41%) |
| | 14 | 8 (40%) | 4 (20%) | 8 (40%) |
| | 3 | 6 (29%) | 0 ( 0%) | 15 (71%) |
| | 1 | 5 (25%) | 6 (30%) | 9 (45%) |
| | *Subtotal* | 49 (39%) | 21 (17%) | 55 (44%) |
| | *Median* | 38% | 19% | 42% |
| No | 17 | 16 (80%) | 1 ( 5%) | 3 (15%) |
| | 8 | 16 (76%) | 0 ( 0%) | 5 (24%) |
| | 9 | 16 (84%) | 0 ( 0%) | 3 (16%) |
| | 2 | 13 (65%) | 5 (25%) | 2 (10%) |
| | 4 | 12 (63%) | 4 (21%) | 3 (16%) |
| | 19 | 10 (50%) | 0 ( 0%) | 10 (50%) |
| | 18 | 9 (43%) | 6 (29%) | 6 (29%) |
| | 11 | 8 (62%) | 3 (23%) | 2 (15%) |
| | 7 | 8 (40%) | 2 (10%) | 10 (50%) |
| | 10 | 7 (33%) | 4 (19%) | 10 (48%) |
| | 16 | 7 (35%) | 3 (15%) | 10 (50%) |
| | 12 | 5 (25%) | 7 (35%) | 8 (40%) |
| | 13 | 4 (20%) | 8 (40%) | 8 (40%) |
| | 5 | 4 (20%) | 0 ( 0%) | 16 (80%) |
| | *Subtotal* | 135 (49%) | 43 (16%) | 96 (35%) |
| | *Median* | 46.5% | 17% | 34.5% |
| *Total* | | 184 (46%) | 64 (16%) | 151 (38%) |
| *Median* | | 41.5% | 18.5% | 40% |

**(b) Tablets**

**Table 5.1 Participants categorized phone (a) and tablet (b) applications into applications they wanted *Always Available*, applications that should be available only after the device is unlocked (*After Unlock*) and applications they wanted to *Split* such that only some application functionality would be always available and some would be available only after the device is unlocked. Participants who currently use their device's lock (*yes* in the leftmost column) appear above those who do not. Participants are ordered based on the number of applications placed in the Always Available category.**

The allocation of tablet applications to categories is shown in Table 5.1 (b). The results are very similar to those for phones, with participants preferring to make a slightly (not significantly) larger fraction of applications Always Available. Six participants commented that they mostly kept their tablets at home, which might have made them less concerned about security and privacy on these devices.

I asked participants to explain the motivations behind their categorizations. Not surprisingly, the two main factors mentioned were privacy (for locking the application) and convenience (for making it always available).

Most participants reported that they wanted applications containing personal data to be available only After Unlock (18 participants for phones and 13 for tablets). P6 said, "These all contain my personal information, which I don't want people to see."

Conversely, several participants reported they wanted applications that did *not* contain personal data to be Always Available (13 participants for phones and 6 for tablets). P19 commented, "For always available [category], I put things that won't have direct connections to my private information."

Seven participants mentioned quick access as a reason to make phone applications always available, and ten did so for tablets. P15 said, "If you have this stuff available without unlocking it, it's just handier." P12 also said, "[Applications in always available are] stuff I want to access quickly." Furthermore, P12 had a particularly strong desire for quick access while driving. She told us: "I'm using my iPhone for navigation. But it locks. Then I have to type my password to unlock it while driving. So I disable the lock when I drive."

Some participants also reported that they wanted applications that could be used to make purchases available only After Unlock (6 participants for phones and 11 for tablets). P5 commented, "there are things where you can purchase things, which I don't want somebody to access."

## Types of Applications

I hypothesized that certain types of applications would be more likely than others to be made Always Available. I classified applications into types, using the grouping taxonomy of the iTunes application store, to examine how different types of applications might have different security/accessibility tradeoffs. Three researchers manually classified other applications (*e.g.*, applications for other phones or applications distributed using different channels) working together to resolve any disagreements. The results are presented in Table 5.2.

Supporting participants' qualitative comments about how they categorized their applications, types of applications likely to require personal information (*e.g.*, communication) were likely to be made available only After Unlock or Split, whereas those unlikely to hold personal information (*e.g.*, entertainment) were more likely to be made Always Available.

| Applications Types | Always Available | Split | After Unlock | *Type Total* |
|---|---|---|---|---|
| Utilities | 31 | 6 | 32 | 69 |
| Communications | 4 | 27 | 21 | 52 |
| Productivity | 7 | 12 | 32 | 51 |
| Photography | 14 | 12 | 10 | 36 |
| Entertainment | 18 | 4 | 11 | 33 |
| Social | 5 | 4 | 19 | 28 |
| Reference | 10 | 7 | 9 | 26 |
| Navigation | 14 | 1 | 3 | 18 |
| Games | 12 | 0 | 2 | 14 |
| Lifestyle | 4 | 4 | 5 | 13 |
| Weather | 11 | 0 | 0 | 11 |
| Travel | 4 | 0 | 2 | 6 |
| News | 5 | 0 | 0 | 5 |
| Music | 3 | 0 | 1 | 4 |
| Finance | 1 | 0 | 3 | 4 |
| Books | 1 | 0 | 0 | 1 |
| Others | 6 | 0 | 1 | 7 |
| *Total* | 150 | 77 | 151 | 378 |

**(a) Phones**

| Applications Types | Always Available | Split | After Unlock | *Type Total* |
|---|---|---|---|---|
| Entertainment | 40 | 8 | 18 | 66 |
| Productivity | 10 | 9 | 27 | 46 |
| Games | 32 | 1 | 12 | 45 |
| Utilities | 9 | 7 | 29 | 45 |
| Photography | 16 | 7 | 12 | 35 |
| Communications | 1 | 9 | 15 | 25 |
| Social | 0 | 9 | 16 | 25 |
| Reference | 12 | 5 | 5 | 22 |
| Lifestyle | 15 | 2 | 1 | 18 |
| Books | 6 | 2 | 9 | 17 |
| News | 14 | 1 | 1 | 16 |
| Travel | 8 | 0 | 2 | 10 |
| Music | 6 | 1 | 1 | 8 |
| Navigation | 7 | 0 | 0 | 7 |
| Finance | 0 | 2 | 3 | 5 |
| Weather | 2 | 0 | 0 | 2 |
| Others | 6 | 1 | 0 | 7 |
| *Total* | 184 | 64 | 151 | 399 |

**(b) Tablets**

**Table 5.2 Participants' classification of applications into Always Available, Split, and After Unlock shown by the type of application.**

Among the 378 phone applications, the most common categories were *utilities* (69, 18%), *communication* (52, 14%), and *productivity* (51, 13%). On the other hand, among the 399 tablet applications, the most frequent types were *entertainment* (66, 17%), *productivity* (46, 12%), *games* (45, 11%), and *utilities* (45, 11%). This difference suggests that our participants' tablets were primarily used for entertainment, while their phones were used for practical purposes. Participants' comments also suggest that these differences in types of installed applications may make them less conservative about sharing tablets than sharing phones. Table 5.4 shows how frequently applications on phones (Table 5.4 (a)) and tablets (Table 5.4 (b)) were shared. The distribution of the applications by sharing frequency shown in the rightmost columns indicates that applications on tablets were more frequently shared. P9 commented, "For the most of part I use this [tablet] for entertainment but I don't have any critical information saved. There may be some passwords, Pandora, YouTube, Netflix, Live Strong. But, they are not a big deal." P3 also commented, "My phone is more like my personal thing. This [tablet] is not a big deal because it's a shared device. It wouldn't affect me."

## Application Usage and Sharing Frequency

I hypothesized that a desire for convenience might cause users to make their most frequently used applications always available. Contrary to our expectations, I found that the Always Available category contained a disproportionately small number of frequently-used applications (Table 5.3). The participants reported 101 phone applications to be used most frequently (*i.e.*, more than 10 times a day). However, they wanted only 22% of their most frequently used phone applications, and 29% of their most frequently used tablet applications, to be always available. Alas, the applications participants used most also contained the most sensitive information, such as e-mail.

I also hypothesized that applications that were more frequently shared would be more likely to be made always available. I asked the participants to indicate how frequently they shared each application, using a scale with five options: *weekly*, *less than once a week*, *less than once a month*, *less than once a year*, *never*. Indeed, there does appear to be a correlation between applications that are frequently shared and those that participants wanted to be always available (Table 5.4). Applications that were shared weekly were made always available 57% (26 out of 46) of the time on phones, and 72% (78 out of 109) of the time on tablets.

| Usage Frequency | Phones | | | | Tablets | | | |
|---|---|---|---|---|---|---|---|---|
| | Always Available | Split | After Unlock | Total | Always Available | Split | After Unlock | Total |
| 5 (10+ times a day) | 22 | 33 | 46 | 101 | 10 | 9 | 15 | 34 |
| 4 | 49 | 21 | 43 | 113 | 46 | 23 | 40 | 109 |
| 3 | 37 | 12 | 27 | 76 | 47 | 16 | 43 | 106 |
| 2 | 24 | 4 | 17 | 45 | 31 | 8 | 23 | 62 |
| 1 (less than weekly) | 18 | 7 | 18 | 43 | 50 | 8 | 30 | 88 |
| Total | 150 | 77 | 151 | 378 | 184 | 64 | 151 | 399 |

**Table 5.3 Participants' classification of applications into Always Available, Split, and After Unlock shown by the usage frequency of applications. The frequency metric (5 to 1) stands for 5) more than 10 times a day, 4) one to 10 times a day, 3) more than or equal to once in three days, 2) more than or equal to once in a week, and 1) less than once a week.**

| Sharing Frequency | Phones | | | | Tablets | | | |
|---|---|---|---|---|---|---|---|---|
| | Always Available | Split | After Unlock | Total | Always Available | Split | After Unlock | Total |
| 5 (Weekly) | 26 | 13 | 7 | **46** | 78 | 12 | 19 | **109** |
| 4 | 19 | 12 | 14 | 45 | 17 | 7 | 21 | 45 |
| 3 | 25 | 9 | 20 | 54 | 15 | 3 | 10 | 28 |
| 2 | 10 | 9 | 8 | 27 | 2 | 4 | 4 | 10 |
| 1 (Never) | 70 | 34 | 102 | 206 | 72 | 37 | 97 | 206 |
| Total | 150 | 77 | 151 | 378 | 184 | 63 | 151 | 398 |

**Table 5.4 Participants' classification of applications into Always Available, Split, and After Unlock shown by the sharing frequency of applications. The frequency metric (5 to 1) stands for, 5) more than or equal to once a week, 4) less than once a week, 3) less than once a month, 2) less than once a year, and 1) never.**

## Applications in Split Category

Perhaps most interesting are the applications our participants put in the Split category when they wanted some functionality to be Always Available and other functionality available only After Unlock. As the Total row in Table 1 shows, overall, participants wanted 20% of the phone applications and 16% of tablet applications to be split in this way. I asked participants to explain which functionality should be Always Available and which functionality should be available After Unlock. They described three different ways to split applications:

**Feature sensitivity**: Protecting a particular feature or set of features was the most common reason for splitting. Five participants wanted inbound communications (*e.g.*, receiving phone calls) to be always available while outbound communications (*e.g.*, making a phone call) available only after unlocking. Six participants also said that browsing existing entries or creating new entries in an application should always be possible, while modifying or deleting existing entries should be possible only after unlock. Furthermore, for the applications involving purchasing of goods, such as App Store or coupon applications (*e.g.*, Groupon), eight participants wanted browsing information to be always available while purchasing to require unlocking the phone.

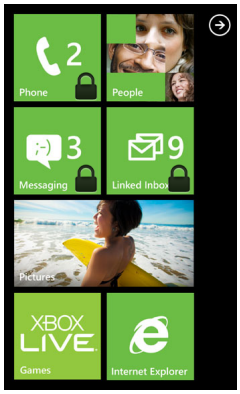
**Data sensitivity**: Protecting some of the data in an application was another reason given for splitting. For example, three participants wanted emergency contacts to be always available, but access to most contacts to require unlocking the phone. Two participants mentioned they wanted some photos, such as those of their children, to be available only after unlocking. Lastly, in the calendar application, a participant

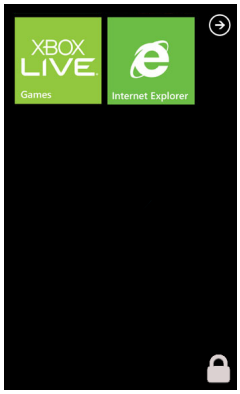
wanted business appointments to be always available, and private appointments to be available only after unlocking.

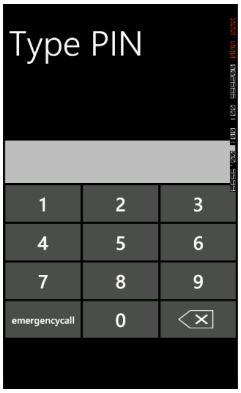
**Freshness**: Splitting between showing the most recent data and older data was the final reason described to us. For the communication applications (*e.g.*, text messaging, instant messenger and e-mail), four participants wanted new incoming messages to be always available. In contrast, they felt old messages should be available only after unlocking.

## User Interface for Unlocking Applications

Table 5.5 and Table 5.6 show participants' preferences between the prototype user interfaces for showing or hiding inaccessible applications when the phone is locked. For phones, about half of the participants preferred showing inaccessible applications (*i.e.*, those requiring the device be unlocked) using a padlock icon to indicate that they were currently inaccessible. For tablets the trend was less clear, but 40% of participants preferred showing only the applications available while the tablet was locked, hiding the locked ones until after the user authenticates.




| Device | Showing Inaccessible | Hiding Inaccessible | PIN |
|---|---|---|---|
| Phones | **11 (55%)** | 5 (25%) | 4 (20%) |
| Tablets | 6 (30%) | **8 (40%)** | 6 (30%) |

**Table 5.5 Participants' preferences among designs for navigating a locked device. Most preferred method for each device is bolded.**

| Sharing | Showing Inaccessible | Hiding Inaccessible | PIN |
|---|---|---|---|
| Frequent | 4 (20%) | **12 (60%)** | 4 (20%) |
| Infrequent | **13 (65%)** | 4 (20%) | 3 (15%) |

**Table 5.6 Participants preferred hiding inaccessible applications on frequently shared devices and showing inaccessible applications on devices infrequently shared.**

Participants' comments suggested that their choice of preferred user interface had a social explanation. Some participants did not want other users to see applications that were inaccessible to them. For example, P16 said that knowing these applications were not available to others might "piss them off". P5 said a user interface showing inaccessible applications might "tease" his daughter. In contrast, participants who preferred showing inaccessible applications with lock icons indicated it was for visibility and convenience. Participants liked the fact that they could see all applications. P1 said that the design "shows you what's there," even if it is not currently available. Another popular reason to choose this design was that it made clear when the PIN was necessary.

Given these comments, to understand if the amount a device was shared is a good predictor of which interface was preferred I categorized the devices into two groups based on sharing frequency. I categorized as *frequently shared* devices with five or more applications shared at least monthly. Other devices were categorized in *infrequently shared*. As Table 6 shows, many participants did favor hiding inaccessible applications on frequently shared devices (60%) and showing inaccessible applications on infrequently shared devices (65%).

### 5.2.2 Access when Sharing

Participants reported sharing a mean of 12% of their phone applications (median: 9%) more than once a week—primarily *photography* applications. Participants reported sharing a mean of 27% of their tablet applications (median: 25%) more than once a week—primarily *entertainment* applications. Similar to the informal and spontaneous types of sharing reported by Karlson et al. [21], our participants were more likely to mention focused, short-term sharing scenarios on their phone, such as sharing a photo, making a quick phone call, as compared to longer term-sharing scenarios on their tablet, such as watching movies or browsing the web.

At the top of many participants' security concerns were individuals with questionable judgment and frequent access to their devices—their children. Of the 11 participants with children, five referred to their children as one of the threats they wanted to protect against. Parents wanted to limit the access privileges of children because unintentional actions could cause unintended modification or deletion of data.

Furthermore, they expressed concerns that children might, accidentally or purposefully, make purchases or perform other actions that cost money.

## Activity Lock and Group Accounts Preferences

Participants were divided as to which sharing mechanism would work best for them, as illustrated in Table 7. Fourteen participants preferred to have Activity Lock (6) and/or Groups (9) on their phones over a lock alone. This included one participant who wanted both on her phone. Three participants wanted both on their tablets. For nine phones and ten tablets, participants chose group accounts. For six phones and six tablets, participants chose to have an activity lock. While I suggested that participants elicit a single preference, I allowed participants to choose more than one option if they chose to do so; the options are complementary as a phone could have both group accounts and an activity lock. On six phones and seven tablets, the owners prefer to have neither activity lock nor group accounts, if they can configure which applications are always available and which applications are available only after their devices are unlocked.

Participants who wanted the activity lock were more likely to describe ad-hoc or irregular sharing scenarios. P12 described using the activity lock "if I want to show you something, that's all I want." In contrast, those who expressed preferences for group accounts were more likely to discuss recurrent sharing scenarios.

Four participants expressed concerns configuring groups on phones would take too long. P11 said, "I just think it's interesting although I won't use it. It may be more hassle than worth… it takes too much time to configure". No participants raised the configuration time concern for tablets. P2 described his desired simple configuration in which he could tell his kids to "dial 1234 and you can use your applications".

While 70% of the participants preferred to have either the activity lock or the group accounts on their devices, 30% of the participants chose neither of the additional sharing. Most of them reported that the extra effort was not worth the benefits, assuming the feature that allows them to categorize applications into the three categories (always available, split and after unlock) was available. For example, P7 said, "these offer a lot of granularity but I don't look for a lot granularity because I have always available and not available".

### 5.2.3 Receptiveness to Biometric Authentication

Participants had some applications for which they wanted to require a lock in both their own use and when sharing. While phones and tablet devices typically use a PIN lock, biometrics is another possible mechanism for authentication. In the study, participants authenticated five times with five authentication methods: a PIN, password, secret question, face recognition, and a method I referred to as "automatic" authentication, which combined face and voice biometrics. (Recall that, unbeknownst to our participants, the biometric authentications were only simulated—our researcher triggered the unlock mechanism remotely.) I asked participants their preferred method overall, which method they felt was most convenient and which method they felt was most secure.

| Scheme | Overall | | Convenience | | Security | |
|---|---|---|---|---|---|---|
| | Phone | Tablet | Phone | Tablet | Phone | Tablet |
| PIN | 2 | 6 | 1 | 3 | 2 | 3 |
| Password | 1 | 1 | 0 | 0 | **10** | **12** |
| Secret | 0 | 0 | 0 | 1 | 0 | 0 |
| Face | 2 | 3 | 4 | 6 | 4 | 2 |
| Automatic | **15** | **10** | **15** | **10** | 4 | 3 |

**Table 5.6 Participants' preferred authentication method overall and based on convenience and security for phones and tablets.**

Table 5.6 shows three quarters of participants preferred automatic authentication for their phone, using voice and face biometrics, despite recognizing that it might not be as secure as a password or PIN. Ten participants preferred automatic authentication for tablets, whereas six preferred using a PIN. Only two had preferred a PIN for their mobile phone. Participants who preferred using PINs for their tablets often liked the simplicity PINs offer when devices are shared. P19 said, "It's shared all the time. Simplicity is important considering how often it is shared." Another reason given by P3 for preferring PIN was that he did not need high security for the tablet. He said "I chose PIN because there is no reason for it to be super duper private. My phone comes with me everywhere, but my iPad stays at home. So there is no necessity for security."

Five participants mentioned that automatic authentication would work well when they drive. For example, P4 reported that his preference was influenced by not wanting to "push buttons" to authenticate while driving.

I was also surprised that only one participant mentioned privacy as a concern in using the two biometric authentication schemes. This surprised us as privacy is often cited as a concern with biometrics [100]. Once users register their biometrics information to an authentication system, the information could be easily replicated and shared among multiple parties to track the users. Furthermore, the users cannot change their biometrics information even if they noticed that their biometrics information is leaked. Although leakage is less of a risk when the biometrics information is stored in a local device, there is no clear way for users to distinguish whether the biometrics information is stored locally or sent to a server. Perhaps few participants expressed privacy concerns because they already speak into their mobile devices and use them to take photographs. It's also possible that participants would have felt uncomfortable expressing privacy concerns to researchers, as it might imply distrust.

Although overall the results are encouraging for using biometrics for authentication, participant preferences do come with some caveats. Once again, participants were not aware that the system they preferred was not real. Participants may have believed that the researchers had put great effort into perfecting a working biometric authentication system and wanted to please the researchers by expressing a preference for that system. Also, real biometric authentications might not perform as seamlessly or inspire as much confidence as a Wizard-of-Oz simulation.

## Multi-level Authentication

I initially asked participants to categorize their applications based on whether functionality should be Always Available or only After Unlock (or Split between those two options). However, there is no reason that users must be limited to only two authentication states. After participants had tried the five different authentication methods I asked them whether they wanted to add one or more additional authentication states in addition to locked and unlocked, and, if so, what method of authentication they wanted to use for that state. Note that additional authentication states could be accessible via authentication that was weaker than what a participant preferred for unlock, but stronger than no authentication at all. Or conversely, additional states could use methods more secure than what the participant preferred for unlock.

Ten participants expressed a preference for adding another authentication level. Four of these described using biometric authentication as a weak authenticator and using PIN or password for stronger authentication. For example, for applications such as banking, e-mail or social networking, they wanted to be more protected. Perhaps not surprisingly, most participants (8 of 10) that were interested in an additional authentication level were currently using security locks on their phone.

The decision to add additional authentication levels was more popular for phones than for tablets. Only three participants were interested in additional authentication levels on tablets. One participant classified tablet levels as "Always", "Kids", and "No kids", essentially using a third authentication level in place of a group account.

## 5.3 Limitations

The results of our study, like all studies, should be interpreted with a full understanding of the limitations of our methodology. Participants were asked hypothetical questions, and their self-reported responses may not match the choices they would make in reality. Unfamiliarity with Windows phones could have also caused confusion. On the other hand, all but one participant were unfamiliar with Windows phones and so they were equally inexperienced.

For all participants, I first asked questions about their phones and then repeated questions for their tablets. I did not randomize or counterbalance. Thus, statistical differences between what participants reported for their phones and what they reported for their tablets could be the result of changes in preference that occur over the progression of the study.

While one benefit of our lab study was the ability to gather qualitative data from participants about the reasons behind the choices they made, I want to acknowledge that our participants used the authentication mechanisms only in a laboratory environment. The mechanisms worked when they could reasonably be expected to and were never exposed to security attacks. Real biometric authentication systems may not be able to perform as seamlessly. By the time such systems become available, participants' preferences may have changed based on other experiences, the reports of others, or other factors.

Finally, for some participants the 20 applications I sampled might not be representative of the full set of applications on their devices. Regardless of this limitation, this subsample alone seems sufficient to disprove the assumption that all-or-nothing access to applications meets users' needs.

## 5.4 Summary of Chapter

In this chapter, I reported a study investigating how well all-or-nothing access control on mobile phones and tables fit with users' needs. The results showed that, for our participants, all-or-nothing access to applications did a remarkably poor job of meeting their self-reported preferences. Allowing applications to be made accessible in the locked state would go a long way to meeting our participants' needs, and likely mobile users in general. The results also provided useful implications for designing access control and/or account management systems. The implications include:

- All-or-nothing access control scheme barely satisfied users' access control needs on mobile devices.
- Layered access control scheme would significantly improve user satisfaction on access control.
- Participants reported that three layers are enough for their needs.
- Participants showed their interests in ad-hoc access control mechanisms.
- Young children were viable threats for their parents in terms of access control on mobile devices (especially tablets).

# 6. Unified Authentication Framework

The previous chapters described novels findings about password usage in the wild, user authentication to devices, and efficient access control schemes. In this chapter, I report on the Unified Authentication Framework, which I designed based on insights drawn from my exploration of better techniques for user authentication[4].

While passwords are the most common form of authentication today, a great deal of past research has shown that people have many difficulties memorizing and managing passwords in a reliable and secure manner (e.g. [54,65,88]). When passwords were first introduced in the 1960s, only experts had access to computers, and they only had to manage a few passwords. However, the landscape has changed dramatically: there are far more users, each managing numerous passwords, using systems with different security requirements, and facing new kinds of attacks. The computing landscape is changing as well. More and more physical objects are equipped with computation, storage, sensing, and networking capabilities. These smart devices will need some sort of user identification or authentication. However, not all smart devices will have input capabilities suitable for password-based authentication (e.g. typing and pointing). We are at an inflection point for authentication. The increasing burden on end-users, the growing number of security

[4] The results reported in this chapter appeared in [51]

breaches, and the rise of the Internet of Things all point to the need for better user authentication.

One approach is to completely replace passwords. However, a survey paper of authentication techniques suggests that passwords have many positive properties over alternatives [22], in particular high deployability. Furthermore, passwords are still good enough for many cases if used appropriately, e.g., if they are long, randomly generated, unique among multiple accounts, and updated periodically.

I take an alternative tack: instead of completely replacing passwords, I propose to build machine-readable interfaces to handle users' credentials, and have a smart device manage these credentials appropriately (e.g., using strong passwords, updating them periodically, and doing authentication on behalf of users). By using passwords as a backend, my approach maintains backward compatibility with existing services and user practices. This preserves the positive properties of password-based authentication while addressing the aforementioned problems. Furthermore, letting a smart device manage credentials opens up new opportunities, such as using sensors on the device to authenticate a person to the credential manager (instead of always relying on a master password), supporting authentication to physical devices (as opposed to existing password managers that only support authentication to online services), and offering a potential transition path to stronger forms of authentication in a manner that is transparent to end-users.

In this chapter, I present UniAuth, a Unified Authentication Framework for facilitating authentication both for online services as well as physical devices. I also present the design, implementation, and evaluation of Knock x Knock, an implementation of the UniAuth authenticator that can manage credentials for online services and Mac laptops.

## 6.1 UniAuth Design Goals

Based on my past research on user authentication and the existing literature, I compiled a list of security and usability design goals to address important issues in user authentication. UniAuth has been designed to satisfy these goals.

***G1: Should provide secure user authentication from the smart device managing a user's credentials to services and devices***

In terms of security, the primary goal of this work is to provide secure user authentication to services and devices. To guarantee the security of user authentication, first, the user authentication from a smart device managing credentials to service and devices should be secure.

***G2: Should provide secure and usable user authentication to the device managing credentials***

The security of user authentication to services and devices also depends on the security of user authentication to the device managing credentials. When a smart device manages a user's credentials, user authentication to the device becomes crucial. Thus, it is of great importance to ensure secure and usable authentication to the device.

***G3: Should support existing password-based authentication***

It is often claimed that passwords should be replaced with alternatives. However, passwords also have advantages against alternatives [22]. For instance, password-based authentication is easy to deploy and does not incur learning cost on users because users are already familiar with the system. Furthermore, modifying all existing password-based authentication system incurs huge cost. Thus, the system should support services using password-based authentications.

***G4: Should be easy to deploy***

Existing alternative to password-based authentication system have their challenges in their deployability [22]. Some of them require simultaneous adoption both on service sides and user sides, need trusted third parties, or incur considerable learning cost on users. To address this challenge, UniAuth should be designed to maintain high deployability.

Past and on-going attempts to improve user authentication focus on G1 and G2 paying less attention to G3 and G4. For instance FIDO requires supports of their protocol on server sides, adoption of FIDO client on users' sides, and trusted third parties to make FIDO framework work. Clef does not need trusted third parties; however still need its adoption on both server sides and users' sides. Not satisfying G3 and G4 causes bootstrapping problems, which is likely to result in low adoption

in practice. In contrast, I develop UniAuth taking G3 and G4 into account to address bootstrapping problems and make a real world deployment possible.

## 6.2 UniAuth Overview

The core idea behind UniAuth is to have one's smart device handle all tasks related to credential management, such as account creation, authentication, password updates, and account termination, with minimum interaction by users. The main strategy is to offer machine-readable interfaces that UniAuth clients can use to communicate with services. This approach also allows us to offer several features not supported in existing password management systems, such as notifications of logins and authentication to physical devices.

With UniAuth, users only need to authenticate themselves to their smart devices a small number of times a day (with the assistance of sensors) to access their accounts for online services as well as physical devices.

UniAuth consists of three software components: a Universal Identity Management Protocol (UIMP), UniAuth clients with Context-Aware Scalable Authentication, and UniAuth proxy.

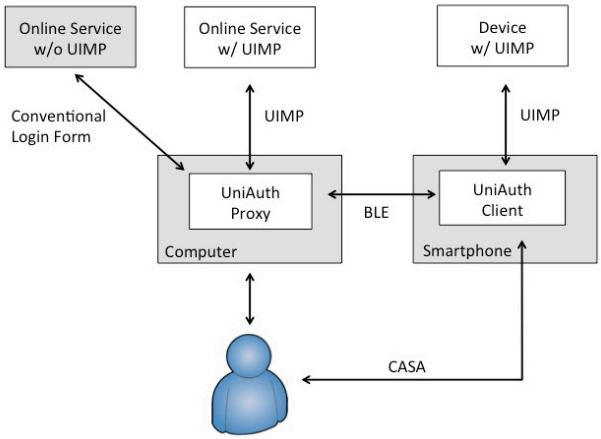



**Figure 6.1 Overview of UniAuth software components. a User's credentials are stored in UniAuth clients running on his smartphone. When he logs into his account on computers, the UniAuth proxy on the computer communicate with the UniAuth client via BLE to obtain credentials.**

*UIMP* is a set of REST-based APIs that enable UniAuth clients and proxy to communicate with services to complete tasks related to credential management. Today, many aspects of authentication are only human-readable or require manual intervention. Examples include password composition policies, password updates, account creation, and login pages. UIMP aims to create a protocol that machines can understand and support to complete these tasks. If websites and other devices implement UIMP, then any UniAuth client can interact with them with minimum human intervention. This enables more secure and usable credential managements.

The core functionality of UIMP is intentionally designed to mimic the account and authentication functionality that users see in HTML. This allows services to make them compatible with UIMP by only adding a thin wrapper on top of existing systems. Furthermore, there are optional functionalities including notifications.

*UniAuth clients* are authenticators that manage users' credentials using UIMP. The clients can be implemented on many different types of smart devices, e.g., smartphones or wearable devices. UniAuth clients communicate with services through UIMP on behalf of users. For instance, when users want to sign up for a service, their UniAuth client can create an account by generating a strong and unique password that complies with any required password policies, providing requested personal information (such as email addresses), and storing the credentials in a secure manner. When they want to be authenticated to a service (or a physical device), their UniAuth client automatically provides a credential. These features provide secure and usable authentication to services. To address the bootstrapping problem, UniAuth clients should also work with services that do not support UIMP. In these cases, functionality is limited to what is possible without UIMP. Nevertheless users can get immediate benefit by using UniAuth clients without waiting for services to support UIMP. This would facilitate adoption of the clients and push service providers to support UIMP.

By having ones' smart device manage authentication, the burden of authentication can be shifted from end-users to their devices. However, a new threat is having the smart device stolen. I expect UniAuth clients to help protect users' credentials based on the CASA concept described in Chapter 4 and onboard sensors. For example, for places with reasonable physical security (e.g., physical locks) like work and home,

UniAuth clients can operate focusing more on convenience, while for places that they rarely or never go to, or for situations that the system deems risky, the client can operate in a high security mode. I also expect UniAuth clients to provide tiered access control on the credentials stored in it rather than enforcing all-or-nothing access control. For instance, a credential for the New York Times web site may only need a low level of assurance that a person is indeed the legitimate user, whereas a credential for one's bank may want a high level assurance. This tiered credential management system should satisfy users' need better than conventional all-or-nothing approach as reported in Chapter 5.

A *UniAuth proxy* is an applications typically running on users' computers that mediate the communication between browsers and UniAuth clients. A UniAuth proxy consists of a native application (e.g., a Mac application) and a browser extension. The native application communicates with a UniAuth client through short distance wireless communication standards such as Bluetooth Low Energy or Near Field Communication. Also, the native application communicates with the browser extension to pass or receive users' credentials. When working with UIMP-compliant online services, the native application and the browser extension mediate communications between services and a UniAuth client. For services that are not compliant with UIMP, the browser extension provides a password auto fill feature by communicating with a UniAuth client through the native application.

As noted earlier, UniAuth is based on user authentication using pairs of a user ID and a password. I made this design choice for backward compatibility with existing services, which allows partial adoption of UniAuth. That is, users can get immediate benefit by adopting UniAuth clients, without waiting for servers change anything. However, as servers adopt UniAuth, clients can gain more functionality and security, offering a potentially smoother transition to stronger security, compared to approaches that require end-users and/or servers to completely change (as is the case with FIDO [6]).

In the following, first, I describe UIMP specifications and results of expert reviews. Second, I describe design, implementation, and evaluation of Knock x Knock, an implementation of UniAuth client and proxy.

## 6.3 Unified Identity Management Protocol

Universal Identity Management Protocol (UIMP) is a set of APIs that UniAuth-compliant services should support. UniAuth clients manage users' identities stored in the services through the APIs. Although eventually UIMP will be defined and implemented to support both online services and local devices, an initial version of UIMP is designed as a set of RESTful APIs for web services. This is because majority of existing services that require identity management are web services. Later, this API set will be ported another form which is suitable for communication with local devices, such as function calls over Bluetooth or NFC.

### 6.3.1 Design Principles

UIMP is designed to enable password management applications to handle account management tasks with minimum user intervention. To satisfy the design goals of UniAuth Framework (G1 to G4), I defined four design principles. Following these principles, UIMP essentially converts tasks that users have to do with html forms to APIs as well as supporting a notification feature.

***Should be based on passwords:*** To satisfy "G3: Should support existing password-based authentication", the system should support services using existing password-based authentications. Additionally, to satisfy "G1: Should provide secure user authentication from the smart device managing a user's credentials to services and devices", UIMP should make it easy for users to adopt randomly generated passwords.

***Should be small:*** To satisfy "G4: Should be easy to deploy", UIMP should be simple and small.

***Should support partial adoption:*** Even if UIMP is small, it may be difficult for service providers to adopt all UIMP APIs at once. Thus, APIs should be able to adopted partially to further improve its deployability.

***Should support notification:*** My prior study showed that there are strong needs for notification features (Chapter 3). Thus, UIMP should include notification features as a part of its specification. This feature should be optional to satisfy "G4:

Should be easy to deploy" because supporting the notification feature require additional implementation on server sides.

### 6.3.2 List of APIs

UIMP consists of four categories of APIs: Information, Authentication, Account, and Notification. Its authentication mechanism primarily follows that of OAuth [14]. In terms of authentication mechanism, one difference between UIMP and OAuth is whether they delegate user authentication to third parties. While OAuth delegates it to OAuth provider (e.g., Google, Facebook, and Twitter), UIMP does not delegate it to third parties. This simplifies its authentication mechanism and makes it more similar to existing password-based authentication mechanism. To be compatible with UIMP, a service should support `authentication/login` API. Supporting other APIs are optional.

| Method | URI | Description |
|---|---|---|
| **Information APIs** | | |
| GET | information/service | Obtain information about service |
| GET | information/authentication_policy | Obtain authentication policy, typically a password composition policy |
| GET | information/required_user_information | Obtain a list of information required to create an account |
| **Authentication APIs** | | |
| POST | authentication/login | Provide a user ID and a password. Then, API returns a html source of a first page after login. This function is equivalent to typing a user ID and a password, and clicking a login button |
| POST | authentication/tokens | Obtain service token by providing a user ID and a password, which is used to call APIs that require authentication |
| DELETE | authentication/tokens/{id} | Revoke a specified service token |
| GET | authentication/tokens/ | Obtain a list of currently valid service tokens |
| **Account APIs** | | |
| POST | account | Create an account |
| DELETE | account | Revoke an account |
| PUT | account | Update account information |
| PUT | account/password | Change a password |
| GET | account/password/onetime | Obtain a onetime password |
| POST | account/password/recover | Request a password recovery |
| **Notification APIs** | | |
| POST | notification/entries | Create a notification entry that defines conditions and media to send out notifications |
| DELETE | notification/entries/{id} | Delete an existing notification entry |
| GET | notification/entries | Obtain a list of notification entries. |

**Table 6.1 List of UIMP APIs.**

### 6.3.3 Typical Use Cases

Here is an example illustrating how a browser interacts with a UIMP-compliant web service. A UIMP-compliant web service (e.g., twitter.com) indicates that the service supports UIMP by including a meta tag in their web pages and a link tag indicating the URI for UIMP. When UniAuth-compliant browsers (including browsers with UniAuth extensions) read the tags, they interact with web services through UIMP.

```
<meta name="uniauth" content="version=1.0">
<link ref="uimp.apis" href="/uimp/">
```

One typical use of UIMP between a browser and a web service is user authentication. If an account for this service is stored in a UniAuth client, the browser displays a dialog asking whether a user wants to log into his account (see Figure 6.2). Using the example above, when he clicks the login button, the browser obtains a user ID and a password for the account from the UniAuth client and posts the following JSON to a UIMP login API at:

```
https://twitter.com/uimp/authentication/login
```

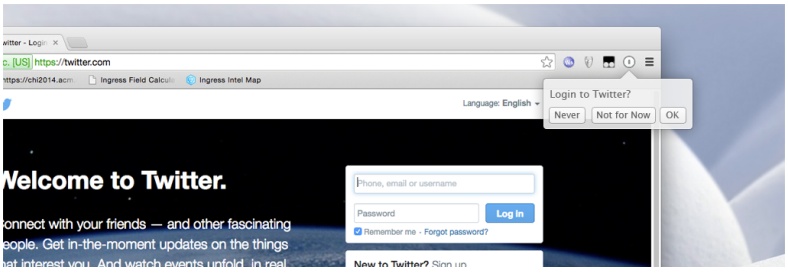



Figure 6.2. An example of login dialog with UIMP-compliant browser

```
{
        "userid" : "john@gmail.com",
        "password" : "hsagion2%dAFga!faiu2314"
}
```

The JSON contains a user ID and a plain-text password is posted to twitter.com over https. This is essentially equal to filling a user ID and a password in a login form and clicking a log in button. Thus, the server can purse the JSON and pass the

parameters to existing authentication framework. This minimizes server-side modifications to support UIMP.

In practice, websites have many hidden input parameters in their login forms. These parameters are passed to the websites when users log into the services. These parameters are typically used for traffic analysis such as from which page a user logs into the service if a service has login forms in multiple pages. If a service wants to send additional parameters in UIMP, the service can include a status parameter in a meta tag. Then, a UIMP-compliant browser can send the status along with arguments required for APIs.

```
<meta name="uniauth" content="version=1.0,status="pageid=mainpage">
```

```
{
        "userid" : "john@gmail.com",
        "password" : "hsagion2%dAFga!faiu2314"
        "status" : "pageid=mainpage"
}
```

### 6.3.4 Communication Sequence Examples

This section describes example communication sequences using UIMP APIs to complete typical tasks such as authentication and account creation. In the following examples, I assume that the browser does not directly support UIMP, and instead has a browser extension that makes the browser UIMP compatible. In future versions, the browser can take on the extension's role. The following diagram illustrates a communication sequence where a UniAuth client processes user authentication with a web service.

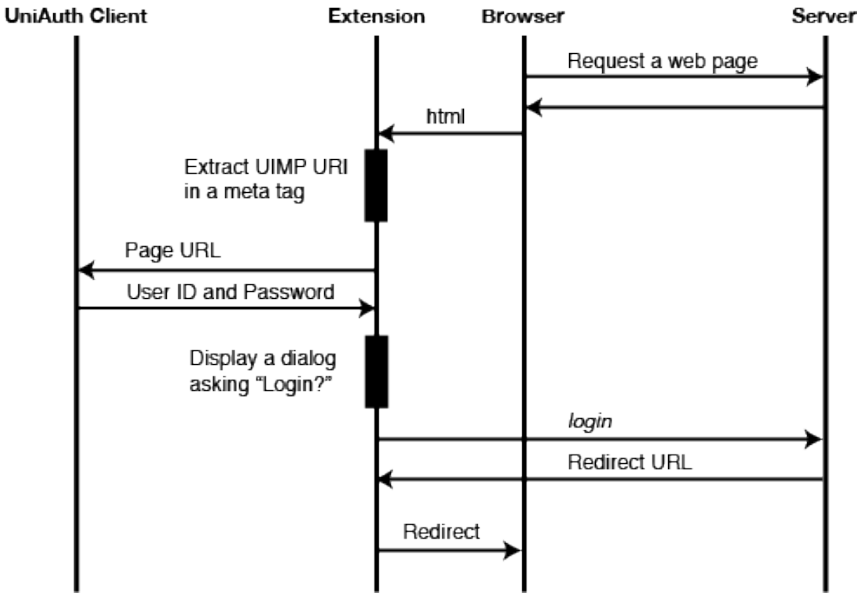



**Figure 6.3 An example of communication sequence in logging into a web service that support authentication/login API.**

When a user opens a web page, UniAuth extension tries to extract a UIMP meta tag and UIMP URI in the tag. If the extension finds these (which means the web service supports UIMP), the extension request UniAuth client to send login credentials. After the user indicates that he wants to login to the web service, the extension sends the login credentials to the web service, then, redirect to a URL received from the service after completing a user authentication on the server side. The extension sends credentials only when the domain of URL and UIMP URI match to avoid cross-site attacks. For other examples, refer to Appendix A.

### 6.3.5 Expert Review

Although UIMP is rather straightforward protocol that translates existing identity management practices to RESTful APIs, there could be issues that we were not aware of. To validate its specifications, I conducted a review consisting of interviews with 12 experts. These experts ranged from graduate students to professional system administrators. The interviews were semi-structured. First, I asked demographic questions to understand participants' expertise. Second, I explained the details of UIMP asking whether they found any issues in its design. Finally, we asked their

higher-level opinions on the concept of UIMP. The interviews took about one hour and we paid $40 for participations.

### Demographics

Among the 12 interviewees, six of them were graduate student studying web related technologies and/or information security. Two of them were researchers with Ph.D. degrees working on information security. Four of them were system administrator at Carnegie Mellon University. The researchers had been working on information security related topic eight and 10 years. The system administrators had been managing IT infrastructure at Carnegie Mellon University from three to eight years. Two of them had masters' degree in information security related fields and one had a Ph.D. degree in computer science. I started our interviews from graduate students to address basic issues, then, interviewed researchers and system administrators later in our expert reviews to identify advanced issues.

### Results

Throughout the expert review, I collected feedback from our interviewees. After each review, I updated UIMP specifications based on the feedback. After 12 expert reviews, I made 62 updates consisting of five removal/addition of APIs, 21 updates of API details such as arguments or return values, and 36 editorial updates such as descriptions of APIs. Figure 6.4 shows the number of updates that were made at $x$-th review. Because I did not find any issues other than editorial ones in three successive reviews from the tenth review, we stopped our expert review at the 12th review. The APIs described earlier in this section is the final specification after these modifications. Refer to Appendix A for the full specifications.


**Figure 6.4 The number of modification made at each review. After the 10th review, no modifications other than editorial updates were made in three successive reviews. Thus, I stopped the review at 12th review.**

As a part of this expert review, I collected qualitative feedback on the UniAuth concept. In general, experts were positive about the concept of UIMP. All experts agreed that being backward compatible with password-based authentication was important. While they agreed with the importance of backward compatibility, three experts commented that it was also important to clearly explain that passwords would be eventually replaced. Otherwise, people may think UIMP is just a marginal improvement on the existing password-based authentication.

I also asked experts to speculate what would be the biggest challenge having UIMP be adopted by existing services. Each expert raised multiple issues. Seven experts answered that it would be whether supporting UIMP made sense economically. Even though the UIMP is a thin layer over existing password-based authentication system, supporting UIMP adds extra costs, especially management cost. Thus, the additional cost should be smaller than expected benefit of supporting UIMP such as making singing up for services easier should higher. Related to this issue, six experts commented that adoption of UIMP depends on whether the number of UniAuth client users reaches a critical mass because the benefit of supporting UIMP correlates to the number of UniAuth client users. These comments were not surprising; however, they did give strong support to the importance of facilitating initial adoption of UniAuth clients. Five experts commented that ensuring the security of UIMP was the most challenging issue. After being reviewed by 12 experts, the final specifications of UIMP were less likely to have an obvious flaw; however, further review from security community would be necessary for ensuring the security of the specifications.

### 6.3.6 Summary of Findings

In this section, I reported on design on UIMP and the results of an expert review consisting of 12 interviews with experts. After the 12 reviews by experts, I believe the final UIMP specification does not have obvious issues; however, more discussion about the specification in information security community would be necessary to further confirm its security. Additionally,

- Experts are generally positive about the concept of UIMP
- Being backward compatible with password-based authentication is of great importance
- Adoption of UIMP will be strongly affected by adoption of UniAuth client.

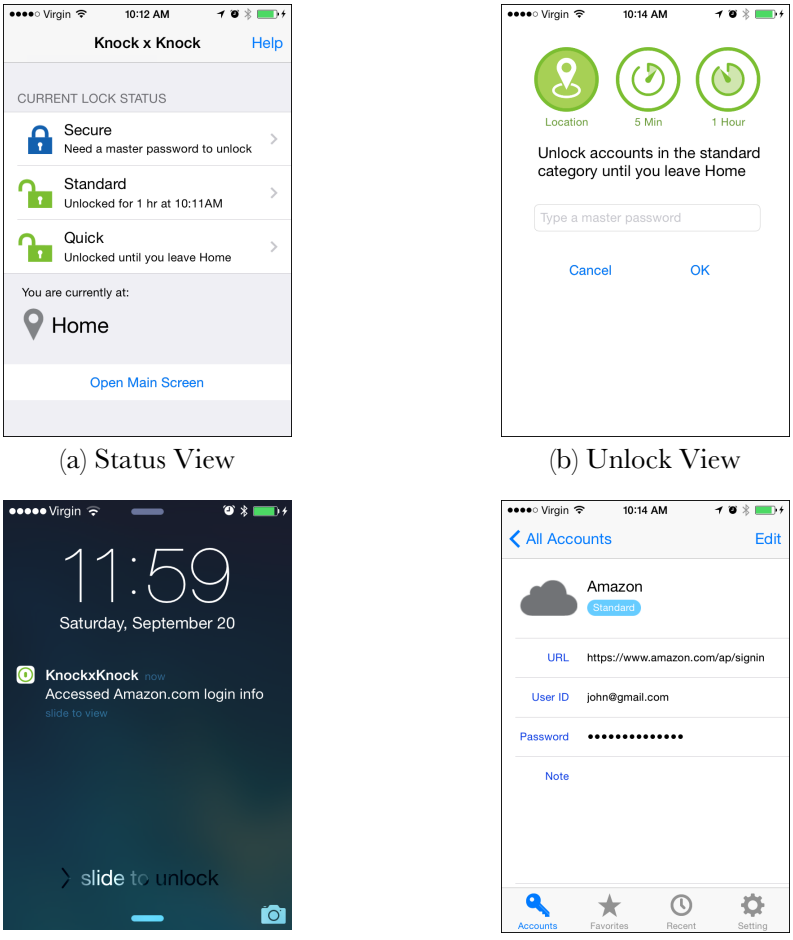
## 6.4 Knock x Knock: UniAuth Client for iPhone

The previous section confirmed that the adoption of UniAuth clients is a crucial element for adoption of UIMP and the UniAuth Framework in general. To better understand the design space and user experience for UniAuth clients, I designed, implemented, and evaluated Knock x Knock, a UniAuth client for iPhone and Mac.

Knock x Knock is a combined suite comprised of a UniAuth client on iPhone (Figure 6.5) and a UniAuth proxy on Mac (Figure 6.6). I named the system Knock x Knock after the well-known American joke intro, and because it lets users physically knock on their Mac twice to login to the Mac. In Knock x Knock, all credentials are stored in the UniAuth client on iPhone. The UniAuth proxy consists of browser extensions and a Mac application that mediates communication between the browser extension and the iPhone app. The Mac app also provides an interface that allows users to manage credentials stored in their iPhones on their Mac. Currently, we have browser extensions for Chrome and Safari.

The Mac app connects with the iPhone app via Bluetooth Low Energy (BLE) and connects with the browser extensions using a local web server. Users can also set the iPhone's BLE to have a range of 1 to 15 meters. For both connections, they perform

mutual authentication using a pre-shared key to prevent illegitimate connections. Once the BLE connection is established, it is kept until the iPhone goes out of BLE range or the Mac app stops. All data exchanges after the mutual authentication are encrypted using AES-128. Because existing online services do not support UIMP, Knock x Knock currently only supports storing/auto filling user IDs and passwords for online services and Mac. For the features requiring UIMP support on server-side, I showed paper prototypes in the post-study interviews and asked for participants' opinions on these features.




(a) Status View (b) Unlock View

(c) Notification (d) Account Detail View

**Figure 6.5 Knock x Knock displays a status view when launched by a user (a). A user can lock/unlock these tiers based on locations (b). When accessed from a UniAuth proxy, it shows a notification message (c). A user can see account information, including a password, by clicking Open Main Screen and typing his master password (d).**

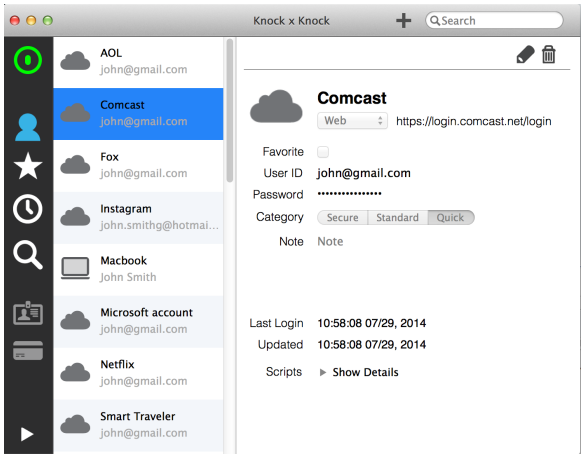



**Figure 6.6 On a Mac, users can also manage account information stored on one's an iPhone.**

### 6.4.1 Threat Model

With Knock x Knock, I focused on preventing common attacks on online services. The most common threats are online/offline attacks against users' accounts such as dictionary attacks and reverse brute force attacks. Reverse brute force attacks are a type of attacks against password-based authentication where an attacker tests a password against multiple users. I also consider attacks targeting users such as phishing attacks and shoulder surfing attacks. In contrast, I assume that cryptographic primitives protecting the framework are secure and properly used. Thus, communications between entities are secure, and the credential database cannot be compromised without knowing a master password. Also, I assume that there is no malware running on users' computers or smartphones. Another potential threat against Knock x Knock is device theft. I discuss the potential risks of device theft in the discussion section in detail.

### 6.4.2 Tiered and Location-Aware Access Control

In terms of access control, most existing password management systems have all-or-nothing access, i.e., allow access to all accounts or do not allow access to any accounts stored in the systems. However, the study described in Chapter 5 showed that it was difficult to satisfy different security and usability requirements for different accounts with this approach. Thus, to satisfy "G2: Should provide user authentication to UniAuth client balancing security and usability", I adopted a tiered and location-aware access control system to address this challenge.

In Knock x Knock, each account is stored in one of the three different tiers: Secure, Standard, and Quick (see Figure 6.5 (a)). Each tier has a lock state and can be locked or unlocked independently. Thus, unlocking the Secure tier does not unlock the Standard or Quick tiers. To access account information in a tier, the tier should be in unlocked state. For instance, if a user saved his Amazon account in the Standard tier, he has to unlock the Standard tier first to let Knock x Knock fill his credential for Amazon on Mac. The number of the tiers is based on the study in Chapter 3 reporting that three categories were appropriate when people categorize applications on their smart phones.

Each tier has different security level (Table 6.2) to balance different security and usability needs for wide variety of credentials. Knock x Knock automatically locks and unlocks the Standard and Quick tier based on whether users are at *trusted locations* to make account accesses easier. Users can register trusted locations such as homes and workplaces in their iPhone clients. In the current implementation, a location is a 100 meter radius from a registered geolocation. Entering a trusted location unlocks the Quick tier automatically. This lets users log into accounts in the Quick tier on their Mac without touching their phones. Exiting a trusted location locks the Quick tier automatically. The Standard tier can be unlocked by typing a master password on iPhone (Figure 6.5 (b)). If this happens at a trusted location, the Standard tier is unlocked until an iPhone exits from the trusted location. The Secure tier can be unlocked for one-time access by typing a master password regardless of location. It gets locked after one of the accounts in that tier is accessed from a Mac. In addition to these options, a user can unlock any tier for a specified period (from five minutes to one day) by typing his master password regardless of location. Finally, a user can lock a tier manually at any time.

| | Trusted Locations | Other Locations |
|---|---|---|
| Secure | Typing a master password unlocks the Secure tire. After accessing one credential Knock x Knock automatically lock the tier | |
| Standard | Typing a master password unlocks the Standard tier until a user leaves the locations | Typing a master password unlocks the Standard tire for 5 minutes |
| Quick | Knock x Knock automatically unlocks the Quick tier while a user is at trusted locations | Typing a master password unlocks the Quick tire for 5 minutes |

**Table 6.2 Unlocking and locking conditions for each tier/location pair. Credentials in the secure tier always require a master password to be accessed.**

### 6.4.3 Storing and Managing Credentials

When a user logs into an account on a web site for the first time, Knock x Knock pops up a dialog asking if he wants to save his credentials. The user can edit the name of the entry and its tier. When he clicks OK, account information is sent to his UniAuth client. The account information consists of a user-configured name of the account, a URL, a user ID, a password, and IDs of a user ID field and a password field in a login form. After storing the information, UniAuth automatically fills the credentials when he accesses the same website later.

He can also open a main screen by typing his master password to browse and edit stored account information (Figure 6.5 (d)). This allows users to view their user IDs and passwords to log into accounts manually if needed. I made this design choice to satisfy "G4: Support current credential management practices". This feature could make the system vulnerable against certain types of phishing or social engineering attacks. On the other hand, if the system does not allow users to see their password directly, users might not adopt the system over concerns about corner cases (e.g., logging into their accounts on foreign computers or sharing an account with others). Thus, I decided to make it possible to access account information by typing the master password.

A user can also browse and edit account information using a UniAuth proxy on Mac (Figure 6.6) as well. To access account information on Mac, categories that the account information belongs to should be in an unlocked state. To mitigate the risks of phishing and/or social engineering attacks, it is possible to display warning messages about these attacks when users directly access their credentials by typing their master passwords.

Another important design choice here is where to store users' credentials. Fundamentally, there are three different places to store the credentials: a centralized server, a user's computer, and a mobile device. The primary advantage of storing credentials in a centralized server is that a user can access their credentials from multiple devices as long as these devices are connected to network. LastPass [9], a commercial password management system, adopted this approach. However, because a centralized server stores many credentials, attackers could launch costly

sophisticated attacks against the server to obtain the credentials. In 2015, attackers compromised LastPass's servers and stole all of their users'encrypted master passwords. This potentially gives attackers access to all credentials stored in LastPass [10]. Availability is another challenge in storing credentials in a centralized server. LastPass has also faced server outages lasting more than a day. During this server outage, LastPass users could not access their credentials and there was nothing they could do to recover the availability.

The second option is to store credentials on a user's computer. Most existing password management systems (e.g., 1Password and browser built-in password managers) adopted this approach. This architecture works quite well if a user always accesses their accounts from one computer. However, if a user wants to access their accounts from multiple devices, credentials have to be synchronized among the devices. This increases the risk of security breaches because attackers can compromise one of the devices to gain access to all of one's credentials. Furthermore, this approach cannot support authentication to physical devices because computers are not always on and users do not carry them all the time.

Finally, the third option is to store credentials on mobile devices and access the credentials from other devices over short-range wireless communication (e.g., Bluetooth Low Energy) when necessary. An advantage of this architecture is that a user does not have to synchronize credentials among multiple devices. Furthermore, this approach can support authentication to physical devices. As a disadvantage of this approach, attackers could target the mobile devices. For instance, attackers could physically steal the devices to obtain credentials. However, it would be less likely to happen because the cost of physically stealing devices is much higher than remotely attacking the devices.

Availability of the mobile devices could be another challenge in this approach. For instance, devices could become unavailable when a user loses their device or its battery runs out. If a user loses their device, the user can recover their credentials from an encrypted backup stored in iCloud. Similarly, if a user's device runs out, the user can simply charge the device. In the case where charging the device is infeasible, the user can fall back to resetting passwords using conventional schemes (such as answering secret questions) provided by services to gain immediate access to

their accounts. Then, after their device becomes available, they can update passwords stored in the device. In either case, they could have more control over the availability of their credentials compared to the centralized server approach.

As described, there are many tradeoffs related to where to store credentials. In this work, I would like to support authentication to physical devices (e.g., Internet of Things). Furthermore, the study reported in Chapter 3 showed that users preferred to have more control on their password management tools. Thus, I decided to store credentials in iPhones in Knock x Knock.

### 6.4.4 Logging into Web Accounts and Mac Laptop

When a user opens a web page with a password field, our browser extension sends the URL to the UniAuth client, which then looks up whether a corresponding account is stored for the domain. If an account is stored and its tier is unlocked, it sends a credential to the browser extension. The browser extension then changes the colors of text fields in the login form to green, indicating that an account was found in Knock x Knock. A user can double click on one of these fields to fill the credential. If the tier is locked, a dialog asking the user to unlock the tier appears when he double clicks the text fields.

Users can also log into their Mac by physically knocking on their Mac twice, as if knocking on a door. Alternatively, users can double click the command button. This login will typically happen when a user wakes up a Mac from sleep mode or from a password-protected screensaver (Figure 6.7). I implemented this feature to let participants experience logging into physical devices with authenticators so that they can speculate how the user experiences could be in a situation where authenticators manages authentication to physical devices as well as online services. When the UniAuth proxy detects either of these cases, it requests that the appropriate UniAuth client send a password for the Mac. If the credential for the Mac is stored in the client and its tier is unlocked, the client sends a password back to the proxy; then, the proxy generates fake key type events to fill a password field and log into the Mac automatically.

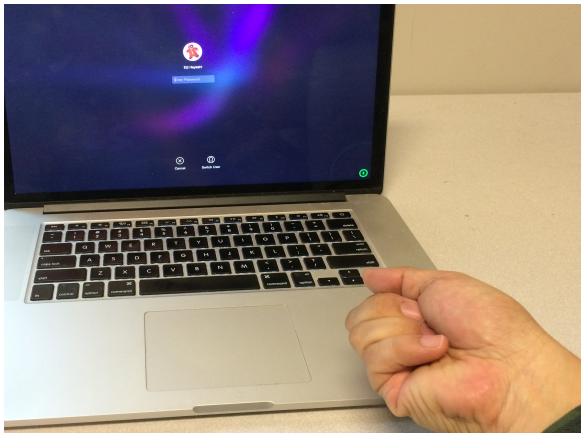


**Figure 6.7 Knocking Mac twice to unlock. The green icon at the bottom right of the display indicates that a user's iPhone is nearby and Knock x Knock app on the iPhone is connected to this Mac.**

When a credential is accessed from UniAuth proxy, a UniAuth client shows a notification message (Figure 6.5 (c)). This notification helps lets a user know that someone accessed his account if he is still in BLE range but not in front of the computer (e.g., right after he left his desk).

## 6.5 User Study

The overall goal of the user study was to evaluate the user experience in using Knock x Knock to manage one's authentication needs. Knock x Knock has several features that have been individually investigated in past work (e.g., [48, 58, 92]), but have not yet been evaluated as an integrated whole. Furthermore, Knock x Knock has several novel features not present in existing systems. Although I expected strong interaction effects among these features, no work has investigated them empirically. Thus, I conducted a one-week preliminary study and a three-week field study to investigate user experiences with Knock x Knock, and their perceptions on the concept of UniAuth.

### 6.5.1 One-Week Preliminary Field Study

The first study was a one-week field study with six participants, where I tested our study protocol and looked for possible technical glitches. In general, Knock x Knock worked well. I found one technical issue that caused a problem when iOS killed the Knock x Knock process because of memory pressure. Further analysis showed that

this was a bug in iOS. We reported the bug to Apple, and Apple fixed the bug before I started the next study. In terms of the study protocol, I added a few data logging capabilities to better understand participants' usage patterns and to refine the interview questions. Other results found in this study overlapped with ones found in the next study, and so I do not report the results here.

### 6.5.2 Three-Week Field Study

I conducted a three-week-long field study to explore user experiences in managing credentials with Knock x Knock. The study consisted of two in-lab sessions at the start and end, and two interviews in between. In the first session, we installed Knock x Knock on participants' iPhones and Macbooks, and explained how the system worked. I also asked participants to store their user IDs and passwords for their Macbooks and for five of their existing online accounts. Participants chose online accounts ranging from casual accounts such as ones for online bulletin boards to important accounts such as banking accounts. If participants had already saved user IDs and passwords for these account in web browsers, we deleted the information to let them use Knock x Knock to log into the chosen accounts.

After the first session, I had two semi-structured interviews for each participant at the end of the first and second week. The interviews were done either face-to-face or over the phone and were 15 minutes long. I conducted these interviews to investigate how the participants' perceptions of the system changed over the study period.

Three weeks later, I had the second in-lab session where I conducted post-study interviews. The interviews were semi-structured and took about 45 minutes. In the interviews, I asked for participants' thoughts on Knock x Knock as well as potential features that Knock x Knock can support with services supporting UIMP. I created paper prototypes showing how these potential features would work and asked for participants' opinions on them. Along with interviews, I collected application logs to analyze participants' usage pattern objectively, totaling 68,032 logs from the participants' Macs and iPhones. We paid $75 USD for completion of the study.

### 6.5.3 Participants for the Three-Week Field Study

I recruited 13 participants who were using both iPhones and Macbooks, and using either Chrome or Safari as their primary web browsers. The recruitment was done

through a university's recruitment website which is meant to public. Seven participants were male and six were female. Their age ranged from 19 to 42 with a mean age of 27. Four of the participants were students, one was a university staff, seven were employed outside of the university, and one was unemployed. All participants, except one participant using KeePass [7], used browser password managers. In addition to browser password managers, four participants used physical memos. Two participants used text files. One participant used 1Password [1] to manage their credentials.

### 6.5.4 Summary of Knock x Knock Usage

On average, the participants stored 7.7 accounts after three weeks of using Knock x Knock (1.1, 3.4, and 3.3 accounts in the Secure, Standard, and Quick categories respectively). In the first session, If asked them to register five online accounts and a Mac account. After the first session, six participants added one to nine accounts on their own accord. All participants stored their accounts in at least two different tiers. Eight participants used all tiers.

Our participants logged into their Mac 10 to 171 times with a mean of 48.2 (SD=50.2). They also logged into their online accounts 9 to 155 times with a mean of 45.1 (SD=47.8). In total, they logged into their accounts 22 to 192 times with a mean of 93.3 (SD=58.5). Figure 6.8 shows the distribution of the login frequencies. The white and gray parts of the bars represent logins for Mac and online accounts, respectively. The data showed that the number of logins for online accounts was a few times a day on average. This was because most of our participants did not log out from their accounts. In the post-study interview, most of the participants told that they usually kept themselves logged into online services and put their computers in sleep mode when they finish. Thus, logins happened only when they restart browsers, typically by restarting their computers. Altogether, the data showed that the participants had reasonable amount of exposure to Knock x Knock to evaluate it.

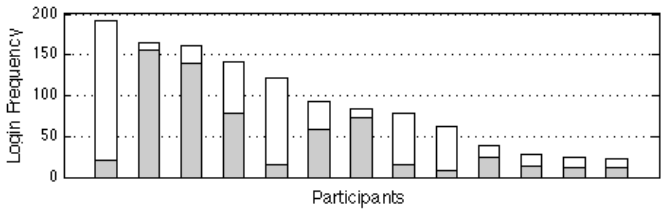



**Figure 6.8 Numbers of logins for each participant in rank order. White denotes logins to Mac and gray represents logins to online accounts.**

In the semi-structured interviews at the end of weeks one and two, I asked about participants' experiences with and perceptions of Knock x Knock. The participants were generally positive about Knock x Knock and its features. After two weeks, all participants told me that their experiences using Knock x Knock were consistent with those reported in previous interviews.

In the post-interview, I solicited participants' opinions on Knock x Knock using a 5-point Likert scale (5 = very positive) as well as open questions. In the following description, the numbers in parentheses denote the means of our participants' ratings. In general, participants were very positive about Knock x Knock. They agreed (4.1) with the sentence "I will use Knock x Knock if it is available." P15 commented, "This is not only convenient. If I were the person with million passwords, this streamlines everything. But then also being someone who is less secure, really really makes it very simple and helps make things more secure." P9 said, "It is [an] easy way to storing passwords. It's safe. I feel that way because it's in your phone, and your phone should be in a certain distance." P10 also commented on its security, "It seems more secure since you could use that when the phone is close to a computer you are accessing compared to randomly save passwords in a browser." During the interviews, all participants asked whether there was a plan for public release. Two participants asked whether they could keep using Knock x Knock after the study. These responses clearly illustrated that participants were very positive about Knock x Knock.

I also asked whether they noticed an increase in battery consumption on their iPhones. All except one participant answered that they did not notice it. When I asked participants to describe the disadvantages of Knock x Knock, they pointed out that sometimes establishing Bluetooth connection between an iPhone and a Mac was

slow. Figure 6.9 shows the distribution of the connection time extracted from the application logs. On average, the connection took 9.8 seconds. The connection time was not an issue in authentication for online services because, in most cases, there would be enough time to establish connections in the background after users activate their computers, before they log into their online accounts. In contrast, this was a clear issue for logging into Macs.

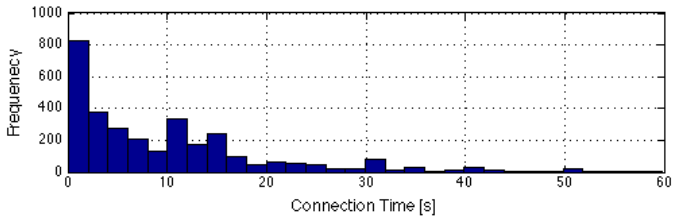



**Figure 6.9 BLE Connection time. 31% of times, connections were established within 3 seconds. Because of memory management bugs in iOS, 42% of connections took longer than 10 seconds to be established. This bug has been fixed in the latest iOS.**

I found two causes for delays. First, to preserve privacy, the iPhone changes its Bluetooth ID about every 15 minutes. This implementation led to two different cases in establishing connections in UniAuth. The first is where a UniAuth proxy tries to establish a connection before an iPhone's Bluetooth ID changes. A typical scenario is a user walking away with the iPhone, and then coming back before the iPhone's Bluetooth ID has changed. In this case, the connection is established within a second because the UniAuth proxy knows which Bluetooth device to reconnect to. However, in the second case, the Bluetooth ID has changed, and the proxy now has to search all nearby Bluetooth devices, establish a connection, and perform mutual authentication. This took up to about 10 seconds depending the number of nearby BLE devices. One potential solution is to use a wireless communication standard with shorter connection time, such as NFC. However, with Apple's BLE implementation, I do not see any easy workarounds.

The other cause for delays is a bug in iOS's BLE implementation. iOS fails to initialize BLE when there is not enough memory space. Once this error happens, it takes 10 to 30 seconds to timeout. After this timeout, the proxy can then restart the connection process. These cases consisted of data points where connection took longer than 10 seconds in Figure 6.9. This bug has been fixed in the latest iOS. Although the connection delay undermines the usability of Knock x Knock, it was

very encouraging that participants reported that they wanted to use Knock x Knock despite these failure cases.

### 6.5.5 Post-Study Interview Results

After participants users Knock x Knock for three weeks, I conducted post-study interviews. The interviews were semi-structured and lasted about 45 minutes. In the interviews, I investigated the participants' perceptions on both security and usability aspects of Knock x Knock. Table 6.3 summarizes the participants' response to Likert-scale (1 to 5 where 5 denotes strongly positive) questions asking about security and usability of features in Knock x Knock. The participants were generally positive about the security and usability of these features. In the followings, I will discuss qualitative findings about these features. The numbers in parentheses are medians/means of the participants' responses to Likert-scale questions.

| Feature | Security | | Usability | |
|---|---|---|---|---|
| | Median | Mean | Median | Mean |
| Storing credentials on iPhones | 4.0 | 3.9 | 4.0 | 4.2 |
| Knocking Mac to log in | 4.0 | 3.7 | 5.0 | 4.3 |
| Three security tiers | 4.0 | 4.6 | 5.0 | 4.6 |
| Location-aware lock control | 5.0 | 4.6 | 4.0 | 4.2 |

**Table 6.3 Summary of participants' responses to Likert-scale questions asking about security and usability of Knock x Knock features. 5 denotes strongly positive and 1 denotes strongly negative.**

#### Storing All Accounts in a Device Makes Things Simple

The participants rated storing account information in Knock x Knock on their iPhones as secure (4.0/3.9) and easy to use (4.0/4.2). The participants were very positive about storing account information in one device. P15 commented, "You don't have to invest a lot of time in thinking through what is my password, where did I store it. […] The transition to come to put them all here makes things very simple." Furthermore, many participants believed that mobile phones were more secure than computers because they were more personal devices. P5 noted, "No one else has my iPhone because it's always with me. No one knows my [PIN] code. I let people use my computer, but don't let people use my phone." Two participants were concerned that their phone may be compromised; however, they also commented that they were not worried about it too much because they always keep their phones with them.

## Knocking Mac to Unlock was Enjoyable

The participants rated logging into their Mac using Knock x Knock as secure (4.0/3.7) and easy to use (5.0/4.3). They generally thought it was secure because of proximity. P13 said, "If my phone is nearby, and I'm nearby." P5 also commented on the combination of tiered security. She said, "It's good, because you can control on phone what other accounts they can access. Even someone logins my computer, just having my computer doesn't mean they can access everything else."

Three participants reported that knocking their Macs to unlock was enjoyable. While users regard most of the security systems as *burden*, it was very interesting that they reported it enjoyable. In terms of security, P1 commented, "I think it's OK. Even if someone knocks my computer right after I leave my office, I will receive a notification [on my phone]. Then, I can come back to see what's going on because I'm still in a BT [Bluetooth] range." As an interesting side effect, P5 reported, "One time, my friend asked me what I was doing on my computer, I explained it, and he was really interested." A high level of observability has been suggested as one factor that can help facilitate wider adoption of security systems [31]. On the other hand two participants showed their concerns over observability. P10 noted, "Someone can just come and click this button especially when this becomes something widely noted. […] There are cases where I have to leave my stuff [both laptop and phone] like this." For these cases, users can opt out from unlocking Mac with Knock x Knock by not storing its credential in Knock x Knock. Another possible solution would be putting Knock x Knock on wearable devices, such as smart watches, to maintain close proximity to users.

## Not Unlocking Everything at Once

The participants were very positive about having tiered security levels. They rated it as very secure (4.0/4.6) and very easy to use (5.0/4.6). P7 reported, "I think it's secure because I have an option which one to unlock. So, it's not something like you turn on everything at once. There are different security levels for different websites." A participant using 1Password (P1) told us that, "1Password locks when a screensaver becomes active. So, I have to type my master password again and again throughout a day. I understand it's necessary to protect important accounts, but, I don't want to type master password to access a news website." If a password

manager only has one tier for accounts, it makes sense to put more weight on security rather than usability so as to protect highly important accounts. However, this approach is suboptimal for people who have many less important accounts, and access those accounts frequently. Having multiple tiers addresses this issue and allows users to balance security and usability based on their needs.

One non-trivial design choice here is the number of tiers to have. In the post-study interviews, we asked our participants whether three was appropriate for them. Nine said that three was appropriate. Three answered that they needed only two for now, but having three tiers made sense to them. One said that two tiers would be better in terms of simplicity. None of the participants wanted to have more than three tiers. These results indicated that three is a reasonable number for tiered account management.

## Tiered Access Control with Location-Awareness

Knock x Knock locks/unlocks Quick and Standard categories based on whether participants' phones are in trusted locations. Ten of our participants registered two locations and three of them registered three locations. These were mostly their workplaces and homes where physical access was limited. The participants rated this feature as very secure (5.0/4.6) and easy to use (4.0/4.2). P10 commented on its security, saying "I trust people at my home or workplace [enough] to use [location-based lock control]. Or, there is no huge risk of someone hacking into my places. So I think it's secure." P13 commented on its usability, "Especially for the quick category, […] these are accounts that I really don't care. So, having easy access is important." These comments indicate that, combined with tiered access control, location made access to the accounts in Standard and especially Quick tiers very easy while maintaining desirable levels of security for participants.

The current simple definition of a trusted location (100 meters from a user specified location) was sufficient for most people, but I also saw some hard cases. For example, a university staff member (P7) reported that she regularly visited multiple buildings on campus to attend meetings, and that she opted to unlock categories for a day instead of registering all of them as trusted locations. One possible solution is more flexible or ad hoc configurations of trusted locations. Another possibility is to automatically infer trusted locations. However, it is important to note that the

participants reported that they chose trusted locations based on their perceptions of security at these places rather than frequency of visits. For example, P12 noted, "Even if I am frequently at a coffee shop, I don't want to add it to [trusted] locations because its security is unclear." I believe that algorithmic techniques for determining the level of physical security of a location could be a compelling area for future research, for the case above, as well as for modulating the level of security needed for the Internet of Things.

### 6.5.6 Potential Features for Future Versions

In the post-study interview, I also showed participants paper prototypes of three UniAuth features that require server-side modifications, to gauge the potential usefulness of these features. For each feature, I asked participants to estimate the usefulness and their willingness to use the feature. Table 6.4 summarizes the participants' responses to the Likert-scale questions. While I used paper prototypes to illustrate the features, these features were rather simple. Furthermore, the participants had used Knock x Knock for three weeks. Thus, the participants would have enough exposure to Knock x Knock to speculate how the potential features would work in practice.

| Feature | Usefulness | | Willingness to use | |
|---|---|---|---|---|
| | Median | Mean | Median | Mean |
| Unified account creation | 4.0 | 4.2 | 4.0 | 4.1 |
| Managing password updates | 4.0 | 4.4 | 5.0 | 4.5 |
| Server-side account access notification | 5.0 | 4.5 | 4.0 | 4.5 |

**Table 6.4 Summary of participants' responses to Likert-scale questions asking about usefulness and willingness to use of potential Knock x Knock features. 5 denotes strongly positive and 1 denotes strongly negative.**

#### Unified Account Creation

Unified Account Creation feature provides users a way to create their accounts easily. When a user visits a UIMP-enabled website for the first time, Knock x Knock can ask if he wants to create an account. If so, it displays a dialog with input fields for all information requested by the website such as a name and an email address. These fields are pre-populated with personal information stored in Knock x Knock along with a randomly generated password. Also, if he prefers, he can overwrite the password with his own manually created password. Then, when he clicks the OK

button, Knock x Knock communicates with the website to create an account and store the account information in Knock x Knock.

The participants were generally positive about this feature, rating it as useful (4.0/4.2) and agreeing that they would use it if it were available in Knock x Knock (4.0/4.1). Also, they agreed that they were not concerned about storing personal information in Knock x Knock (4.0). Most participants preferred that they did not have to type their information multiple times to create accounts for different services. Three of them specifically commented that they liked that passwords were randomly generated. Two participants commented that this feature smoothed out the account creation processes. P15 noted, "When you create your account, it signs you in, and you buy something. But, then, you have to log out, revisit a website, and log in to save a password. This makes all in one seamless step." As illustrated in this comment, in existing account creation procedures, a user has to take extra steps to save credentials in password managers after creating an account. However, with UIMP, password managers can create accounts and save credentials seamlessly.

## Managing Password Updates

With this feature, users can receive a reminder after a specified time since one's last password update. Users can select the period (e.g., 3 months), or it can be specified by a service if periodic password updates are enforced. When receiving a reminder, users can manually update a password in Knock x Knock via UIMP, or let Knock x Knock automatically update a password and notify afterward.

The participants were quite positive about this password update management feature, rating it as useful (4.0/4.4) and strongly agreeing they would use it if available (5.0/4.5). Only one participant said he would configure his passwords by himself when receiving password update requests. Other participants preferred to let Knock x Knock generate random passwords. Interestingly, this result goes against the finding reported in Chapter 3.2 that users did not like using automatically generated passwords. I believe the discrepancy is due to the expected availability of password management tools. While the above study was about password managers in general (software, paper, etc), the participants evaluated this feature as a part of Knock x Knock on users' iPhones, which are almost always available to them. In my post-study interviews, four participants commented it was important for them to be

able to see their passwords on their phones when using randomly generated passwords. P15 noted, "[If I use automatic password update] I even don't know what my password is. But, it is so easy to go to my phone, type the master code and look at it." These results indicate it is important to provide users dependable ways to access their passwords directly, to convince them to use randomly generated passwords, even if they hardly access their passwords in these ways.

### Server-Side Account Access Notification

In Knock x Knock, I have already implemented a notification feature that shows a message on iPhone when account information is accessed by a UniAuth proxy. On the other hand, this feature would be better if implemented on the server-side, sending notifications to users when somebody accesses a user's account.

The participants rated this feature very useful (5.0/4.5) and they strongly agreed that they wanted to use this feature if it were available (4.0/4.5). P8 commented, "I think it's really useful and makes me trust Knock x Knock more. It makes me feel more secure." Participants also requested detailed control on when to receive notifications. P8 said, "I don't want to receive notifications when someone accessed using Knock x Knock. But, something more important ones, like my bank accounts, I'd like to receive notification always." Eight participants mentioned they wanted to receive notifications only for account accesses without using Knock x Knock. As illustrated by these responses, our participants strongly desired notifications, but did not want to be overwhelmed by them. Thus, it is crucial for UniAuth Framework to have the capability of distinguishing whether an account access is from a user's UniAuth client or not.

## 6.6 Discussion

The results demonstrated that Knock x Knock worked well in real world settings and that participants would prefer to use it if it were available. However, there are many interesting questions that still can be investigated.

### 6.6.1 Availability of Devices

One of the fundamental assumptions in the Knock x Knock system architecture is that users' iPhones are always available to them. However, there are three typical

cases where a user's iPhone might be unavailable. The first case is where an iPhone is not in close proximity. For instance, a user could leave their iPhone at home. The second case is where an iPhone stops working, typically because it runs out of battery. In these cases, users can fall back to resetting passwords using conventional schemes (such as answering secret questions) provided by services to gain immediate access to their accounts. Then, after their iPhone becomes available, they can update passwords stored in Knock x Knock with new passwords. The last case is where an iPhone becomes unavailable permanently (e.g., gets lost or damaged). In this case, a user can recover their credentials from an encrypted backup stored in iCloud to a new iPhone. Although these are not a perfect solution, a user can still access their credentials.

### 6.6.2 Physical Security of Devices

Another fundamental assumption in Knock x Knock is that users' iPhones are physically secure. Because users tend to maintain physical proximity to their iPhones, and attackers have to physically come close to the devices, physically attacking (e.g., device thefts) is less likely to happen compared to remote attacks (e.g., dictionary attacks against users' online accounts). However, because the Knock x Knock stores all credentials in one device, an attacker may steal it to obtain credentials.

In the cases where an attacker steals a user's iPhone, the attacker would first have to circumvent the iPhone's lock features (i.e., typing 4-digit PIN or Touch ID) to access the Knock x Knock application if the lock features are enabled. After accessing the Knock x Knock application, the attacker would have to find out the user's master password to obtain credentials because accessing plaintext passwords always requires typing a master password. It would still be possible for the attacker to launch brute force attacks or educated guess attacks on the master password. However, while the attacker was trying to figure out the master password, the user could either remote wipe the iPhone using the feature provided by Apple, or recover Knock x Knock data from a encrypted backup stored in their iCloud on another iPhone and reset all of their passwords. For the services that support UIMP, users can let Knock x Knock reset their passwords because UIMP provides a password-update API. Thus, I

believe that expected risk to Knock x Knock as a result of iPhone theft is reasonably small.

Another device-theft scenario is that an attacker steals both a user's iPhone and Mac that has the UniAuth Proxy installed. In addition to the attack described above, in this scenario, the attacker could access credentials stored in the Quick tier if they can gain access to browsers on Mac and know the user's trusted locations (typically home and workplace). However, without knowing the user's master password, credentials stored in the Standard and Secure tiers cannot be accessed. Considering that credentials stored in the Quick tier are less critical ones, the expected risk would be acceptable, although further investigation into this type of attack is warranted.

As described, even if attackers steal physical devices, the expected risks are limited. Additionally, users can remote wipe their data or change their passwords to prevent further damage. Thus, I believe that the expected cost of device theft is low enough compared to the benefits of using Knock x Knock.

### 6.6.3 Long-Term Investigation of Knock x Knock Usage

Another interesting research topic would be how people use Knock x Knock in practice. There are many cases where security systems become vulnerable because people use them in insecure ways, e.g., reusing passwords. In Knock x Knock, one of the insecure practices that people may adopt is storing all credentials in the Quick tier to maximize usability. In my field study, all participants stored at least one credential in each tier; however people may ultimately decide to store everything in the Quick tier.

One potential solution to this issue is to limit tiers that a certain credential can be assigned. With UIMP, a service provider can specify a tier where a credential for the service should be stored. However, service providers could have incentive to store credentials for their services in less secure tiers to maximize usability so that people use their service more easily. A real world deployment and longer-term studies would be necessary to investigate how people and service providers use the Knock x Knock and UIMP in practice.

Furthermore, it would also be interesting to investigate how Knock x Knock affects users' password choices in the long run. The study reported in Chapter 3 showed

that participants were reluctant to use randomly generated passwords. However, in post-study interviews in the Knock x Knock field study, participants indicated willingness to use randomly generated passwords. Possibly this is due to the high availability of passwords stored in Knock x Knock (i.e., they can look up their passwords with their iPhones, which they almost always carry with them). However, long-term investigation is necessary to see whether people continue using randomly generated passwords or ultimately go back to manually choosing their own passwords.

## 6.7 Limitations

I believe that this work has provided novel insights on credential management tools and frameworks. However, it also has several limitations. For example, its field studies were not long enough to capture long-term effects on participants' credential management practices.

I also focused our evaluation on user acceptance rather than security, because a system with low desirability will be unlikely to be adopted, regardless of its security. As a result, our security analysis in this work is limited to users' subjective evaluations. More formal evaluation of its security is necessary. However, I still believe that our system addresses a class of security issues in password-based authentication such as weak and reused passwords, and that our work provides insightful design guidelines for practitioners, standards groups, and researchers.

Knock x Knock also needs better protection against phone theft. Currently, users can remotely wipe credentials and recover them to new phones from backups. However, for better protection, Knock x Knock could incorporate sensor-based anomaly detection algorithms to throttle access to the client.

## 6.8 Summary of Findings

In the near future, it is very likely that our authentication needs will be managed by a single smart device. The goal of the study in this chapter was to understand what this user experience might be like. Through two field studies, I evaluated a combination of several features that balance usability with security, looking at common uses as well as edge cases. The results of the studies showed:

- Participants were very positive about the concept of UniAuth
- Participants reported Knock x Knock improved both usability and security
- The combination of tiered access control and location-based access control worked very well
- Physical proximity and baseline availability of iPhones storing credentials strongly affect participants' trust on Knock x Knock
- Participants preferred to delegate periodical password updates to Knock x Knock.

# 7. Discussion

In this work, I investigated challenges in user authentication with a user-centered approach. Studies described in Chapter 3 revealed that people have strong needs for systems that manage their credentials. However, the studies also showed that existing credential management systems did not work well in practice. These systems tended not to be dependable enough, did not support current credential management practices well, or had a high cost for adoption. To address this gap, I took a human-centered approach in designing the UniAuth Framework. In the design process, I found interesting discrepancies between those who design the systems and those who use the systems. This chapter discusses these discrepancies. I believe these findings contribute to investigation and developments of credential management systems.

## 7.1 Physical Proximity Strongly Affects Perceived Security

When designing credential management systems, there are many possible places for storing credentials: cloud storage, our laptops and desktops, smartphones, and wearable devices. The results in this work suggest that storing information in a place physically close to users had a very positive effect on our participants' trust in the system. In Chapter 3, I reported that there were participants who prefer having physical memos or notepads because they could keep them physically close. When using Knock x Knock (Chapter 6), many participants commented that they felt safe

because their passwords were stored on their phones, which was always nearby. The fact that credentials are stored in a place physically close to a user seems to affect a user's subjective evaluation of their security in a positive way significantly.

Researchers rightfully focus more on underlying security mechanisms rather than *perceived* security. However, I argue that users are also influenced by perceptions of security, which might also include aspects of usability and utility, when making decisions to or not to adopt new security systems. In the context of credential management systems, from a technical perspective, one's credentials cannot be easily accessed as long as they are encrypted with a master password. However, people were still worried that attackers could access the credentials because of insufficient understanding of cryptography. In contrast, physical security is easier to understand. People believed that, for instance with Knock x Knock, because no one touches their phones and because credentials are transmitted using short-range wireless, it would be difficult for attackers to access their credentials. This finding suggests that improving perceived security could be as important as improving actual security to facilitate wider adoption of a new security system.

## 7.2 Guaranteeing Baseline Availability

People also preferred physical proximity in terms of its availability. In the study reported in Chapter 3, a participant commented, "I write down all my passwords in my notebook. I always carry the notebook. So I can look up them anytime." In Knock x Knock, participants liked the fact that, if they have their phones with batteries charged, they can access their passwords without relying on external infrastructures such as network connectivity. P6 said, "Your passwords are around you all the time. When you need them, they are already there." Users would have difficulties in accessing availability in some technical aspects of Knock x Knock, such as BLE connection and location awareness. However, they have clear ideas about availability of basic aspects, such as battery life, physical closeness, loss, or theft. Because the baseline availability of their credentials only depends on availability of these basic aspects, the participant felt comfortable with storing their credentials in a new system.

Another important design choice related to availability is whether a credential management system should allow manual access to credentials. Allowing it increases the risk of social engineering attacks. Thus, from security perspectives, it should not be allowed. However, my study (Chapter 6.4) revealed that people had strong preference for directly accessing passwords. For instance, a participant noted, "it's also important for me to have an option to see my passwords, just in case." Interestingly, they also admitted that they would probably not do so in practice. However, they wanted to have availability assurance concerning corner cases where you need direct access to their passwords such as the case where they have to share their passwords with others to complete their tasks (Chapter 3.1).

To assuage concerns, Knock x Knock provides a fallback case, letting users see their passwords on their phones after entering a master password. Simple fallback features like this, even if rarely used, may be useful in helping to convince participants of the reliability of new kinds of authenticators. As such, I recommend offering users backup options to guarantee baseline availability even in cases where complicated systems do not work as expected. As described, one tradeoff is that fallback features mean that social engineering attacks cannot be entirely avoided. However, making it harder to access passwords and presenting reminders of social engineering attacks can help mitigate potential attacks.

## 7.3 A Path Towards Better User Authentication

User authentication involves multiple stakeholders by it nature. It always has one doing authentication (e.g., a service provider) and one to be authenticated (e.g., a user). Sometimes, user authentication involves third parties helping the processes depending on architecture (e.g., OAuth and FIDO). Introducing a new authentication system often results in requiring modifications to each stakeholder *simultaneously*.

For instance, the current FIDO specifications [6] require a service provider to support FIDO protocol, a user to adopt FIDO clients, and a third party (e.g., Google) to provide a FIDO server. To make the user authentication with FIDO

work, we need all of these changes. Changing one or the other alone does not provide immediate benefit, which will likely make initial adoption very challenging.

In contrast, the UniAuth clients provide immediate benefit to users by supporting existing password-based authentication without server-side modifications. The studies in Chapter 6.5 demonstrated that participants were actually willing to adopt the UniAuth clients without server side support because of the immediate benefit. This implies that there would be reasonable initial adoption of UniAuth clients when it becomes publicly available. Once enough users adopt the clients, supporting UIMP provides benefit to service providers because they can streamline account management, such as account creation and password updates. Finally, after users become familiar with letting clients manage authentication and after service providers adopt UniAuth, we can replace passwords with stronger methods (e.g., private-public key pair), with minimum impact on the user experience.

It is easier to design an alternative to password-based authentication if we assume that multiple stakeholders adopt changes simultaneously. However, in practice, stakeholders will not adopt changes unless there is enough incentive to do so. Even if we develop a better user authentication system, it would provide little benefit if it is not actually adopted. Therefore, it is crucial to consider deployment strategies when designing and implementing user authentication systems.

# 8. Conclusion

Passwords are the most common form of user authentication today. When passwords were first introduced, it was an appropriate design choice to let users choose, memorize, and manage their credentials. However, because of the growing number of credentials that users have to manage, it is becoming increasingly infeasible for users to manage their credentials in a secure and usable manner. To address this challenge, a great deal of past work has investigated alternatives to password-based authentication. However, only a few alternatives have been actually adopted in limited contexts primarily because of their low deployability [22].

The goal of this dissertation was to take another look at user authentication systems with a human-centered approach. The body of this work consists of three phases of research. In the first phase, I started by understanding how people do user authentication in their daily lives and what kinds of tools they use to manage their credentials. This work revealed that people log into their accounts in a limited number of contexts (such as number of locations and computers used). Furthermore, it revealed that the adoption of password management tools is limited despite vital necessity of them, and that people are very concerned about the possible failure of password management tools that would prevent them from accessing their accounts.

Given the finding that users mostly authenticate themselves in a limited number of contexts, in the second phase of work, I first investigated whether I could utilize contextual information captured by smartphones' onboard sensors to improve user authentication to the smartphones. A series of studies demonstrated that participants

were quite positive about this concept and its implementation, particularly the configuration where the unlocking of smartphones did not require user authentication at trusted locations (e.g., their homes) while a PIN was required for unlocking at other places. These findings indicate that we can significantly improve usability of user authentication to smartphones by utilizing sensor data. However, the access control model used on smartphones was an *all-or-nothing* model, where users could access everything when smartphones were unlocked and could access nothing when they were locked. An all-or-nothing model is simple to implement from a system perspective; however, it was unclear whether the model satisfied users' access control needs well. The results of a subsequent user study clearly illustrated that the all-or-nothing model did not fully satisfy users' needs and that it would be quite useful to build a layered access control scheme by dividing applications into multiple categories where different layers required different security assurance to be accessed.

The last phase of this work designed, implemented, and evaluated the Unified Authentication Framework (UniAuth in short), a layer over existing password-based authentication system. The design of UniAuth was grounded in the findings of the first and second phases of the work. With its human-centered approach, UniAuth fit nicely with users' needs and improves both security and usability of user authentication while maintaining high deployability. Using the UniAuth Framework, UniAuth clients immediately conquer some of the common challenges in password-based authentication such as weak and reused passwords, and the Unified Identity Management Protocol provides a seamless transitional path toward a better authentication system. The results of three weeks of field study demonstrated the system's desirability in real world settings.

I believe that this work has made significant contribution to improve the security and usability of user authentication systems. However, there are still interesting research topics, such as user authentication to physical devices. Knock x Knock sheds light on the user experience of a smart device managing user authentication to a physical device (i.e., Mac laptop). Nevertheless, it is still unclear how the user experience would change when the smart device manages user authentication to variety of physical devices, and how the smart device should manage credentials for the devices. Another interesting question is whether we can further utilize sensor data in

user authentication to UniAuth client. In Knock x Knock, I chose to only use location information based on my previous findings that participants preferred a simple model in utilizing sensor data for authentication. However, there could be other ways of combining multiple sensor inputs for user authentication. Finally, it would be of great interest to make UniAuth publicly available to further evaluate whether the concept and its implementation can make real world impact.

In summary, this work took a new look at user authentication systems with a distinct, human-centered approach. Based on the results of my early studies investigating how people manage tasks related to user authentication, I developed and evaluated the Unified Authentication Framework. The results clearly demonstrate that UniAuth improves both the security and usability of user authentication. I believe that this work has made a significant contribution to the search for an ideal user authentication system.

# 9. REFERENCES

1. 1Password. https://agilebits.com/onepassword. Mar 8th, 2013.
2. Clef. https://getclef.com. May 18, 2015.
3. eToken. http://www.aladdin.com/etoken/. April 3rd, 2011.
4. Facebook Social Authentication. https://www.facebook.com/notes/facebook/a-continued-commitment-to-security/486790652130. July 9th, 2015.
5. Finger Print Verification Competition. https://biolab.csr.unibo.it/fvcongoing/UI/Form/IJCB2011.aspx. July 9th, 2015.
6. FIDO. https://fidoalliance.org/. Feb. 26, 2015.
7. KeePass. http://keepass.info/. Mar 8th, 2013.
8. KeyChain Access. http://www.apple.com/osx/apps/all.html#keychain. Mar 8th, 2013.
9. LastPass. https://lastpass.com/. Mar 8th, 2013.
10. LastPass Hacked. http://lifehacker.com/lastpass-hacked-time-to-change-your-master-password-1711463571, June 30th, 2015.
11. Lax Passwords Expose Quarter of PC Users to Theft. http://www.washingtonpost.com/wp-dyn/content/article/2007/10/09/AR2007100901896.html. April 4th, 2010

12. Microsoft .NET Gadgeteer,, http://www.netmf.com/gadgeteer/. May 31st, 2012.

13. Your Top 20 most frequently used passwords. http://www.tomshardware.com/news/imperva-rockyou-most-common-passwords,9486.html

14. OAuth. http://oauth.net. May 10th, 2015.

15. QRCode.com. http://www.qrcode.com/en/index.html. July 9th, 2015

16. RSA securID http://www.rsa.com/node.aspx?id=1156. July 9th, 2015.

17. When Security Gets in the Way http://jnd.org/dn.mss/when_security_gets_in_the_way.html. July 9th, 2015.

18. Wired.com. http://www.wired.com/2012/11/ff-mat- honan-password-hacker/all/. Sep. 17th, 2014.

19. Adams A., Sasse M. A. 1999. Users are not the enemy. Communication of the ACM.

20. Amini S., Lindqvist J., Hong I. J., Lin J., Sadeh N., and Toch E. 2011. Caché: Caching Location-Enhanced Content to Improve User Privacy. In *Proceedings of* the International Conference on Mobile Systems, Applications, and Services.

21. Amy K. Karlson, A.J. Bernheim Brush, and Stuart Schechter. 2009. Can I Borrow Your Phone?: Understanding Concerns When Sharing Mobile Phones. In *Proceedings of* the Annual Conference on Human Factors in Computing Systems.

22. Bardram J. E., Kjær R. E., Pedersen MØ. 2003. Context- Aware User Authentication Supporting Proximity-Based Login in Pervasive Computing. In *Proceeding of* the Conference on Ubiquitous Computing..

23. Bonneau, J., Herley, C., Oorschot, P.C.V., and Stajano, F. The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. IEEE, 553–567.

24. Brostoff S. and Sasse M. 2000. Are passfaces more usable than passwords: A field trial investigation. *In Proceedings of* HCI.

25. Burr W. E., Dodson D. F., and Polk. W. T. 2006 Electronic authentication guideline. Tech report, NIST

26. Buthpitiya S., Zhang Y., Dey A. and Griss M. 2011. n-gram Geo- Trace Modeling. In *Proceedings of* Pervasive Computing.

27. Chiasson S., Biddle R., and Oorschot P. V. 2007. A second look at the usability of click-based graphical passwords. In *Proceedings of* the Symposium on Usable Privacy and Security.

28. Corlett N.E., and Clark S.T. 1995. The Ergonomics of Workspaces and Machines: A Design Manual. CRC Press. 1995.

29. Corner M. D. and Noble B. D. 2003. Protecting applications with transient authentication. In *Proceedings of* the International Conference on Mobile Systems, Applications, and Services.

30. Cranshaw J, Toch E., Hong J. I., Kittur A., and Sadeh N. 2010. Bridging the gap between physical location and online social networks. In *Proceeding of* the Conference on Ubiquitous Computing.

31. Das, S., Kim, H.J., Dabbish, L., and Hong, J. 2014. The Effect of Social Influence on Security Sensitivity. In *Proceedings of* the Symposium on Usable Privacy and Security.

32. Dhamija R. and Tygar J. 2005. The battle against phishing: Dynamic security skins. In *Proceedings of* the Symposium on Usable Privacy and Security.

33. Dirik A., Memon N., and Birget J. C. 2007. Modeling user choice in the PassPoints graphical password scheme. In *Proceedings of* the Symposium on Usable Privacy and Security.

34. Dourish P., Grinter E., Flor J. D., Joseph M. 2004. Security in the wild: user strategies for managing security as an everyday, practical problem. Personal and Ubiquitous Computing. vol. 8 (6)

35. Egelman S., Cranor L. F., Hong J. 2008. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of* the SIGCHI Conference on Human Factors in Computing Systems.

36. Everitt K., Bragin T., Fogarty J., and Kohno T. 2009. A comprehensive study of frequency, interference, and training of multiple graphical passwords. In *Proceedings of* the SIGCHI Conference on Human Factors in Computing Systems.

37. Fisher I., Kuo C., Huang L., and Frank M. 2012. Smartphones: Not Smart Enough? In *Proceedings of* Workshop on Security and Privacy in Smartphones and Mobile Devices.

38. Florencio D., Herley C., Coskun B. 2007. Do strong web passwords accomplish anything? In *Proceedings of* USENIX Hot Topics in Security.

39. Florencio D., Herley C. 2007. A large-scale study of web password habits. In Proc. of the International World Wide Web Conference.

40. Gafurov, D. 2007. A Survey of Biometric Gait Recognition: Approaches, Security and Challenges, In *Proceedings of* NIK-2007 Conference.

41. Gartner. Automated password resets can cut it service desk costs. 2004.

42. Gaw S. and Felten E. 2006. Password management strategies for online accounts. In *Proceedings of* the Symposium on Usable Privacy and Security.

43. Gehringer E. F. 2002. Choosing passwords: security and human factors. In *Proceedings of* Technology and Society.

44. Hallinan J. T. 2009. Why We Make Mistakes. Broadway.

45. Hayashi E., Dhamija R., Christin N., and Perrig A. 2008. Use Your Illusion: Secure Authentication Usable Anywhere. In *Proceedings of* the Symposium on Usable Privacy and Security.

46. Hayashi E. and Hong J. 2011 A diary study of password usage in daily life. In *Proceedings of* the SIGCHI Conference on Human Factors in Computing Systems.

47. Hayashi E., Pendleton B., Ozenc F., and Hong J. 2012. WebTicket: account management using printable tokens. In *Proceedings of* the SIGCHI Conference on Human Factors in Computing Systems.

48. Hayashi, E., Riva, O., Strauss, K., Brush, A.J.B., and Schechter, S. 2012. Goldilocks and the Two Mobile Devices: Going beyond All-or-Nothing Access to a Device's Applications. In *Proceedings of* the Symposium on Usable Privacy and Security.

49. Hayashi, E., Das, S., Amini, S., Hong, J., & Oakley, I. 2013. CASA: Context-Aware Scalable Authentication. In *Proceedings of* the Symposium on Usable Privacy and Security.

50. Hayashi E., and Hong J. 2013. "It's Hidden in My Computer": Exploring Account Management Tools and Behaviors. CyLab Technical Note.

51. Hayashi E., and Hong J. 2015. Knock x Knock: The Design and Evaluation of a Unified Authentication Management System. In *Proceeding of* the Joint Conference on Pervasive and Ubiquitous Computing.

52. Heusch, G., and Marcel, S. A novel statistical generative model dedicated to face recognition. Image and Vision Computing. vol. 28, Issue 1, 2010, 101-110.

53. Hulsebosch J. R., Salden H. A., Bargh S. M., Ebben P. W. G, and Reitsma J. 2005. Context sensitive access control. In *Proceedings of* the Symposium on Access Control Models and Technologies.

54. Inglesant P. and Sasse M. A. 2010. The true cost of unusable password policies: password use in the wild. In *Proceedings of* the SIGCHI Conference on Human Factors in Computing Systems.

55. Jakobsson M., Shi E., Golle P., and Chow R. 2009. Implicit authentication for mobile devices. In *Proceedings of* USENIX.

56. Jain, A., Nandakumar, K., and Ross, A. 2005. Score normalization in multimodal biometric systems, Pattern Recognition, vol. 38, Issue 12, 2005, 2270-2285.

57. Jain, A., Bolle, M.R., and Pankanti, S. 2005. Biometrics: Personal Identification in Networked Society. Springer.

58. Kalamandeen A., Scannell A., Lara E., Sheth A. and LaMarca A. 2010. Ensemble: cooperative proximity-based authentication. In *Proceedings of* the International Conference on Mobile Systems, Applications, and Services.

59. Kaye J. J. 2011. Self-reported password sharing strategies. In *Proceeding of* the Joint Conference on Pervasive and Ubiquitous Computing.

60. Jiayang, L., Lin, Z., Jehan, W., and Venu V. 2009. User evaluation of lightweight user authentication with a single tri-axis accelerometer. In *Proceedings* of the International Conference on Human-Computer Interaction with Mobile Devices and Services.

61. Kam L., Konrad, J., Ishwar, P. 2012. Towards Gesture-Based User Authentication, *Advanced Video and Signal-Based Surveillance,* 282-287.

62. Klein D. V. 1990. ”foiling the cracker”: A survey of, and improvements to, password security. In *Proceedings of* USENIX Security.

63. Klemmer S., Newman M., and Farrell R. 2001. The designers’ outpost: a tangible interface for collaborative web site. In *Proceedings of* the Symposium on User Interface Software and Technology.

64. Komanduri S., Shay R., Kelley P. G., Mazurek L. M., Bauer L., Christin N., Cranor L. F., and Egelman S. 2011. Of passwords and people: measuring the effect of password-composition policies. In *Proceeding of* the Joint Conference on Pervasive and Ubiquitous Computing.

65. Kuo C., Romanosky S., Cranor L. 2006. Human selection of mnemonic phrase-based passwords. In *Proceedings of* the Symposium on Usable Privacy and Security.

66. S. L. 1967. Learning 10,000 pictures. Quarterly Journal of Experimental Psychology.

67. Leyvand, T., Meekhof, C., Wei, Y., Sun, J. and Guo, B. 2011. Kinect identity: Technology and experience. IEEE Computer Society. Computer Magaznine, 94–96.

68. MacKay W. 1999. Is paper safer? The role of paper flight strips in air traffic control. ACM Transactions on Computer-Human Interaction.

69. McCune J., Perrig A., and Reiter M. 2005. Seeing-is-believing: Using camera phones for human-verifiable authentication. In IEEE Security and Privacy.

70. McGee D., Cohen P., Wesson R., and Horman S. 2002. Comparing paper and tangible, multimodal tools. In *Proceeding of* the Joint Conference on Pervasive and Ubiquitous Computing.

71. Moran T., Saund E., Melle W. V., Gujar A., Fishkin K., and Harrison B. 1999. Design and technology for collaborage: collaborative collages of information on physical walls. In *Proceedings of* the Symposium on User Interface Software and Technology.

72. Mäntyjärvi, J., Lindholm, M., Vildjiounaite, E., Makela, S.M., and Ailisto, H. 2005. Identifying users of portable devices from gait pattern with accelerometers. In *Proceedings of* the IEEE International Conference on Acoustic, Speech, and Signal Processing. vol. 2, 2005, 973-976.

73. Maler, E. and Reed, D. 2008. The Venn of identity: Options and issues in federated identity management. IEEE Security and Privacy, 6:16–23, 2008.

74. Maltoni, D., Maio, D., Jain, K.A., and Prabhakar, S. 2005. Handbook of Fingerprint Recognition. Springer.

75. Maurer, M. E., Waxenberger, R., and Hausen, D. 2012. BroAuth: evaluating different levels of visual feedback for 3D gesture-based authentication. In *Proceedgins of* the International Working Conference on Advanced Visual Interfaces. 737-740.

76. Mayrhofer, R. and Gellersen, H. 2009. Shake well before use: two implementations for implicit context authentication. *IEEE Transactions on Mobile Computing*, vol.8, no.6, 792-806.

77. Napa, S.B., Kowsar, A., Katherine, I., and Nasir, M. 2010. Biometric-rich gestures: a novel approach to authentication on multi-touch devices. In *Proceeding of* the Joint Conference on Pervasive and Ubiquitous Computing..

78. Nelson L., Ichimura S., Pedersen E., and Adams L. 1999. Palette: a paper interface for giving presentations. In *Proceeding of* the Joint Conference on Pervasive and Ubiquitous Computing.

79. Paivio A. and Rogers T. 1968. Why are pictures easier to recall than words? Psychonomic Science (1968).

80. Parno B., Cuo C. and Perrig A. 2006. PhoolprofPhishing Prevention. In *Proceedings of* the Financial Cryptography and Data Security.

81. Patel, S.N., Pierce, J.S., and Abowd, G.D. 2004. A gesture-based authentication scheme for untrusted public terminals. In *Proceedings of* the Symposium on User Interface Software and Technology, 157–160.

82. Peacock, A., Ke, X., and Wilkerson, M. 2004. Typing patterns: A key to user identification. IEEE Security and Privacy, vol. 2, Issue 5, 2004, 40-47.

83. Pheasant, S., and Haslegrave, M.C. 2005. Bodyspace: Authropometry, Ergonomics and the Design of Work. CRC Press.

84. Riva, O., Qin, C., Strauss, K., Lymberopoulos, D. 2012. Progressive authentication: deciding when to authenticate on mobile phones. *In Proceedings of USENIX*.

85. Ross B., Jackson C., Miyake N., Boneh D., and Mitchell J. 2005. Stronger password authentication using browser extensions. In *Proceedings of* the USENIX Security.

86. Sasse M., Brostoff S., and Weirich D. 2001. Transforming the 'weakest link'— a human/computer interaction approach to usable and effective security. BT technology journal.

87. Schmidt, D., Chong, M.K., and Gellersen, H. 2010. HandsDown: hand-contour-based user identification for interactive surfaces. In *Proceedings of NordiCHI*, 2010, 432–441.

88. Shay R., Komanduri S., Kelley P. G., Leon P. G., Mazurek M. L., Bauer L., Christin N., Cranor L. F. 2010. Encountering stronger password requirements: user attitudes and behaviors. In *Proceedings of* the Symposium on Usable Privacy and Security.

89. Singh S., Cabraal A., Demosthenous C., Astbrink G., and Furlong M. 2007 Password sharing: implications for security design based on social practice. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.*

90. Singh S., Cabraal A., Demosthenous C., Astbrink G., and Furlong M. 2007. Password sharing: implications for security design based on social practice. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

91. Snelick, R., Uludag, U., Mink, A., Indovina, M., and Jain, A. Large-scale evaluation of multimodal biometric authentication using state-of-the-art systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol.27, no.3, 2005, 450-455.

92. Seifert J., De Luca A., Conradi B. and Hussmann H. 2010. TreasurePhone: Context-Sensitive User Data Protection on Mobile Phones. In *Proc. of* Pervasive Computing.

93. Stajano F. 2004. One user, many hats; and, sometimes, nohat–towards a secure yet usable PDA. In *Proceeding of* Security Protocols Workshop. 51-64.

94. Supriya Singh, Anuja Cabraal, Catherine Demosthenous, Gunela Astbrink, and Michele Furlong. 2007. Password sharing: implications for security design based on social practice. In *Proceeding of* the Joint Conference on Pervasive and Ubiquitous Computing.

95. Tian, J., Qu, C., Xu, W., and Wang, S. 2013. KinWrite: Handwriting-Based Authentication Using Kinect. In *Proc. of* Annual Network & Distributed System Security Symposium.

96. Walsh K. R., Ives B., Schneider H. 2004. The Domino Effect of Password Reuse. Communication of the ACM.

97. Wan, V., and Renals, S. Speaker verification using sequence discriminant support vector machines. *IEEE Transactions on Speech and Audio Processing.* vol.13, no.2, 2005, 203- 210

98. Whitten A. and Tygar J. 1999. Why Johnny can't encrypt. In *Proceedings of* USENIX Security.

99. Wiedenbeck S., Waters J., Birget J., and Brodskiy A. 2005. Passpoints: Design and longitudinal evaluation of a graphical password system. International Journal of Human-Computer Studies.

100. Woodward, J.D. 1997. Biometrics: Privacy's Foe or Privacy's Friend?.

Proceedings of the IEEE , vol.85, no.9, pp.1480- 1492, Sep 1997.

101. Yan J., Blackwell A., Anderson R., and Grant A. 2004. Password memorability and security: Empirical results. IEEE Security & Privacy (2004) vol. 2 (5) pp. 25-31

102. Yee K., Sitaker K. 2006. PassPet: convenient password management and phishing protection. In *Proceedings of* the Symposium on Usable Privacy and Security.

*(More dramatic effect)*

# APPENDIX A UIMP Specification

## A.1 Unified Identity Management Protocol

Universal Identity Management Protocol (UIMP) is a set of APIs that UniAuth-compliant services should support. UniAuth clients manage users' identities stored in the services through the APIs. Although eventually UIMP will be defined and implemented to support both online services and local devices, an initial version of UIMP is designed as a set of RESTful APIs for web services. This is because majority of existing services that require identity management are web services. Later, this API set will be ported another form which is suitable for communication with local devices, such as function calls over Bluetooth or NFC.

## A.2 List of APIs

UIMP consists of four categories of APIs: Information, Authentication, Account, and Notification. Its authentication mechanism primarily follows that of OAuth. In terms of authentication mechanism, one difference between UIMP and OAuth is whether they delegate user authentication to third parties. While OAuth delegates it to OAuth provider (e.g., Google, Facebook, and Twitter), UIMP does not delegate it to third parties. This simplifies its authentication mechanism and makes it more similar to existing password-based authentication mechanism.

A result of API call (i.e., success or error) is returned by a HTTP response code (see A.9). When a call succeeds, a HTTP message body may contain return values in JSON format. When a call fails, a HTTP message body may contain an error code and an error message.

The following table shows a list of UIMP APIs. For a service to be compatible with UIMP, the service should support `authentication/login` API. Supporting other APIs is optional.

| Method | API | Description |
|---|---|---|
| **Information APIs** | | |
| GET | information/service | Obtain information about service |
| GET | information/authentication_policy | Obtain authentication policy, typically a password composition policy |
| GET | information/required_user_information | Obtain a list of information required to create an account |
| **Authentication APIs** | | |
| POST | authentication/login | Provide a user ID and a password. Then, API returns a html source of a first page after login. This function is equivalent to typing a user ID and a password, and clicking a login button |
| POST | authentication/tokens | Obtain service token by providing a user ID and a password, which is used to call APIs that require authentication |
| DELETE | authentication/tokens/{id} | Revoke a specified service token |
| GET | authentication/tokens/ | Obtain a list of currently valid service tokens |
| **Account APIs** | | |
| POST | account | Create an account |
| DELETE | account | Revoke an account |
| PUT | account | Update account information |
| PUT | account/password | Change a password |
| GET | account/password/onetime | Obtain a onetime password |
| POST | account/password/recover | Request a password recovery |
| **Notification APIs** | | |
| POST | notification/entries | Create a notification entry that defines conditions and media to send out notifications |
| DELETE | notification/entries/{id} | Delete an existing notification entry |
| GET | notification/entries | Obtain a list of notification entries. |

# A.3 Typical Use Cases

This is an example illustrating how a browser interacts with a UIMP-compliant web service. A UIMP-compliant web service (e.g., Amazon.com) indicates that the service support UIMP by including a meta tag in their web pages and a link tag indicating URI for UIMP. When UniAuth-compliant browsers (including browsers with UniAuth extensions) read the tags, they interact with web services through UIMP.

```
<meta name="uniauth" content="version=1.0">
<link ref="uimp.apis" href="http://amazon.com/uimp/">
```

One typical example of interactions between the browsers and web services is user authentication. If an account for this service is stored in a UniAuth client, the browser display a dialog asking whether a user wants to log into his account (see Figure 1). When he clicks the login button, the browser obtains a user ID and a password for the account from the UniAuth client and posts the following JSON to `http://amazon.com/uimp/authentication/login`.

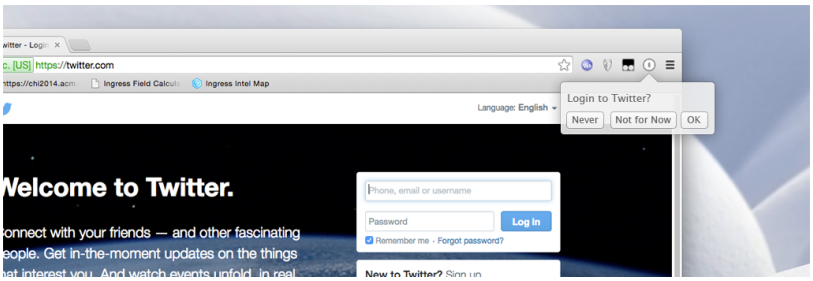



**Figure A.1. An example of login dialog with UIMP-compliant browser**

```
{
   "userid" : "john@gmail.com",
   "password" : "hsagion2%dAFga!faiu2314"
}
```

In practice, websites have many hidden input parameters in their login forms. These parameters are passed to the websites when users log into the services and are used for traffic analyses. If a service wants to send additional parameters in UIMP, the service can include a status parameter in a meta tag. Then, a UIMP-compliant browser sends the status along with arguments required for APIs.

```
<meta name="uniauth" content="version=1.0,status="pageid=mainpage">
```

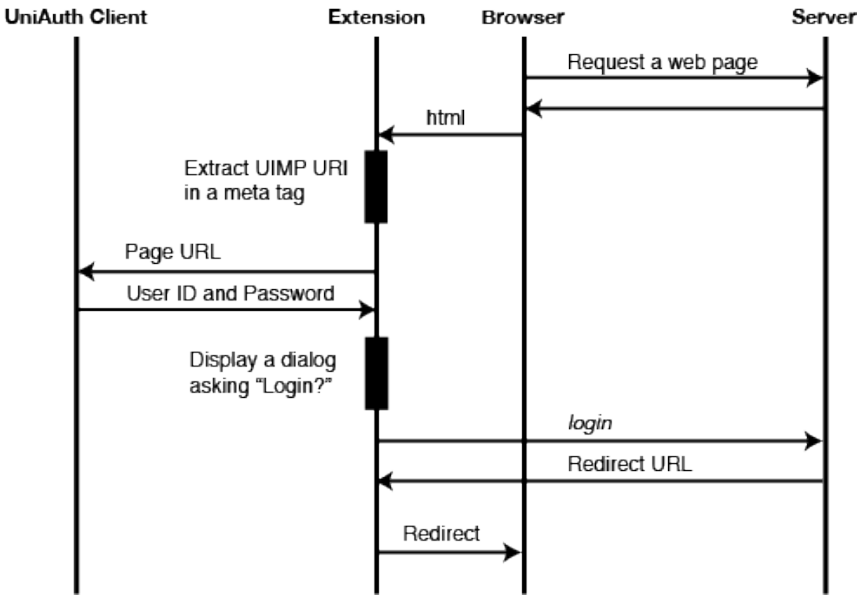
```
{
   "userid" : "john@gmail.com",
   "password" : "hsagion2%dAFga!faiu2314"
   "status" : "pageid=mainpage"
}
```

# A.4 Communication Sequence Examples

This section describes examples of communication sequences using UIMP APIs to complete typical tasks such as authentication, account creation, and password update. In the following examples, we assume that a browser is not supporting UIMP. Thus, there is a browser extension that makes the browser UIMP compatible. When a browser supports UIMP, the browser will handle the extension's role also.

**Authentication to a service with minimal UIMP support**

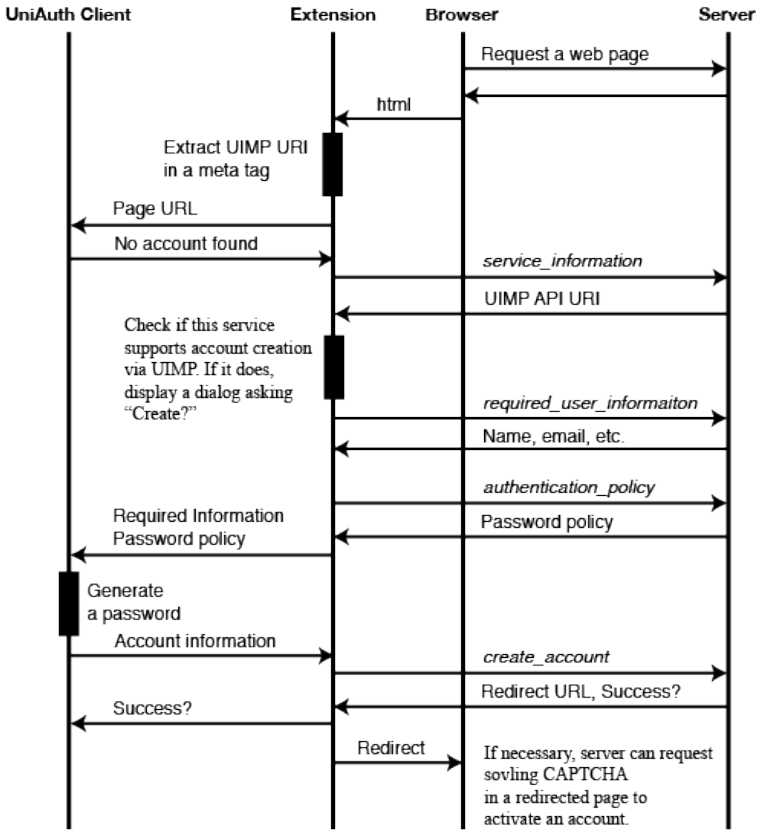
The following diagram illustrates a communication sequence where a UniAuth client processes user authentication with a web service that provide minimal UIMP support (i.e., only supporting *login* API).




When a user opens a web page, UniAuth extension tries to extract a UIMP meta tag and UIMP URI in the tag. If the extension finds these (which means the web service supports UIMP), the extension request UniAuth client to send login credentials. After the user indicates that he wants to login to the web service, the extension sends the login credentials to the web service, then, redirect to a URL received from the service after completing a user authentication on the server side.

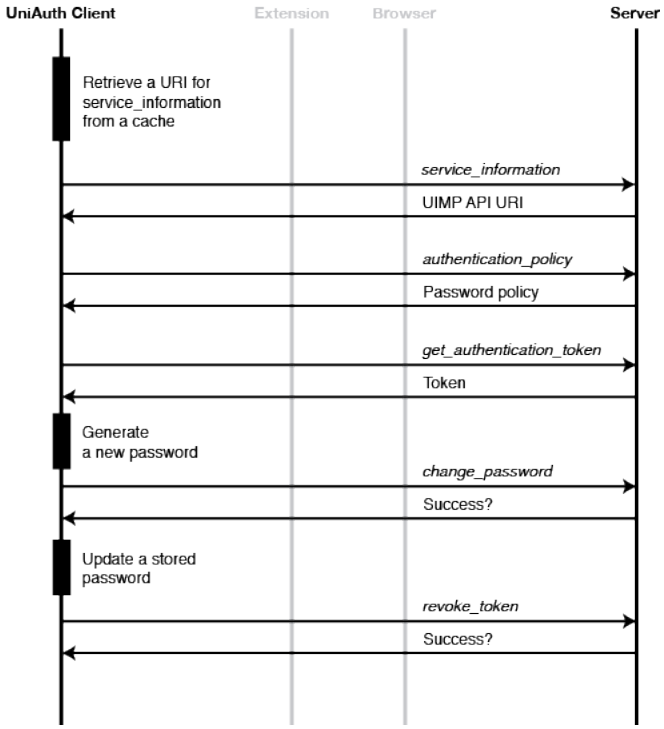
### Account creation

The following diagram illustrates a use case where a user creates an account via UIMP.



### Password Update

As an example of tasks than can be managed via UIMP other than user authentication, the following diagram illustrates a sequence where UniAuth client updates a password without a user's intervention. In Knock x Knock, which is an implementation of UniAuth client, a user can configure the client to change passwords periodically. This is how Knock x Knock updates passwords automatically.



# A.5 Information APIs

Information APIs provide information about a service. Because these APIs only handled public information, the API can be called over HTTP without authentication.

## Service Information

### URI

information/service

### Method

GET

### Description

This API returns information about a service

### Arguments

None

### Return Values

| Required | | |
|---|---|---|
| url | string | A URL of a service (e.g., for Amazon, this is http://amazon.com) |
| name | string | A name of the service |
| uimp_version | string | Supported UIMP version |
| uimp_uris | array | A array of supported UIMP APIs an their URIs |
| Optional | | |
| description | string | A description of a service shown to users when accessing this service on UniAuth clients |
| icon_url | string | A URL of an icon image |
| security_level | int | A default security level (1 to 5 where 1 means high and 5 means low) |
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

uimp_uris is an array consisting of dictionaries below.

| Required | | |
|---|---|---|
| api_name | string | A name of supported UIMP API |
| Optional | | |
| api_uri | string | A URI of a UIMP API. If omitted, default URI is used. |

### Note

The security level parameter specifies a recommended security level in which an account for this service will be placed in UniAuth client (if the client supports a layered security model). In Knock x Knock, an account with a security level 1 and 2 will be placed in Secure category on creation, ones with level 3 and 4 will be placed in Standard, and ones with level 5 will be placed in Quick. Please note that users can still place the accounts in a different security tier manually if they want.

## Requested User Information

### URI

information/requested_user_information

### Method

GET

### Description

This API returns an array containing information about user information required to create an account in this service. If a service support multiple types of accounts and these accounts require different information, this function returns an array.

### Arguments

None

### Return Values

| Required | | |
|---|---|---|
| required_user_information | array | An array containing required items |
| optional_user_information | array | An array containing optionally required items |
| required_non_default_information | array | An array containing dictionaries of required non-default items |
| optional_non_default_information | array | An array containing dictionaries of optional non default items |
| Optional | | |
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

| Store these strings in the array when they are required or optionally required | | |
|---|---|---|
| client ID | string | |
| email | string | |
| phone_number | string | |
| first_name | string | |
| middle_name | string | |
| last_name | string | |
| address1 | string | |
| address2 | string | |
| state | string | |
| zip_code | string | |
| country | string | |
| credit_card_company | string | |
| credit_card_holder | string | |
| credit_card_number | string | |
| credit_card_expiration_year | string | |
| credit_card_expiration_month | string | |
| credit_card_security_code | string | |

| Dictionary structure for non-default items | | |
|---|---|---|
| name | string | A string that should be used as a variable name for this item when calling create_account API |
| type | string | A type of this item (string/int/boolean) |
| description | string | A human-readable description of this item. This description will be displayed in a UniAuth client when a user creates an account. |

## Information/Authentication Policy

### URI

information/authentication_policy

### Method

GET

### Description

This API returns authentication policies that include what scheme can be used for user authentication for a service and details of the scheme. In UIMP 1.0, it only support password as an authentication scheme.

### Arguments

None

### Return Values

| Required | | |
|---|---|---|
| scheme | string | supported authentication scheme. (In UIMP 1.0, this should be "password") |
| policy | dictionary | A dictionary that contains policies for the scheme |

### Policy dictionary for password

All values are optional. If omitted, default values are used.

| Optional | | |
|---|---|---|
| description | string | Human readable descriptions of the password composition policy |
| minimum_length | int | A minimum length of a password (default 8) |
| maximum_length | int | A maximum length of a password (default 32) |
| is_lower_case_allowed | boolean | Whether use of lower cases are allowed in a password (default true) |
| is_upper_case_allowd | boolean | Whether use of upper cases are allowed in a password (default true) |
| in_number_allowed | boolean | Whether use of numbers are allowed in a password (default true) |
| supported_special_characters | array | An array of special characters that can be used in a password (default all special characters) |
| minimum_lower_cases | int | A minimum number of lower case letters required in a password (default 1) |
| minimum_upper_cases | int | A minimum number of upper case letters required in a password (default 1) |
| minimum_numbers | int | A minimum number of numbers required in a password (default 1) |
| minimum_special_characters | int | A minimum number of special characters required in a password (default 1) |
| valid_for_days | int | A number of days in which a password valid. After this number of days, a user has to change his password. (default 0, meaning that update is not enforced) |
| prohibited_pattern | string | A regular expression that represent prohibited pattern in passwords such as "*[0-9]" meaning no digit at the end of password. |

# A.6 Authentication APIs

Authentication APIs handle user authentication related tasks. To support UIMP, a service should support POST method to login URL, which is essentially equal to filling a user ID and a password in a standard login form and clicking a login button. This allows automatic user authentication with a UniAuth client.

Other APIs allows UniAuth client to manage authentication tokens. The tokens should be included in HTTP headers when calling APIs that need user authentication. The authentication architecture using access tokens follows one in OAuth 2.0.

## Login

### Default URI

authentication/login

### Method

POST

### Description

This API processed user authentication based on a user ID and a password. Calling this API is equal to filling a login form with a user ID field and a password field and clicking a login button. Different from the get_token API, this API does not return an authentication token. Authentication status should be managed by existing schemes such as session controls. After authentication this function returns a redirect URL.

### Arguments

| Required | | |
|---|---|---|
| user_id | string | A user id |
| password | string | A password |
| Optional | | |
| client_id | string | A UniAuth client ID |

A client ID is a random string that a UniAuth client MAY configure when creating an account using UIMP. This is a shared secret between a service and a UniAuth client. Users SHOULD NOT be able to access client IDs. By providing a client ID, a UniAuth client can prove that a client calling this API is a client that a legitimate user is using.

User authentication is solely based on a pair of a user ID and a password. Thus, even if a client ID is wrong (or not provided), it does not affect the result of user authentication.

A typical use case of the client ID is to filter notifications. A user can set a service to send login notifications only when a client ID is wrong (or not provided) i.e., someone accessed his account without using his UniAuth client. Another use case could be to limit accesses to an account only via UniAuth client and to prevent accesses through manually typing a user ID and a password.

### Return Values

If a use authentication is successful, the API returns a redirect URL. Otherwise, it returns error code and message.

| Required | | |
|---|---|---|
| redirect_url | string | A URL for users to be redirected |
| Optional | | |
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

## Get Access Token

### Default URI

authentication/tokens

### Method

GET

### Description

This API does user authentication based on a user ID and a password, and returns an access token that can be used to call other UIMP APIs.

### Arguments

| Required | | |
|---|---|---|
| user_id | string | A user id |
| password | string | A password |
| Optional | | |
| client_id | string | A UniAuth client ID |

### Return Values

If a use authentication is successful, the API returns an access token. Otherwise, it returns error code and message.

| Required | | |
|---|---|---|
| access_token | string | access token |
| expires_in | int | A number of seconds in which the access token is valid |
| Optional | | |
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

## Revoke Access Token

### Default URI

authentication/tokens

### Method

DELETE

### Description

This API revokes an access token provided in a HTTP header.

### Arguments

A valid access token should be included in a HTTP header.

### Return Values

| Required | | |
|---|---|---|
| result | string | result |
| Optional | | |
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

## Revoke Access Token with ID

### Default URI

authentication/tokens/{id}

### Method

DELETE

### Description

This API revokes an existing access token with an ID specified in URL.

### Arguments

A valid access token should be included in a HTTP header.

### Return Values

| Required | | |
|---|---|---|
| result | string | result |
| Optional | | |
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

## Get Access Token List

### Default URI

authentication/tokens

### Method

GET

### Description

This API returns a list of valid access tokens.

### Arguments

A valid access token should be included in a HTTP header.

### Return Values

| Required | | |
|---|---|---|
| access_token_list | array | An array of currently valid access tokens |
| Optional | | |
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

### Example

This is an example of return values.

```
{"access_token_list":
    [
          {"id":91,"access_token":"ksajhkhsj12nasgoi9243"},
          {"id":54,"access_token":"ks14&510gh10*65$%k"},
          {"id":134,"access_token":"pa8)1596jngkgt)-1"}
    ]
}
```

# A.7 Account APIs

APIs in the account category handle account creation, update, and revocation including password update and reset.

## Create Account

### Default URI

account

### Method

POST

### Description

This API creates an account for a service with given information. A client can obtain what information is required to create an account using Required User Information API.

### Arguments

This API does not require an access token.

| Required |
|---|
| A dictionary containing required information |

### Return Values

| Optional | | |
|---|---|---|
| redirect_url | string | A service can provide a redirect URL where it can do additional processing such as CAPTCHA to activate the account. |
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

## Change Password

### Default URI

account/password

### Method

PUT

### Description

This API changes a password.

### Arguments

A valid access token should be included in a HTTP header.

| Required | | |
|---|---|---|
| password | string | A new password |

| Required | | |
|---|---|---|
| userid | string | A user ID |
| old_password | string | A current password |
| new_password | string | A new password |

### Return Values

| Optional | | |
|---|---|---|
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

### Note

For this account/change_password API, two different sets of arguments can be passed. The first one uses access_token for user authentication, and the second one uses a pair of a user ID and an old password for user authentication. The second set of arguments is included in this specification to let service support this API without implementing access token handling.

## Request Password Recovery

### Default URI

account/password/recover

### Method

POST

### Description

This API requests a service to recover an access to a user's account by resetting password. This typically happens when a user lost his password. Upon a request, a service returns an instruction about how to reset a password.

### Arguments

None

### Return Values

| Required | | |
|---|---|---|
| redirect_url | string | A URL to be redirected. Typically, the URL contains information about password reset. |
| Optional | | |
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

## Update Account Information

### Default URI

account

### Method

PUT

### Description

This API updates user information in an existing account.

### Arguments

A valid access token should be included in a HTTP header.

| Required | | |
|---|---|---|
| user_information | dictionary | A dictionary containing user information to be updated |

### Return Values

| Optional | | |
|---|---|---|
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

## Onetime Password

### Default URI

account/password/onetime

### Method

GET

### Description

This API returns a onetime password that can be used to log into a service on untrusted computer and/or to let somebody access an account temporally.

### Arguments

A valid access token should be included in a HTTP header.

| Required | | |
|---|---|---|
| valid_for | int | A number of seconds in which issued tokens are valid for |

### Return Values

| Required | | |
|---|---|---|
| onetime_password | string | Onetime password |
| expires_in | int | A number of seconds in which the access token is valid |
| Optional | | |
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

# A.8 Notification APIs

## Create Notification Entry

### Default URI

notification/entries

### Method

POST

### Description

Calling this API creates a notification entry that defines conditions and medium for notifications.

### Arguments

A valid access token should be included in a HTTP header.

| Required | | |
|---|---|---|
| event | string | An event to send a notification. |
| medium_type | string | A medium used to send notifications (email/sms) |
| medium_information | string | Either an email address or a phone number to send a notification |

### Return Values

| Optional | | |
|---|---|---|
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

### Note

The following table contains events that can be specified in the event parameter.

| Required | |
|---|---|
| login_success | When a person log into this account |
| login_success_without_client_id | When a person log into this account without providing a registered client ID |
| login_failure | When a person tried to log into this account and fails |
| get_access_token_success | When get access token is called |
| get_access_token_success_without_client_id | When get access token is called without providing a registered client ID |
| get_access_token_failure | When get access token is called and authentication fails |
| <UIMP API name> | When <UIMP API> is called |
| <UIMP API name>_without_access_token | When <UIMP API> is called without a valid access token |

## Delete Notification Entry

### Default URI

notification/entries/{id}

### Method

DELETE

### Description

Calling this API delete an notification entry that matches to the specified properties.

### Arguments

A valid access token should be included in a HTTP header.

### Return Values

| Optional | | |
|---|---|---|
| error_code | int | Error code (see Section X for details) |
| error_description | string | Human-readable description of the error |

## Get Notification Entry List

### Default URI

notification/entries

### Method

GET

### Description

This API returns a list of existing notification entries.

### Arguments

A valid access token should be included in a HTTP header.

### Return Values

| Required | | |
|---|---|---|
| notification_entry_list | array | An array of notification entry dictionaries |

The list contains IDs of the entries and the configurations for the entries.

Example:

```
[
	{
		"id":"0",
		"event":"login_success",
		"medium":"email",
		"medium_information":"example@example.com"
	},
	{
		"id":"1",
		"event":"login_success_without_client_id",
		"medium":"sms",
		"medium_information":"0123456789"
	},
	{
		"id":"2",
		"event":"login_failure",
		"medium":"email",
		"medium_information":"example@example.com"
	}
]
```

## A.9 Error Codes

UIMP APIs return error information in using HTTP response codes. The APIs can optionally return detailed error information in response bodies as JSON format data.

| Response code | Response message | Description |
|---|---|---|
| 200 | OK | Processing was successful |
| 201 | Created | New resource was created successfully |
| 202 | Accepted | Request was accepted normally |
| 400 | Bad Request | Request data contains an invalid value |
| 401 | Unauthorized | Authorization failed |
| 404 | Not Found | Resource does not exist |
| 405 | Method Not Allowed | Method is not allowed |
| 500 | Internal Server Error | Processing failed because of an API problem |
| 503 | Service Unavailable | API cannot be accessed temporarily |