A WORLD WITH MANY AUTHENTICATION SCHEMES

by
Alain Forget

A thesis submitted to
the Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY

School of Computer Science

at

CARLETON UNIVERSITY

Ottawa, Ontario
November, 2012

Abstract

Usability and security challenges with standard text passwords have led researchers and professionals to consider alternative authentication schemes. This thesis explores the various challenges inherent in supporting a practical reality of authentication scheme diversity. We address these challenges by proposing the following solutions aimed at providing users with a usable and secure authentication experience through alternative schemes.

We developed a framework for developers, researchers, professionals, and users to identify and compare the user-centred features that may be supported by authentication schemes. We also performed empirical studies on two novel authentication schemes. We demonstrate that our text-based password scheme, Persuasive Text Passwords, can influence users to create more secure passwords that are memorable. We also show that our gaze-based graphical password scheme, Cued Gaze-Points, is usable and may offer resistance against shoulder-surfing attacks at the cost of reduced resistance against password guessing attacks. Furthermore, we built and user tested four different tutorial formats to determine which is most effective at teaching users a novel authentication scheme. Finally, we designed Choose Your Own Authentication (CYOA); an architecture that enables users to choose an authentication scheme amongst several available alternatives. CYOA provides numerous benefits to end-users, as well as security administrators who manage the back-end portion of the authentication, and researchers who develop novel authentication technologies. Finally, we discuss the design of a usability study of CYOA, to evaluate how users will leverage and cope with the power of choice between authentication schemes.

Acknowledgements

First and foremost, I would like to thank my supervisors Sonia Chiasson and Robert Biddle for their guidance, insights, compassion, and limitless patience and faith in me. This work would not have been possible without either of them. Sonia, I am particularly proud to be the first of your many graduates to come.

Thanks to the members of my committee, Kirstie Hawkey, Timothy Lethbridge, Gabriel Wainer, and Tara Whalen, for your guidance, rapid feedback, and strong support of this thesis.

Thanks to all my friends for being flexible, understanding, and supportive throughout this journey. Of most relevance to this work, I would like to thank David Arnold, who first introduced me to graduate studies. Your friendship and tutelage have been invaluable to me. I already miss our late night working sessions, and I deeply appreciate your continued support.

Last but certainly not least, I thank my father, Roger Forget. I cannot think of a single time when you put yourself ahead of me. Only now do I realise and fully appreciate all the sacrifices you have made over the years for me. To you and my entire family, without your unwavering support and love throughout my life, I surely would not be so accomplished. Thank you, all of you.

I have been incredibly fortunate throughout my life. I have had the pleasure of knowing and working with many wonderful people. I have also had numerous incredible opportunities. To whomever or whatever is responsible, you have my eternal gratitude.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Statement of the Problem

Authentication can be defined as one entity proving its identity to another. But how can one entity always know another is who they claim to be? This problem has remained unresolved for millennia. Ali Baba overheard the passphrase to the Forty Thieves' hidden treasure den and stole some of their goods. Robin Hood and other outlaws throughout history could pose as a guard by simply donning a uniform and using stealth and guile to avoid scrutiny. For centuries, we have used signatures to prove our identity, despite evidence that signatures can be copied and forged. Being certain of an entity's identity is usually practically impossible. Thus, entities typically settle for being "reasonably sure" they can accurately authenticate others.

These challenges also exist in modern digital authentication. System administrators and users rely largely on a *text password* authentication scheme for digital authentication. When first registering for a service, a user will provide the system with a sequence of text characters as their password. To later access the service, the user will have to provide the exact same text password. Text passwords remain popular because they have several advantages. They are easy to learn to use, easy to implement, can be easily changed if they are compromised or forgotten, and are very secure (theoretically). Unfortunately, the proliferation of text passwords throughout the dozens of modern users' accounts has resulted in the cognitively impossible task of generating and remembering a distinct and random text password for each account. Thus, users cope as best they can by selecting memorable but often easily-guessed passwords, and re-using passwords across accounts, which lowers the security of said accounts. This is the essence of the *Password Problem*.

Usable security researchers have proposed some novel authentication schemes to either replace or improve the incumbent text password scheme. However, it remains

unclear how end-users can benefit from this ecosystem of schemes. This thesis explores the various challenges inherent in designing, integrating, and adopting multiple novel schemes for public use. We address these challenges by proposing solutions aimed at providing users with a usable and secure authentication experience.

1.2 Motivation

The field of usable security aims to provide users with secure computing environments without sacrificing system usability. Usable security has been a rapidly growing field of research for over a decade. Usable security research bridges psychology, human factors, cognitive science, and human-computer interaction (HCI) research with computer security. The field originally began with observations that theoretically very secure systems (such as passwords) were very insecure in practice because they were not designed with end-users in mind [1]. From such observations, Whitten and Tygar [214] first proposed five key usable security properties:

1. ***Unmotivated user property.*** Security is a secondary (background) task. Users should be able to perform their primary tasks securely without investing time and attention into security-related matters unless absolutely necessary.

2. ***Abstraction property.*** Security management is often abstract and unintuitive. Security systems requiring user interaction should use HCI principles whenever possible to make security tasks easy to understand and perform for users with little security knowledge.

3. ***Lack of feedback property.*** HCI research advocates for providing users with rich feedback to clearly communicate the state of the system. However, this is not always possible in security-related tasks, since this feedback may also be obtainable by an attacker, thereby introducing system vulnerabilities. It is important to provide legitimate users with the information they need to perform their tasks without leaking sensitive information to attackers.

4. ***Barn door property.*** If sensitive data is unprotected, even for an instant, we cannot guarantee that it has not been compromised by an attacker. Thus,

it is important to prevent users from making errors that may leave the system vulnerable, even for a short time.

5. ***Weakest link property.*** A system's security is only as strong as its weakest point, since an attacker need only gain a single point of entry to compromise the system. Thus, users should be supported in performing their tasks, secure and safe from all avenues of attack as possible with minimal inconvenience.

One type of computer security that users interact with daily is authentication. *Authentication* is the process by which an entity (typically a user) proves their claimed identity, by which they are granted the authorisation to perform actions or access resources that identity is permitted. Methods of authentication are typically split into four categories [51]:

- ***Knowledge-based (what you know).*** The user proves knowledge of some secret, such as a password or PIN.
- ***Biometrics (what you are).*** The system scans the user for a particular physical trait, such as a fingerprint.
- ***Token-based (what you have).*** The user proves they possess a physical token, such as a smart card or a one-time password generator.
- ***Behavioural (what you do).*** The user performs some action to prove their identity, such as gait or keystroke patterns when typing a sentence.

Each of these categories has their own advantages and disadvantages [51]. This thesis focuses on knowledge-based authentication mechanisms.

The knowledge-based authentication system most familiar to computer users is text passwords. Over the years, there have been many advances in the technical security of text passwords [66,124,130,140,220]. However, end-users continue to struggle to create and remember secure text passwords. Adams and Sasse [1] published a classic paper demonstrating the lack of usability of password systems. They found that authentication methods and security policies of two organisations did not fit the users' working environment. Unfortunately, the problems and causes described by Adams

and Sasse still exist today in many different systems [74,90,108,157,184,192,212]. We define these persistent challenges as the *Password Problem*, which can be summarised by following three reasons why users choose insecure passwords:

- ***Unreasonable memory demands.*** Users choose, reuse, write down, and publicly post predictable passwords because they cannot otherwise cope with being forced to memorise new passwords. Although password changing policies are intended to increase security, they usually result in *less* security because of human memory limitations, which leads to the emergent coping behaviours.

- ***Lack of security knowledge.*** There is a vast gap in password security knowledge between users and security professionals. Users do not know what constitutes a secure password. To the surprise of many security professionals, users are also often unaware that all dictionary words and names are the most vulnerable passwords to guessing attacks.

- ***Lack of threat awareness.*** Users are largely unaware of any potential security threats to them, their accounts, or the larger system, mostly because such threats are invisible to users. Users typically feel little motivation to behave securely since they know of no threats. However, if users are made aware of the security risks, they can become security conscious and behave appropriately. This challenges the standing assumption that users are never motivated to behave securely.

The Password Problem was initially documented by Morris and Thompson [140] in 1979. Almost thirty years later, Florêncio and Herley [74] found similar results in a study of online passwords and remark, "While much has changed since 1979 [...] it is just as true that many users appear to choose the weakest possible password." However, there has been progress in better understanding the nature of the Password Problem. Adams and Sasse [1] first noted that users can be motivated to behave securely so long as they perceive the need for such behaviours. Users must also be made aware of what constitutes secure behaviour, since the security threats and attack methods are not obvious to those outside the security community. Finally, security is a secondary task, impeding the completion of users' primary task. If behaving

securely is unreasonably difficult or time-consuming, then users will sacrifice security in favour of getting their job done.

Despite their shortcomings, text passwords remain the authentication method of choice for the majority of systems. Herley and van Oorschot [103] assert that, for the foreseeable future, plaintext passwords are unlikely to be replaced as the primary method of computer authentication. An examination of many authentication schemes and support systems by Bonneau et al. [24] revealed that text passwords have not been replaced because no alternative exists that performs better (or equally) across all 25 measured benefits. The authors agree that it is unlikely a single scheme will replace the incumbent text password system. They also state that selecting between authentication schemes and support systems involves a trade-off between the systems' various benefits. Thus, they advocate for a diverse ecosystem of authentication systems, since this provides security professionals and researchers the ability to use different schemes for different usability, security, and deployability requirements.

1.3 Contributions

This thesis proposes to enable and support a diversity of multiple authentication schemes in practice. This endeavour is advanced through the following contributions to the field of usable security and authentication:

1. *A user-centred authentication feature framework for identifying and comparing the features supported by knowledge-based authentication schemes.* This framework can used by authentication scheme researchers when designing or comparing authentication schemes, as well as administrators and users, who can use the framework to identify desirable features in schemes available for selection.

2. *Two novel authentication scheme designs and evaluations.* These schemes are outgrowths of Chiasson et al.'s previous graphical password work [36, 46]. The first presented scheme, Persuasive Text Passwords (PTP), is the first text password scheme to use Persuasive Technology to influence users to select more

secure text passwords. The second scheme, Cued Gaze-Points, is a shoulder-surfing resistant version of Cued Click-Points, whereby users select their password click-point by gazing at them instead of clicking. For both of these schemes, we perform user studies to test the security and memorability of the passwords users create with said schemes.

3. *Authentication scheme tutorial designs and evaluations.* If users will be exposed to diverse authentication schemes, security professionals must provide users with materials to quickly and effectively learn to create secure and memorable credentials with novel schemes. To this end, we harness research in computer system learning [32, 97] and Persuasive Technology [77] to design four different forms of tutorials for a novel authentication scheme. The tutorial types are tested in a user study to determine the most effective form of tutorial.

4. *Design of an architecture supporting multiple authentication schemes.* A world with many authentication schemes can most benefit users only if they can select the schemes suited to their abilities, preferences, and usage context. Thus, we present Choose Your Own Authentication (CYOA), an architecture that supports users in selecting an authentication scheme amongst several alternatives.

1.4 Organisation of Thesis Proposal

This document is written in first person plural prose because this work was performed under the supervision of my thesis advisors, Sonia Chiasson and Robert Biddle. The remainder of this document is organised as follows:

- Chapter 2 provides background on knowledge-based authentication research, including text passwords, graphical passwords, and previously proposed solutions to the Password Problem.

- Chapter 3 presents the User-Centred Authentication Feature Framework, which classifies many features that knowledge-based authentication schemes may support and explores how they can be leveraged to help users authenticate.

- Chapter 4 introduces the design and user study of Persuasive Text Passwords (PTP), a novel text password scheme that uses persuasion to guide users to create more secure passwords by inserting random characters into users' passwords.

- Chapter 5 discusses the implementation and user study of Cued Gaze-Points (CGP), a gaze-based cued-recall graphical password scheme based on Cued Click-Points. The use of gaze to select point instead of the mouse provides resistance against shoulder-surfing observation attacks, at the cost of a lower theoretical password space due to decreased pointing precision.

- Chapter 6 explores four different authentication scheme tutorial formats with two user studies to determine which format best helps users learn a novel and unfamiliar authentication scheme on their own. The tutorials are designed using two well-established techniques for communication information to users, Persuasive Technology [77] and the Minimal Manual [32].

- Chapter 7 details two possible designs for Choose Your Own Authentication, an architecture to empower users to choose an authentication scheme amongst several available alternatives to access their account on a system.

- Chapter 8 reviews the contributions of this thesis and offers concluding remarks.

1.5 Related Publications

This proposal includes research that has been published at academic conferences. The papers that contribute to each particular project include:

- ***Chapter 3: User-Centred Authentication Feature Framework.***

  - **A. Forget**, S. Chiasson, and R. Biddle. Persuasion as education for computer security. World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education (E-Learn). AACE, October 2007. (8-page full paper)

  - **A. Forget**, S. Chiasson, and R. Biddle. Lessons from Brain Age on persuasion for computer security. Computer-Human Interaction Work-in-Progress (CHI WIP). ACM, April 2009. (6-page extended abstract)
  - S. Chiasson, **A. Forget**, and R. Biddle. Accessibility and graphical passwords. Symposium on Accessible Privacy and Security (SOAPS). July 2008. (3-page web-published workshop paper)

- ***Chapter 4: Persuasive Text Passwords.***

  - **A. Forget**, S. Chiasson, P.C. van Oorschot, and R. Biddle. Improving text passwords through persuasion. In Symposium on Usable Privacy and Security (SOUPS). ACM, July 2008. (12-page full paper)
  - **A. Forget**, S. Chiasson, P.C. van Oorschot, and R. Biddle. Persuasion for Stronger Passwords: Motivation and Pilot Study. International Conference on Persuasive Technology. Springer, June 2008. (11-page full paper)
  - **A. Forget** and R. Biddle. Memorability of persuasive passwords. Computer-Human Interaction Student Research Competition (CHI SRC). ACM, April 2008. (6-page extended abstract)

- ***Chapter 5: Cued Gaze-Points.***

  - **A. Forget**, S. Chiasson, and R. Biddle. Shoulder-surfing resistance with eye-gaze entry in click-based graphical passwords. Computer-Human Interaction (CHI). ACM, April 2010. (4-page short paper)
  - **A. Forget**, S. Chiasson, and R. Biddle. Input precision for gaze-based graphical passwords. Computer-Human Interaction Work-in-Progress (CHI WIP). ACM, April 2010. (6-page extended abstract)

- ***Chapter 6: Authentication Scheme Learnability.***

  - **A. Forget**, S. Chiasson, and R. Biddle. Supporting learning of novel authentication schemes. World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education (E-Learn). AACE, October 2012. (10-page full paper)

In addition to these publications, the author has contributed to other published works that inform the contents of this thesis. These publications include:

- ***Cued-recall Graphical Passwords.*** The author was responsible for developing the authentication scheme prototypes used in the lab studies. The author also assisted with running said lab studies, analysing the data from the user studies, and writing up the results. This experience in implementing multiple password schemes gave rise to the ideas for CYOA (Chapter 7).

  - S. Chiasson, E. Stobert, **A. Forget**, R. Biddle, and P.C. van Oorschot. Persuasive Cued Click-Points: Design, implementation, and evaluation of a knowledge-based authentication mechanism. Transactions on Dependable and Secure Computing (TDSC) 9(2). IEEE, March/April 2012. (15-page journal paper)
  - S. Chiasson, **A. Forget**, R. Biddle, and P.C. van Oorschot. User interface design affects security: Patterns in click-based graphical passwords. International Journal of Information Security 8(6). Springer, December 2009. (12-page journal paper)
  - E. Stobert, **A. Forget**, S. Chiasson, P.C. van Oorschot, and R. Biddle. Exploring usability effects of increasing security in click-based graphical passwords. Annual Computer Security Applications Conference (ACSAC). ACM, December 2010. (10-page full paper)
  - S. Chiasson, **A. Forget**, E. Stobert, P.C. van Oorschot, and R. Biddle. Multiple password interference in text passwords and click-based graphical passwords. Conference on Computer and Communications Security (CCS). ACM, November 2009. (12-page full paper)
  - S. Chiasson, **A. Forget**, R. Biddle, and P.C. van Oorschot. Influencing users towards better passwords: Persuasive Cued Click-Points. HCI on People and Computers XXII. British Computer Society, September 2008. (10-page full paper)

- ***Multiple Versatile Passwords (MVP) Authentication Evaluation System.*** The author contributed to the initial design of the MVP architecture,

implemented several authentication scheme prototypes for MVP, designed and ran user studies discussed in this thesis with the architecture, and contributed several practical enhancements to the system, including CYOA.

  - S. Chiasson, C. Deschamps, E. Stobert, M. Hlywa, B.F. Machado, **A. Forget**, N. Wright, G. Chan, and R. Biddle. The MVP web-based authentication framework. Financial Cryptography and Data Security (FC). Springer, February-March 2012. (8-page short paper)

- ***Eye Gaze and Graphical Passwords.*** The prototypes used to display images and capture eye gaze data in this project's user studies were developed by the author.

  - D. LeBlanc, **A. Forget**, and R. Biddle. Guessing click-based graphical passwords by eye tracking. Privacy, Security, Trust (PST). IEEE, August 2010. (8-page full paper)

Chapter 2

Background

2.1 Definition of knowledge-based authentication

This thesis focuses primarily on *knowledge-based authentication (KBA)*, which we define as a human entity authenticating itself by proving their knowledge (by some means) of some secret that was previously shared with the party requesting the authentication [51]. This definition includes all authentication systems that require users to remember a secret, regardless of whether or not the user chooses to rely on coping strategies (such as writing the secret down, see Section 2.6.3) to remember said secrets. This definition **excludes** the following:

- Biometric [112] and token-based [146] authentication systems, since they do not require the user to remember a secret. Biometric systems include any system that measures some biological or behavioural aspect of the user, such as their fingerprint, retina, or gait. Token-based authentication includes any system which requires users to possess an item, such as a key card or mobile phone.
- Key-based authentication systems, such as Kerberos [143] and PGP [31], which typically rely on secret keys intended to be stored by a machine (and not a human).
- Password-authenticated key exchange (PAKE) protocols, such as EKE [14], SPEKE [110], and SRP [223], since the two communicating parties mutually authenticate each other with a shared secret using some form of KBA (usually a text password).

KBA is a popular form of authentication because it is relatively inexpensive to implement and typically requires no additional hardware. KBA schemes can theoretically be very secure, but their practical security is often limited by the lack of

uniqueness and complexity of the shared secrets that humans can remember. Researchers have proposed numerous different varieties of novel KBA schemes. In this chapter, we will describe the well-known forms of KBA, notably text and graphical password schemes. We will also review the literature on KBA issues that have arisen from use and research in the field, such as password space, password creation advice, password restriction policies, and system-assigned versus user-chosen passwords. This chapter concludes with an overview of current proposed solutions to KBA challenges.

This thesis focuses on the usability and security of KBA schemes. Regarding security, there are numerous avenues of attack to passwords, including social engineering (e.g. phishing), man-in-the-middle, malware, and more. Unless otherwise stated, the scope of this thesis is to defend against password guessing attacks while maintaining security against other attacks. We also make no assumptions about additional mechanisms in place to deter or prevent password guessing. Examples of such mechanisms includes gradually slowing down the system's response time when repeated failed login attempts occur, or employing a second authentication procedure under unusual login circumstances (e.g. logging in from a different location). The security provided by these mechanisms is *in addition* to the security of the main authentication scheme, and we believe their respective usability and security should be evaluated separately.

2.2 Text Passwords

Text passwords are the ubiquitous method of authentication in most modern computing environments. Unix systems initially implemented a simple text password system whereby account passwords were stored verbatim in a file [140]. Although the file was protected from casual reading and writing, a privileged and knowledgeable user could gain access to the file, thereby instantly compromising all users' passwords. Wilkes [220] proposed securing each new password with encryption before storing them during account creation. At login, such a system would encrypt the entered password, compare the result to the stored encrypted password for that user, and grant access if they matched. Although encryption added a layer of security, if the encryption key were compromised or broken, all passwords would also be compromised.

To remedy this vulnerability, Evans and Kantrowitz [66] proposed passing the entered password through a *one-way hashing* function [130], instead of using encryption. Such a hash function must have the following basic properties:

1. It must be computationally infeasible to reverse.
2. For two identical inputs, their outputs must also be identical.
3. It must be computationally infeasible to find two distinct inputs that result in the same output.

An attacker who compromises a file of hashed passwords will be unable to obtain the plaintext passwords without performing a *password guessing attack*, whereby the attacker chooses and hashes a candidate password, and compares the result to each of the hashes contained in the password file. Any hashes that match imply that the candidate password is the actual password for the corresponding account.

There have been further advances in secure text password storage technology, such as salting[1] and slow hash functions[2] [124]. It is important to note that these techniques can be used with any authentication mechanism where access is granted if and only if the user-provided credential precisely matches the stored credential.

There are three important KBA issues that have risen from experience and research in text passwords that should be considered during the design and evaluation of any KBA system: system-assigned vs user-chosen passwords, password spaces, and memorability.

System-assigned vs user-chosen. All KBA systems may assign a (typically random) password to users or allow users to set their own password. In general, randomly-generated system-assigned passwords are much more secure than user-chosen passwords, since users typically choose things they believe will be easiest to remember, and are seldom aware of attackers' vast capabilities and resources in guessing predictable passwords [1]. However, system-assigned passwords are typically much

[1]Salting refers to concatenating some account-dependent data (such as the username) to the entered password before hashing. This makes pre-hashed dictionary attacks significantly more costly.

[2]Slower hash functions force attackers to expend more processor time and resources for each password (plus salt) guessed.

harder for users to remember [224], since the user would not have spent the cognitive effort in creating the password (which helps memorability), nor would the password have any intrinsic meaning for the user [208, 228]. Shay et al. [182] recently found no difference in memorability between system-assigned passwords and system-assigned passphrases (i.e. sequences of words) of equivalent password strength. This suggests that system-assigned passwords may place too great a burden on users' memory unless the KBA system provides some form of memory aid. Examples of text password memory aids include semantic content (such as a randomly-generated news article headline system [114, 199]) and cueing (Section 4.8.1). However, Wright et al. [222] found no improvement in memorability between system-assigned passwords and recognition passphrases, whereby users entered their passphrases by selecting their assigned words from sequentially-displayed lists of candidate words. This demonstrates that not all forms of memory assistance are effective, and suggests further studies to evaluate other forms of aid.

Password Spaces. An authentication system's *theoretical password space (TPS)* is defined as the total number of distinct passwords the system can support. Password spaces tend to grow exponentially in size, and are typically expressed in *bits*, which is the base-2 logarithm of the total number of possible distinct passwords. For example, four-digit personal identification numbers (PINs, see Section 2.2.3) have a TPS of $log_2(10^4) \approx 13.29\, bits$, since each of the four digits can be any one of ten values (0 to 9). The TPS is often used to compare the security of authentication mechanisms. However, it is not an accurate measure of the security provided by authentication systems that allow user-chosen passwords. This is because users do not select passwords randomly, but rather passwords that have some meaning to them, which are easier to remember. The space of passwords that users are likely to select is called the *effective password space (EPS)*. Of course, the EPS is always a subspace of the TPS (since users cannot select a password that the system cannot support).

It is desirable for authentication schemes to have not only a large TPS, but also to guide users' password selection towards an EPS that is as close as possible in size to the TPS. Calculating a scheme's TPS is usually simple, but thus far, there is no definitive, objective way to calculate the EPS. The most widely-known attempt at

quantifying an EPS is the NIST [30] scoring model for the security provided by a text password restriction policy (see Section 2.2.1). However, further analysis has shown that this model is an inaccurate measure of the actual security provided by password restriction policies [34,126,186,212]. The EPS can also be estimated by the number of entries in an attack dictionary that would enumerate through the most likely chosen passwords. However, this too lacks rigour, since this measure is dependent on how one determines which passwords are most likely to be chosen.

Other studies have given glimpses into the EPS by examining the content of users' passwords. Unfortunately, these studies' results are not directly indicative of general user behaviour. For example, lab studies [208] lack the ecological validity necessary for their results to be generalisable, while field study results [74, 212] may only apply to the specific type of system the passwords were protecting (e.g. users may choose bank account passwords differently than blog passwords). More generalisable results on password spaces should be forthcoming as more lab and field studies are conducted.

Florêncio and Herley [75] recently found that popular websites with valuable assets (which are likely targets for attack) chose the least-restrictive password policies, requiring users to create passwords of only 20 *bits*. This may be the largest-sized text password EPS that competitive website designers believe users are willing to tolerate.

Memorability. Another important issue in KBA systems is *memorability*. This includes users' difficulty in remembering a single password and multiple distinct passwords (known as *multiple password interference*). Florêncio and Herley [74] estimate each user has about 25 accounts, which should each have a distinct and complex password. This requirement is either unknown or unreasonable to most users [1]. Thus, users typically resort to insecure coping behaviours (discussed in Section 2.6.3) to increase their passwords' memorability and avoid arduous password resetting.

Vu et al. [208] examined the memorability of text passwords created under various password restriction policies. They found users with five passwords had more difficulty remembering their passwords than users with only three, and users choose passwords with an obvious connection to the accounts, as a memory aid coping strategy. Chiasson et al. [42] compared the memorability of multiple text passwords and multiple PassPoints graphical passwords (see Section 2.3). They found that soon after

creating the passwords, graphical password users more easily recalled their passwords than text password users. Text password users often chose similar passwords across accounts to cope with the memory burden (see Section 2.6.3), which may have caused interference regarding which password belonged to which account. Interference has also been studied in a few other graphical password studies [37, 67, 139].

2.2.1 Password restriction policies

Password restriction policies are a set of rules that define the content and format of all passwords for a given system. These policies are usually the first method employed by system administrators to encourage users to select more secure passwords.

To our knowledge, Morris and Thompson [140] were the first to propose a password system that enforced restrictions, in an effort to help users choose more secure passwords. They suggested that password systems should not allow single-case passwords shorter than 6 characters or any password shorter than 5 characters.

The Federal Information Processing Standards (FIPS) released a standard for "Password Usage" in 1985 [71]. It provides detailed suggestions on password content and general use, both for end-users and security system administrators. However, Sasse et al. [176] stated in 2001 that password policies are still usually based on these guidelines, even though the computing environment and capabilities have changed much since 1985. The FIPS Password Usage guidelines were withdrawn in February of 2005 [72], and have not been replaced or updated. In 2006, the National Institute of Standards and Technology (NIST) updated their "Electronic Authentication Guideline" [30] for security system administrators. This guideline provides heuristics to score a password restriction policy in terms of bits of entropy corresponding to the degree of a password value's uncertainty. This model was based on Shannon's estimate of entropy in English text [181]. However, several studies [34, 126, 186, 212] have found that passwords created with particular password policies were more easily guessed than the NIST model suggested.

Komanduri et al. [126] recently performed a large between-subjects web study comparing four password restriction policies. They found a 16-character minimum password provided the best security and was least challenging for users to comply

with, when compared to an 8-character minimum excluding dictionary words or further restrictions. They also identified some misconceptions about how restriction policies affect password strength (which they measured using a calculation of entropy [183]). These findings include that digits added much entropy to passwords, excluding dictionary words added less entropy than expected, and that users created passwords that exceeded the bare minimum requirements.

Klein [124] first proposed another form of password restriction policy commonly known as *proactive password checking* [21], where the system itself emulates an attacker by performing a dictionary guessing attack on the user's password during registration or reset. Some of the passwords they recommended disallowing included dictionary words (and similar derivatives), digit-only passwords, repeating character passwords ("aaaaaaa"), and common keyboard patterns ("qwerty"). Schechter et al. [179] suggested a similar authentication system that would reject passwords that were already chosen by too many other system users.

However, some have claimed that password restriction policies do not improve password security [1]. Lab [208] and field [119] studies have tested this hypothesis. Users consistently have difficulty creating and remembering passwords under stringent and complex password policies. As a result, users are likely to rely more heavily on insecure password practices to cope with the difficult-to-remember passwords (see Section 2.6.3). Attackers can also more efficiently guess passwords by eliminating candidate passwords that do not fit the restriction policy. Florêncio and Herley [75] conclude from their examination of 75 commercial, government, and education websites that stringent password restriction policies are only tolerated by users when they have no other choice. Conversely, systems that compete for users seem to gravitate towards policies requiring passwords 20-bits strong (as described in Section 2.2). This lowers the chance that users are deterred from a cumbersome password restriction policy. The authors note that low-security password sites are also often the largest, most popular, and most likely to be attacked, due to the value and amount of assets they contain. Since such prominent websites continue to operate with less restrictive policies, the authors conclude that restrictive password policies may provide little security benefit at a considerable usability cost.

2.2.2 Password creation advice

Password creation advice is often offered by systems that enforce password restrictions. Unfortunately, both seem to be equally counter-productive on modern websites. Furnell [87] examined the password practices of ten popular Internet sites. Their password restriction policies and password advice were vastly different, and sometimes even conflicted with one another or themselves. Password advice was often ambiguous and unhelpful. All password policies and advice lacked consistency and effectiveness, making it very difficult for users to form accurate mental models of how to create a secure and memorable password.

There has been considerable research regarding how to best advise users to create secure and memorable passwords. Barton and Barton [13] originally proposed *mnemonic phrase-based passwords*: passwords that appear to have little meaning without knowledge of the memorable sentence from which they are derived. The majority of password advice research has focused on mnemonic phrase-based passwords, despite being seldom recommended in practice [87].

Yan et al. [224] published the first and largest experiment on mnemonic phrase-based passwords. Their 288-student field study compared *mnemonic* phrase-based passwords to *control* password advice ("Your password should be at least seven characters long and contain at least one nonletter") and *random* passwords. One month after creating the passwords, the authors successful cracked 32%, 8%, and 6% of the control, random, and mnemonic groups' passwords respectively. All of the cracked random and mnemonic passwords contained dictionary words, contrary to the advice provided. Only users in the mnemonic group used special characters, probably because the examples provided to them contained special characters. The authors measured password memorability as the length of time before users requested a password reset (if ever). They found that mnemonic passwords were at least as memorable as standard passwords and as secure as random passwords. They also concluded that password advice should explicitly encourage users to include special characters in their passwords, even though about 10% of users will not comply with advice.

Kuo et al. [129] later showed that mnemonic phrase-based passwords may not be as secure as originally thought. They asked 145 people to create a mnemonic

phrase-based password. They tested the security of the passwords by using John the Ripper [56] with a custom dictionary of 400,000 candidate mnemonic passwords from popular sources. They cracked 4% of mnemonic passwords. The authors also gathered standard passwords from another 145 people and cracked 11% of them with a 1,200,000 entry standard dictionary. The authors conclude that mnemonic phrase-based passwords were no more secure than regular passwords, because users based their mnemonic passwords on phrases easily found on the Internet. Furthermore, a motivated attacker could easily build a much larger mnemonic dictionary, and be likely to crack a higher percentage of passwords.

There are also different ways to generate a mnemonic phrase-based password. Vu et al. [208] ran a user study of two mnemonic phrase-based password generation methods. All users would choose their own sentence. One group was instructed to use the first letter of every word in a sentence, and the other to transform a sentence into a mnemonic string of characters. For example, "I had four snakes" could become "EyeH@4$nake$". Although they found little difference in password creation times, login times, and recall error rates, the latter mnemonic string method produced more secure passwords mainly because they contained more characters. In previous studies, they had also found that longer passwords were more resistant to cracking [164]. Unfortunately, the word and character substitutions the authors suggested are well-known by attackers [219], and may not provide as much extra security as previously thought.

Apart from mnemonic phrase-based passwords, there is little more research in password advice. Carstens et al. [33] performed a field study on the use of *chunking* [50, 138] to help users create memorable passwords. They compared standard password advice (at least 7 characters, a combination of symbols and letters, no terms repeated more than twice, not a dictionary word or personal data), to two-chunk, three-chunk, and four-chunk passwords. The authors found that four-chunk passwords, each separated by the same delimiting character, were longer and more memorable than the 7-character, two-chunk, or three-chunk passwords. Unfortunately, the authors provided no security analysis of the passwords created, or the password advice itself. The four-chunk passwords may have been less secure, since

they used fewer distinct symbols than the two- or three-chunk passwords. Furthermore, participants were explicitly told what to use for their chunks: participants' first and last initials, spouse's initials, employment start date, and so on. As a result of using predictable information, participants' passwords would have been easy to guess for an attacker, particularly one familiar with the user.

Password advice can also be represented visually with a *password strength meter*, which are typically an actual meter that illustrates the strength of the currently chosen password when a user is registering for an account. Password strength meters are used by popular websites like Gmail [94], PayPal [158], and eBay [62]. Ur et al. [206] recently evaluated 14 password meter variants in an online user study of over 2000 participants. They found that users of any password meter created more difficult to guess passwords than users without a strength meter. Furthermore, more stringent password meters influenced users to create stronger passwords. However, the authors found evidence that meters which are too stringent may discourage users and cause them to ignore the meter. Castelluccia et al. [34] have recently proposed a novel method of measuring password strength they call *Adaptive Password Strength Meters (APSMs)*. APSMs use Markov models [134] to measure a password's strength as the collective probability of each character following the previous characters in the password. These probabilities can be calculated either based on a training set of passwords or the passwords currently in use. The authors argue that APSMs can score passwords closer to the "ideal" password strength meter than any other proposed password strength metric to date. However, a formal usability study of APSMs and a practical security evaluation are yet to be conducted. Kelley et al. [120] recently proposed *guess number calculators* for estimating the number of guesses before a given password is cracked by a particular cracking algorithm. Their approach calculated the percentage of passwords the implemented algorithm would crack given a number of guesses. Cracking performance across algorithms can be compared by implementing and running guess number calculators for several cracking algorithms on the same set of passwords. Guess number calculators may also be a more practical and efficient method of proactive password checking [21] than running a computationally-intensive password cracking algorithm.

2.2.3 PINs

Personal identification numbers (PINs) are essentially passwords that consist of only digits. PINs were originally used for systems whose input was mainly numeric, such as telephone systems and as part of the two-factor authentication at automatic teller machines (ATMs). The problems with PINs were already well-known at the time [5], including the small password space and memorability problems.

In spite of this, PINs have since been used to protect some Internet services and mobile phones, and users continue having problems with them. Clarke and Furnell [48] surveyed 297 mobile users regarding their phone-related attitudes and practices. Three out of ten respondents felt PINs were inconvenient, and 38% of respondents had contacted technical support to unlock their phone after failing three times to enter the correct PIN.

PINs themselves have seldom been studied, but are often used as a point of comparison for the theoretical password space of novel authentication schemes. For example, Moncur and Leplâtre [139] found that users more easily remembered five 4-image graphical passwords than five 4-digit PINs over four weeks. Another example by Roth et al. [173] involved a user study and security analysis on three variants of their shoulder-surfing resistant PIN entry scheme. They claimed that users appreciated the added security of their scheme, despite it requiring more effort to use.

2.2.4 Password verification questions

Another form of text passwords are typically known as *challenge questions* or *personal verification questions (PVQs)*. When registering for an account, the system may ask the user to provide answers to one or more PVQs. If the user is ever unable to access their account through the primary authentication method (typically because they forgot their text password), the system will prompt the user to answer one or more PVQs that were asked during registration. If the user's given answers match the answers provided during registration, then the user is typically asked to reset their primary authentication secret, and allowed access to their account.

The earliest research in this form of authentication was known as cognitive and associative passwords [98, 163, 228]. *Cognitive passwords* were questions relating to

users' facts, interests, or opinions, while *associative passwords* were pairs of words, where the user responded to challenge words with the correct associated words [98]. Cognitive passwords were easier for users to remember, but more likely to be guessed by friends or family. No security analysis of either scheme was published at the time. Mike Just has contributed to this area by classifying the most common types of PVQs in a framework [115, 116] and performing a usability and security experiment and analysis on user-chosen question and answers [117, 118]. He found that users select low-entropy questions and still have difficulty remembering the correct response. Schechter et al. [177] found similar results in a study where users were asked to answer PVQs taken from the four most popular webmail providers (AOL, Google, Microsoft, and Yahoo!). PVQs may be more insecure that originally thought, since personal information about users is becoming increasingly publicly available through social networking websites and the Internet in general [165]. Thus, PVQs may provide more assistance to password crackers than legitimate users.

2.3 Graphical Passwords

Graphical password systems represent passwords in some form of visual format (as opposed to the lexical format of text passwords). Over the past decade, usable authentication researchers have taken great interest in graphical passwords, since humans have a better memory for visual stimuli than text [154, 187].

Biddle et al. [19] have recently published the most complete survey of graphical password research to date. They found that the usability and security of graphical passwords are generally inversely proportional, and advise that the next generation of graphical passwords (and usable security research in general) should aim to find solutions that increase usability and security simultaneously. They classify graphical passwords into three types. *Recall-based systems* (or drawmetric systems [53]) require users to draw a password, either on a blank canvas or grid. An example of a recall-based system is Draw-A-Secret (DAS, Figure 2.1) [113], where the user draws on top of a grid, and the password is represented as the sequential movements from one grid square to another. Further studies have shown that users choose predictable patterns

Figure 2.1: Draw-A-Secret [113]

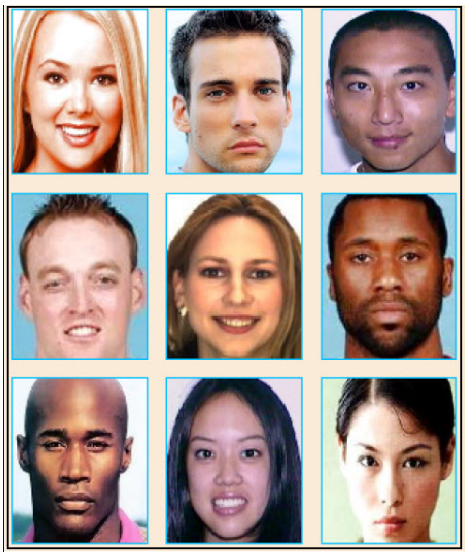


Figure 2.2: Passfaces [52]

in DAS [160], and may also be present in similar drawmetric schemes. *Recognition-based systems* (or cognometric systems [53]) typically present users with a set of images to memorise, so they may recognise and select them amongst decoys when logging in. Passfaces [155] (Figure 2.2) is a commercial recognition-based system where, in multiple panels of photographed faces, the user must select the faces of their "password". Unfortunately, Passfaces users choose faces in predictable patterns [52]. As a result, Passfaces changed the scheme, which now assigns a random set of faces, and no longer allows users to choose the faces in their password. This may be moot, since it has been shown that users have difficulty remembered system-assigned Passfaces passwords [67]. Some recognition-based systems are tested with small theoretical security, only equivalent to PINs, and thus may be suitable only for closed-circuit systems like ATMs. However, Hlywa et al. [105] compared a stronger version of Passfaces with equivalent schemes that used either houses or objects instead of faces, and found that objects were significantly more memorable. Stobert and Biddle [189] compared three similar recall, cued-recall, and recognition graphical password schemes. They found that a cued-recall and recognition hybrid scheme had the highest memorability and usability. *Cued-recall systems* (or locimetric [53]) present users with one or more images as a memory cue to help the user select their particular points on the image(s). We will cover some of these systems in more detail,

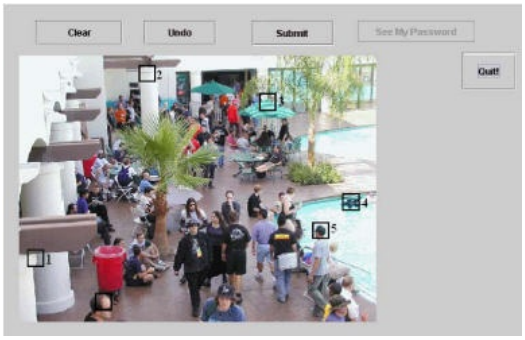


Figure 2.3: PassPoints [113]

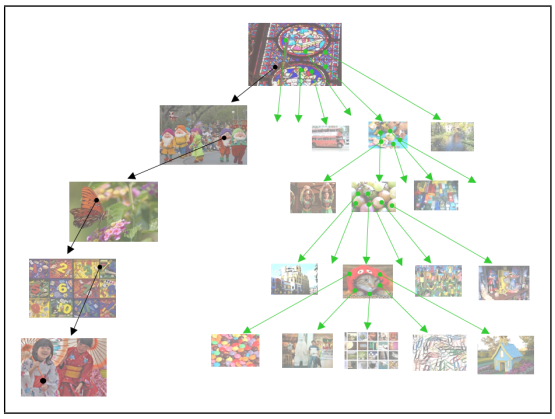


Figure 2.4: Illustration of CCP [44] and PCCP's [46] implicit feedback property, where different click-point locations lead to different subsequent images

since they have shown to provide security at least equivalent to text passwords and useful usability characteristics.

The most well-known and studied of cued-recall graphical password systems is PassPoints [215–217] (Figure 2.3). In the original configuration, a password consisted of a sequence of click-points on a single image. Although usable, PassPoints users often selected predictable passwords [41, 59, 159], raising significant security concerns. In response, Cued Click-Points [44] (CCP) was proposed, where users sequentially choose one click-point on each of 5 distinct images. Each subsequent image is determined by the user's previous click-point location (Figure 2.4). Attacking CCP requires more effort since it uses a large number of images, rather than only one. Chiasson et al. found CCP users less likely to select passwords in predictable patterns [41, 46], although some hotspots were still present in user's passwords. To encourage users to select more secure click-points, Persuasive Cued Click-Points [40, 46] (PCCP, Figure 2.5) added a randomly-positioned viewport on top of the image during password creation. PCCP users were required to select their click-point within this viewport. If they could not find a memorable point within the viewport, users could press a *shuffle* button to reposition the viewport to another random location. PCCP users were even less likely to select passwords with hotspots than PassPoints or CCP [41, 46].

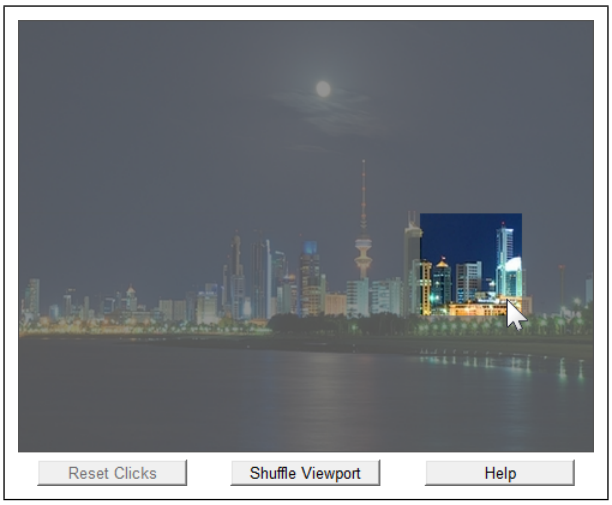


Figure 2.5: PCCP [46] interface for password creation

2.4 Social Knowledge-Based Authentication

The popularity of social networking sites such as Facebook, LinkedIn, and Google+ have attracted interest in *social authentication (somebody you know)* [26], where users can be authenticated by other people they know. Schechter et al. [178] ran a user study on a system where the user designates trustees who would verify the account holder's identity in person or over the phone, should an account recovery be requested. They found most users were able to authenticate successfully, but some people had difficulties remembering their trustees. Yardi et al. [226] implemented a Facebook application widget called Lineup, with which users must answer questions about presented photos to prove they belong to a social group, in order to access group-related resources. More social KBA schemes may be forthcoming, as users become more interconnected than ever before.

2.5 Challenge-Response

A significant concern in KBA are *observation attacks*, when an adversary observes the authentication process by some means. This is typically be done by either shoulder-surfing (watching or recording the authentication within physical proximity of the user) or by compromising the client machine with malware that records the system's state, input, and output, and sends it to the adversary. In KBA systems where

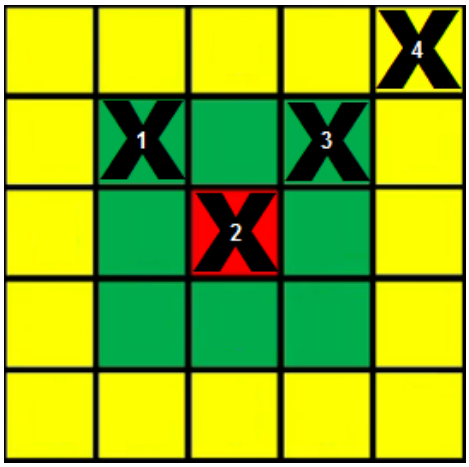


Figure 2.6: Illustration of GrIDsure [209] pattern during registration. Numbers represent the order of selected squares.

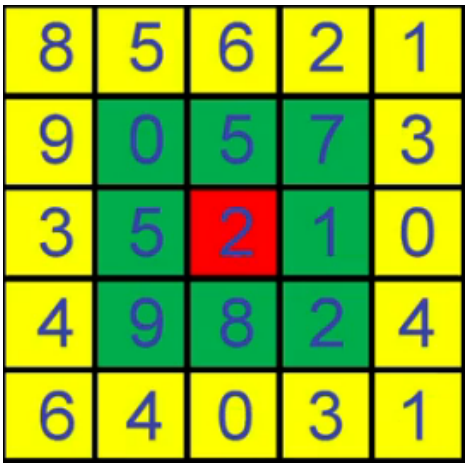


Figure 2.7: GrIDsure [209] login grid.

users must reveal the shared secret to prove their identity, an observation attack could immediately compromise users' secrets. These systems may also be vulnerable to *social engineering attacks*, where the user is deceived by an adversary to reveal the authentication secret. Some KBA systems have been proposed that can resist observation and even social engineering attacks. In so-called *challenge-response* or *obfuscated KBA systems*, users can prove their knowledge of the shared secret without providing the secret itself to the party requesting authentication.

A commercial PIN-type example of an obfuscated KBA system is GrIDsure [209], where users choose a pattern from squares on a 5×5 grid during registration (Figure 2.6). When logging in (Figure 2.7), the user is shown a 5×5 grid with a single randomly-chosen digit (0 to 9) on each square, so each digit appears on at least two squares. The user then proves knowledge of the secret grid pattern by typing the digits on their squares. For example, if the user's secret pattern was the one shown in Figure 2.6, then upon seeing Figure 2.7, the user would enter `0271`. Since each digit appears at least twice, the secret pattern cannot be compromised from a single observation attack (although the pattern could be derived from two or more observations). While Weber's [209] initial security analysis reported GrIDsure to be more secure than PINs, Bond's [23] analysis noted several security weaknesses. Brostoff et

al.'s [28] users achieved an 87% success rate logging into PDAs in a study of 4-digit GrIDsure PINs. When tested again two years later (without any practice since the original study), 12% of participants recalled their password without error.

The Convex Hull Click Scheme [218] is an obfuscated KBA graphical password system. Users initially select a set of small images as their among decoys. Over several rounds, a grid of icons is displayed wherein the user must locate a subset of at least 3 of their previously-chosen images, and click somewhere within the (invisible) triangle they form. The author's user study and security analysis concluded that the observation attack resistance comes at the cost of longer login times. This is typical for obfuscated KBA systems, since the user is usually required to perform some non-trivial mental task in order to respond correctly.

Coskun and Herley [49] criticise challenge-response schemes, since attackers can still derive the secret (or increase their chances of successfully guessing it) by observing one or more logins. They also point out that the shared secret must be stored in the clear by the authenticating system, and cannot be stored as a one-way hash to protect against server-side compromises. Yan et al. [225] compare several challenge-response schemes and demonstrated that they were vulnerable to secret-capture or brute force guessing attacks. The authors proposed five design principles to which challenge-response schemes must adhere to avoid being vulnerable. However, they conclude that such a scheme may be too cumbersome for people to use.

2.6 Other Approaches to the Password Problem

Users have difficulty creating and remembering multiple distinct and secure passwords. A number of potential solutions to the Password Problem (Section 1.1) have been proposed. We will briefly discuss them in this section, and explore possible reasons why these solutions have yet to gain wide acceptance.

2.6.1 Single Sign-On

Single sign-on (SSO) proposes to allow users to log in to a multitude of supporting services with a single username and password (or other authentication credential),

rather than requiring the unique credentials for each service. SSO architectures typically involve three parties: the user, the *relying party* (*RP*, who controls the service the user wishes to access), and the *identifying party* (*IdP*, who ultimately authenticates the user).

There are a number of separate SSO implementations. Perhaps the most well-known is OpenID [148,149,167]. A user's identity is defined by their OpenID, a unique uniform resource identifier (URI) owned by the user, where there is an extensible resource descriptor sequence (XRDS, an XML schema) [150] document containing the location of the user's IdP (amongst other information). When a user with an OpenID wishes to authenticate to an online service supporting the OpenID protocol, the authentication proceeds is as follows (Figure 2.8):

1. The user requests access to a service by providing their OpenID to the RP controlling the service.

2. The RP queries the identity server (located at the provided OpenID URL) for the user's OpenID document.

3. The identity server returns an HTML document that contains *link* tags used by the relying party to construct the URL of the IdP.

4. The RP instructs the client to redirect to the IdP's URL, and passes along two URLs to which the user should be redirected depending on whether the authentication is successful or not.

5. Redirected to the IdP's URL, the client requests to begin authentication.

6. The IdP returns to the client a web form prompting the user to enter their credentials.

7. The user enters their credentials (typically a user ID and password) on the provided web form and submits them to the IdP.

8. The IdP verifies the credentials. If they are invalid, return to step 6.

9. If the provided credentials are valid, then the IdP sends the client a form requesting confirmation that the user trusts the RP with their identity information.

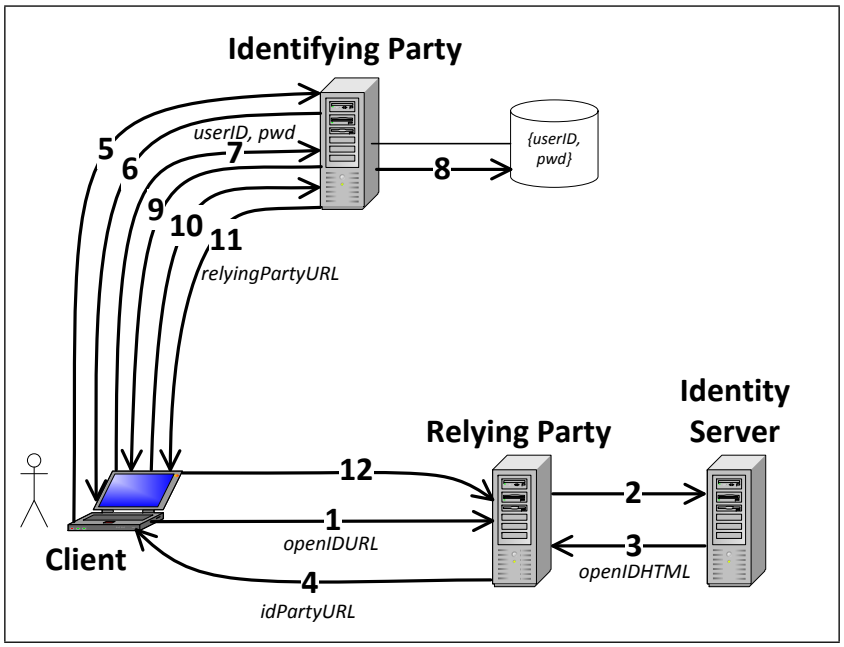


Figure 2.8: OpenID [148, 149, 167] protocol.

10. The user chooses whether or not they trust the RP and sends this response to the IdP.

11. Depending on whether the user choses to trust the RP or not, the IdP redirects the user to one of the two RP's URLs (provided at step 4), respectively signaling either successful or unsuccessful authentication.

12. Redirected to the appropriate RP's URL, the client requests and receives a web page confirming the successful or failed authentication.

Based on results from a user study, Sun et al. [194] list three reasons preventing the wide-adopting of OpenID. First, OpenID specifies no method for an IdP to authenticate itself to users before they provide their username and password. This makes OpenID vulnerable to phishing attacks, where users are tricked by a malicious party in revealing their account credentials. Second, users are accustomed to URLs representing Internet destinations and not identities, which may confuse users. Furthermore, URLs (and thus OpenIDs) are typically not particularly easy to remember [58]. Finally, IdPs can track all the RPs a user has logged in to, which raises privacy concerns. Sun et al. [195] have also analysed the OpenID protocol

using formal model checking, in addition to an empirical evaluation of the protocol's implementation by 132 popular websites. They found several vulnerabilities in both the protocol itself and the implementation thereof, and they suggest modifications to defend against such threats.

Information cards [86] are another form of SSO, where a user's identities are represented by XML documents containing information and various permissions given to the identity. These cards are issued by the user themselves or an IdP, and are managed by software on the user's computer called an *identity selector (IdS)*, such as Windows CardSpace [135] or Higgins Card Selector [63]. Authentication with information cards proceeds as follows:

1. The user uses a web browser to request access to a service controlled by an RP.
2. The RP returns to the browser a security policy detailing what type of identity is required to access the service.
3. The browser passes the RP's security policy to the IdS on the user's machine.
4. The IdS presents the user with a selection of cards that match the RP's security policy.
5. The user selects one of the presented information cards.
6. The IdS asks the IdP that issued the chosen card how the user should be authenticated, and prompts the user to prove credentials (presumably a username and password).
7. The user provides the requested credentials for the chosen card.
8. The IdS passes the credentials to the IdP, and requests confirmation that the user-provided credentials are valid.
9. Assuming authentication is successful, the IdP returns to the IdS a security token with all the claims requested in RP's security policy.
10. With the user's permission, the IdS passes the token to the web browser, which in turn passes the token to the RP.

Sun et al. [194] praise information cards for being phishing resistant and protecting users' privacy by mediating communication between IdP and RPs (so that IdPs cannot track which RPs users visit). However, the authors also criticise the need for client-side software, since this introduces vulnerabilities to malware, privacy issues regarding shared or public computers, and challenges when users switch between multiple computers.

OneID [147] is a commercial SSO solution currently in development whereby a user's identity is defined by the devices through which they access their accounts or make transactions. Users first create a OneID digital identity where they may store information such as their name, mailing address, e-mail address, and other information for the various websites the user frequents (such as credit card information for shopping websites). The user also adds their devices (computer, laptop, mobile phone, etc.) to their OneID account. This process provides the devices with private keys which ensure that the user's OneID account can only be accessed or used to sign in to websites from those devices. Users must provide a password when signing in to their OneID account if they haven't accessed or used it for a period of time (1 hour, by default). When a OneID user wishes to log in to an account on a website (the relying party or RP) supporting OneID (the identity provider or IdP):

1. The user uses a web browser to request access to a service controlled by an RP by clicking on a "OneID Sign In" button.

2. The RP returns to the browser a request for the user's OneID identification for this RP (hereafter referred to as the RPID).

3. The browser requests from the IdP (the OneID server) the user's RPID.

4. If the user has not used their OneID account for chosen period of time, the IdP requests the user provide the password to their OneID account.

5. If the user provides the correct password, or was already signed in to their OneID account, the IdP cryptographically signs the user's RPID with the user's OneID private key and returns the signature and RPID to the user's device.

6. The user's device then cryptographically signs the user's RPID with its own private key, and returns the RPID and two signatures to the RP.

7. The RP will then validate the signatures itself using public keys stored when the user registered for the website, or the RP can request the OneID server validate the signatures. If the signatures are valid, the user is granted access to the desired service.

OneID provides a similar process for filling out web form or completing an e-commerce transaction with a single click. OneID will ask for confirmation from users whenever a website requests more information stored in their OneID than was previously granted. OneID is still under development, so many details remain unclear.

Despite the differences between SSO implementations, Sun et al. [194] also discuss some critical problems with the general SSO architecture. First, organisations are reluctant to trust a third-party to authenticate users without any guarantees of security [141], particularly if such a third-party is a competitor [57]. Second, without any obviously tangible benefits to SSO, organisations may be reluctant to adopt it at the risk of confusing and discouraging users from using their service [57]. Finally, the authors argue that SSO provides little security or usability benefit to end users in exchange for a more complex authentication procedure. These results suggest that the current implementations of SSO systems have significant usability and security barriers to adoption, despite the obvious benefits of reducing the number of passwords users must remember and streamlining the login process across websites.

2.6.2 Password Managers

Password managers (PMs) relieve users from the burden of remembering passwords for different accounts. A PM is typically a stand-alone application or browser extension (plug-in) that a user installs on their client machine(s) to store or generate more secure passwords that users could otherwise create on their own. Users can later retrieve the passwords stored in the password manager by entering a master password. Bicakci et al. [17, 18] describe the two main PM designs, hashing PMs and password wallet PMs.

Hashing PMs

Hashing PMs algorithmically generate unique passwords using retrievable information, such as the master password and the name or URL of the system where the password is used. Most modern systems are implemented as browser extensions. Two such plug-ins in particular gathered significant academic attention, PwdHash [172] and Password Multiplier [99].

To use the PwdHash browser plug-in after installation, the user first clicks on the password field on the website the user wishes to access, and then types `@@` or presses the `F2` key to activate PwdHash. All subsequent key presses are recorded by PwdHash. Once focus leaves the password field, the user's password is hashed (Section 2.2) by PwdHash using the website's domain as a salt. The plug-in then puts the resulting hash into the login form's password field. Thus, the actual password for the user's account is the hash of the user's chosen password and the website domain. PwdHash has a number of defences to common attacks from a compromised browser, and can generate hashes for any website's password restriction policy through a user-configurable rule list. The authors briefly discussed results from a five-person user study, stating users no difficulty using it.

Password Multiplier [99] works quite differently than PwdHash, although they both offer similar benefits. When first installing the Password Multiplier browser plug-in, the user is asked for a username and a master password. The plug-in then performs a lengthy (approx. 100 seconds) iterated hash computation using the username and master password as the initial keys. The iterated hash computation is intended to force attackers to perform significant work for each guess of the username-password pair. The iterated hash result is then stored on disk. Thereafter, when the user wishes to log in to a website, the user either double-clicks on the password field or highlights it and presses `Alt+P`. A Password Multiplier dialog box will appear, requesting the user to enter their master password. The plug-in then generates a hash using the user's master password, the website's domain, and the previously computed iterated hash. The resulting hash is then placed into the password field. The authors did not report on any formal user study.

Chiasson et al. [43] ran a user study of PwdHash [172] and Password Multiplier [99]. The authors found that users not only were unable to use either PM, but also committed dangerous errors that could have revealed their passwords to an observer. They also found that participants felt they did not need a PM, were uncertain of the claimed security improvement, and would be reluctant to give control of their passwords (and hence accounts) to a manager.

More recently, Al-Sinani and Mitchell [106, 107] have proposed using Microsoft's CardSpace [135] as a password manager. However, the future of this work is uncertain, since CardSpace appears to be defunct [136].

Password wallet PMs

Password wallet PMs store users' passwords in a database encrypted with the master password. Most modern web browsers provide this functionality whenever remembering users' passwords. However, the security of browser-stored passwords is unclear, since they can often be easily retrieved by software freely available on the Internet [185].

Numerous password managers which have not been academically evaluated are available on the Internet. Password wallets usually operate locally, from either a hard disk or portable storage device, such as KeePass [168] and Password Safe [180]. LastPass [131] stores passwords locally, but also keeps the encrypted database synchronised with a copy on the LastPass servers. This provides both a backup of the user's passwords as well as a means to access passwords from different machines without the need to carry the database on a portable storage device. A similar password wallet PM is 1Password [2].

The primary drawback of all PMs is the requirement of trust in a third-party application [43]. A PM designed or compromised by a malicious adversary could easily and covertly obtain all the PM's users' passwords.

2.6.3 Coping Strategies

Adams and Sasse [1] were the first to document users' coping strategies with text passwords. Now over a decade later, these strategies are still in use [90], since passwords continue to impose unreasonable memory demands on users [108]. Although research in authentication coping strategies have focused on text passwords, they could also be used for most KBA systems.

Writing passwords down. When faced with complex password restriction policies (see Section 2.2.1) or frequent system-enforced password resets, users are likely to cope with the burden by writing their passwords down. Adams and Sasse [1] first noted this behaviour when half of their 139 survey respondents admitted to writing their passwords down (and the other half of respondents didn't answer the question at all). They quoted one participant stating, "...because I was forced into changing it every month, I had to write it down." Inglesant and Sasse [108] found similar results ten years later when studying password behaviour of participants in two organisations. Nine of fifteen participants in one organisation admitted to writing passwords down, but none of the seventeen participants in a second organisation claimed the same. The authors attributed the lack of writing passwords down to the second organisation's much less stringent password restriction policy, which allowed users to select more memorable passwords.

Password reuse. Another way users cope with the dozens of account passwords they need to remember is by using the same (or very similar) password on different websites. There is much evidence suggesting password reuse is a very common coping mechanism. Adams and Sasse [1] asserted that users were capable of remembering no more than five distinct passwords.

Password reuse has been observed in at least two controlled lab studies. Gaw and Felten [90] found that lab study participants reused three or fewer distinct passwords, despite only having 6 accounts on average. Participants openly stated they reused passwords. Chiasson et al. [42] tested multiple password interference in a lab study of 34 participants. They found over 50% of participants used an obvious pattern to create at least two of their passwords. Furthermore, over one-third of all user-created passwords were the result of obvious pattern reuse.

Password reuse is also prevalent in the field, with passwords to users' live accounts. Florêncio & Herley's [74] analysed the passwords from 544,960 client machines. They found the average client accessed 25 accounts using seven distinct passwords. This may be an overestimate, since it includes all distinct password entries, including incorrect password entries and passwords from different users of the same machine. Further field evidence of password reuse was reported by Stone-Gross et al. [192]. They obtained control of a botnet[3] that collected nearly 300,000 passwords from over 50,000 infected machines. Nearly 28% of victims reused passwords across websites.

Password sharing. Many systems assume (or insist) that every account owned by a single person. This policy is useful for many reasons in different contexts. The account may contain confidential information or give privileges intended only for the account's assigned owner. However, Adams & Sasse [1] pointed out that the one user per account model can fail in organisations where teamwork is key. In these cases, users either need to share their passwords or keep their work data outside of a secured account. Further evidence was found by Patrick's [157] examination of the password sharing behaviour of Enron employees. He performed a social network analysis on a publicly-available database of Enron employees' e-mails. He found several behaviours that may have contravened Enron's security policy, such as passwords e-mailed to or from external e-mail addresses owned by the same person, sharing of corporate and external accounts, and sharing of password-protected documents. This analysis could identify mismatches between the security policy and employees' needs.

Password sharing is not limited to the workplace. Singh et al. [184] revealed cultural and social practices where password- or PIN-sharing is considered acceptable. In fact, password sharing is sometimes expected or necessary, such as between married couples who share financial responsibility or people with special needs who depend on caretakers and bank clerks for assistance.

Weak password selection Over thirty years of evidence suggests that users select predictable passwords [1,74,108,140,192,212] Florêncio et al. [76] and Herley [104] argue that choosing memorable passwords over secure passwords is a rational choice, based on the estimated risk and value users place on the protected assets.

[3] A botnet is a collection of computers that have been compromised by a similar type of malicious software that relays instructions from and information to an attacker controlling the botnet.

All of the aforementioned research has contributed to our understanding of users' coping behaviours, but the solution is far from clear. For example, Gaw and Felten [90] found that users misunderstood the actual threats or capabilities of the attackers. Users have difficulty properly evaluating the security of their digital assets when the threats are abstract and non-obvious [214]. Herley [104] believes users would behave more securely if they better understood the threats. However, Inglesant and Sasse [108] disagree, believing that security managers would select different security policies if they better understood the costs to users and organisations of enforcing stringent policies with only security (and not usability) in mind.

2.7 Conclusion

Knowledge-based authentication (KBA) has several advantages, being relatively easy to implement, no required additional hardware, changing users' credentials is usually inexpensive, and they are theoretically very secure. However, experience and research (particularly with text passwords) has shown that KBA credentials can be difficult for users to remember, which results in a loss of usability and security. There have been many different proposed solutions to the challenges with KBA, including novel authentication schemes (mostly alternative text and graphical passwords), support systems (such as password managers), and architectures (such as single-sign on). While none of these approaches have yet been widely adopted, users continue to rely on a variety of coping mechanisms to deal with the abundance and complexity of text passwords they are supposed to remember. Towards solving this Password Problem, usable authentication research may benefit from an identification and examination of the specific authentication scheme features that can support users in creating secure, memorable, and usable passwords.

Chapter 3

User-Centred Authentication Feature Framework

3.1 Introduction

As discussed in Chapter 2, researchers have proposed numerous different varieties of novel KBA schemes. In our studies, we have noted some elements common in many, if not all, KBA schemes (Figure 3.1). They have two phases; registration and login. The registration phase involves the KBA scheme prompting the user to generate a password, which they must store in their memory. During the later login phase, the user must retrieve the password from memory, and provide it to the KBA scheme. Through both of these processes, different KBA schemes may support particular types of features that may help users in registering and logging in.

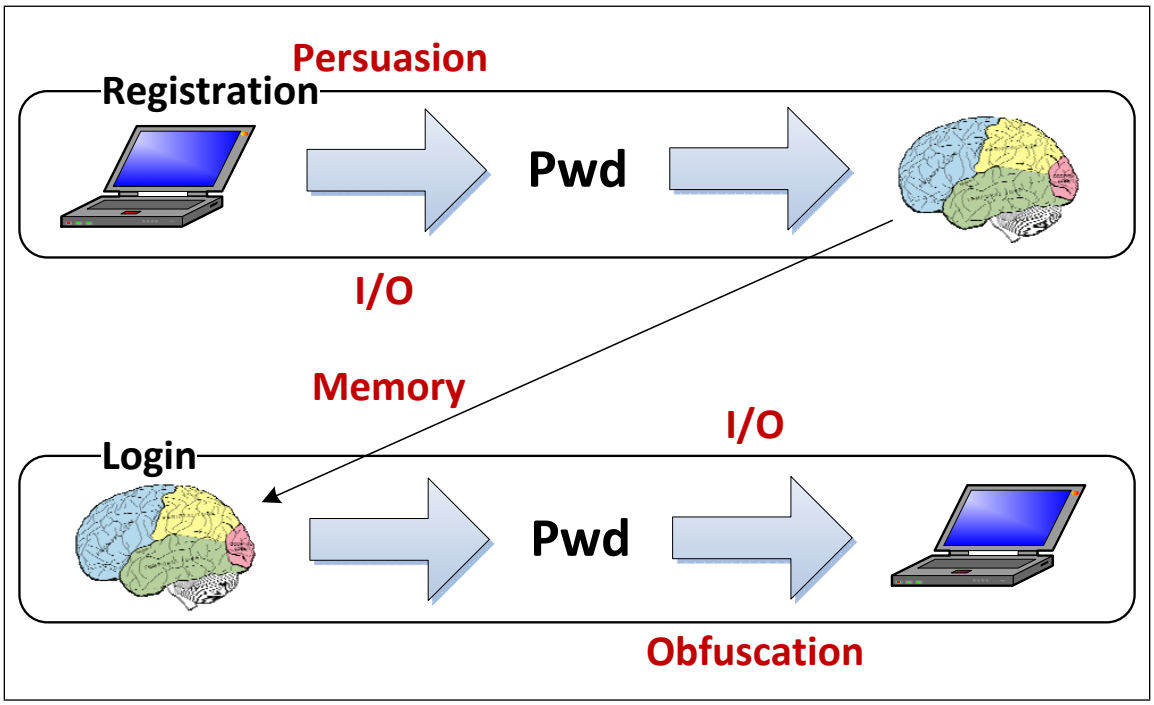


Figure 3.1: Illustration of the elements common to all knowledge-based authentication scheme

This chapter presents the User-Centred Authentication Feature Framework, which distinguishes some of the key features that schemes can support. This conceptual framework currently classifies features into each of the following classes:

- ***Persuasion.*** Persuasion is used during password creation to influence the user to choose more secure passwords. The most prominent use of persuasion in authentication is password strength meters (Section 2.2.2). Our framework identifies persuasive features in terms first described in Persuasive Technology (PT) [77]; a set of tools, media, and social cues that technology can leverage to guide and influence users to perform or change some behaviour.

- ***Memory.*** Another crucial aspect of authentication is how newly created passwords are encoded into memory and later retrieved at login. Authentication schemes should aim to allow users to create passwords in a format that increases memorability. Psychology and neurophysiology research [9, 64, 93, 133, 174] suggest that the human brain stores various types of information in different ways, which impacts people's ability to remember authentication credentials. The brain processes *lexical* and *visual* stimuli separately in short-term memory [10]. Thus, either the lexical or visual modality may be preferable for users with particular cognitive strengths or weaknesses. There are also three types of long-term memory most relevant to authentication; explicit memory, implicit memory, and prospective memory. Factual information that must be deliberately accessed is stored in *explicit memory*, including *semantic memories* such as names and dates, and *episodic memories*, which are specific past events. *Implicit memories* are recalled involuntarily and include *procedural memories* of actions required to perform some tasks, such as riding a bicycle, and *perceptual memory* subconsciously influences people's behaviour given a recent experience. Finally, *prospective memory* supports both explicit and implicit memories when a cue is mentally associated with a memory. For example, upon seeing a pill bottle, people may remember to take their pills.

- ***Input and output.*** How the user inputs their password and receives feedback from the system can affect a scheme's accessibility and potentially limits where

or how it can be used. Most authentication is performed with a keyboard or number keypad, and feedback is provided through a monitor. However, in some situations, it may be desirable to alter the method of credential input or feedback output. For example, using eye gaze (with an eye tracker) for credential entry could make shoulder-surfing observation attacks more difficult in a public place.

- ***Obfuscation.*** Some authentication schemes obfuscate the credential entry process to deter observation attacks. *Obscured input* schemes mask the method of entry, such as an ATM with a covered keypad. *Obscured feedback* schemes obscure sensitive information shown to the user when entering their credentials. The most common example are text passwords, where the user's password is usually echoed with dots or asterisks instead of the password characters. Finally, users of *challenge-response* schemes do not directly enter the shared secret when logging in, as is typically done. Instead, the scheme presents the user with a challenge, and the user must use the shared secret to derive the correct response to the system, thereby proving knowledge of the shared secret without actually revealing the secret itself.

KBA schemes may support any number of features in any of these classes. Each feature has inherent advantages and disadvantages. Developers using the User-Centred Authentication Feature Framework may decide that supporting some features would not be beneficial given their usage context and priorities. For example, supporting both lexical and visual working memory may be too cumbersome to be useful.

A lot of innovative work on authentication can be seen as exploring the different features identified in this framework. However, by presenting a principled set of classes, this framework highlights that there are even more opportunities that follow this structure. For example, the work on graphical passwords [19] addresses memorability (because of the pictorial superiority effect [142, 187]), but we show that, if the aim is to improve memorability, there is a rich set of knowledge in memory and other aspects of authentication that is yet to be explored.

In this chapter, we will elaborate on each of our User-Centred Authentication Feature Framework's various features classes and discuss the member features that authentication schemes may support, and provide examples of existing KBA schemes

supporting these features. We will then discuss the possible applications of our framework, and provide an example of how to identify the features supported by an authentication scheme. Afterwards, we will first describe an authentication framework proposed by Bonneau et al. [24] and discuss how the two frameworks are different and complementary. Finally, we will offer some concluding remarks. Parts of this chapter have been published as a full paper at E-Learn 2007 [82], an extended abstract and work-in-progress poster at CHI 2009 [79], and a workshop paper at SOAPS 2008 [39].

3.2 Persuasion

Authentication systems may leverage *persuasion* to guide users to generate more secure and memorable credentials and promote secure behaviour in general. Weirich and Sasse [213] first proposed using persuasion to encourage users to behave more securely with their accounts and passwords. The authors discuss that users often behave insecurely for many reasons:

1. Users have inaccurate mental models of attackers' identity, motives, targets, and capabilities.

2. At times, users feel required to behave insecurely in order to perform their primary task or achieve organisational goals. For example, users out of the home or office may need to share passwords so their family or co-workers can access time-critical information.

3. Users wish to avoid insulting colleagues by refusing access to their account. The organisation may also frown upon behaviours that foster a culture of mistrust and suspicion.

4. Users may not want to be labelled "paranoid", "nerd", or "not a team-player".

Weirich and Sasse recommend using persuasion to counteract the above stigmas regarding secure behaviour. They advocate using social marketing to change users' understanding and image of secure behaviour. They also suggest using fear appeals to demonstrate that security breaches are a real threat, what users can do to prevent them, and the consequences of users not complying with the security policy.

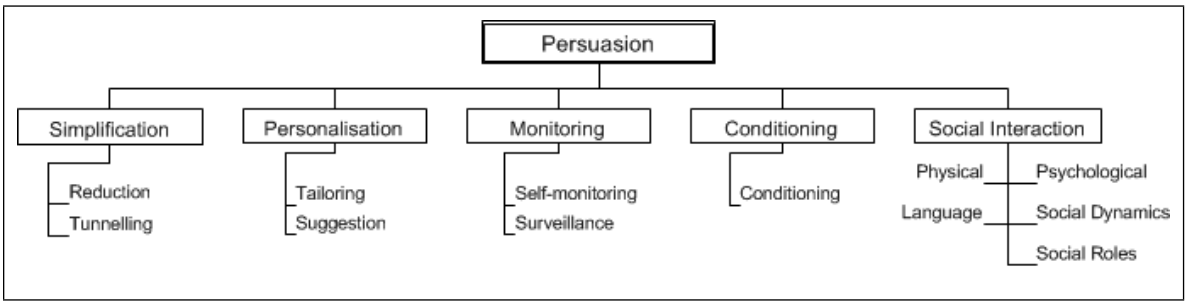


Figure 3.2: Persuasion features and the PT [82] principles on which they are based.

The use of persuasion is gaining popularity in many domains (including healthcare, education, and serious games) as a result of growing interest in Persuasive Technology (PT) [77]. PT is the emerging field of "interactive computing systems designed to change people's attitudes and behaviours". PT is founded on well-established theories from behavioural, personality, and social psychology. PT is a set of tools, cues, and media that technology can implement to influence users to behave in some desired manner. Persuasive tools render tasks quicker and easier to accomplish, persuasive media can convey messages through numerous representations, and persuasive social cues can help products appear friendly, knowledgeable, and trustworthy. Each of the three aforementioned persuasive roles is associated with a set of persuasion strategies. Which strategies to employ depend on many factors, such as the topic, medium, target audience, and desired persuasive strength.

This section groups and describes the PT principles that can be reasonably integrated into authentication systems, since not all of PT theory is practically applicable to authentication. For each principle, we identify the characteristics of usable security that are addressed.

3.2.1 Simplification

Authentication tasks should be as simple as possible. This includes reducing the process to the fewest actions as well as reducing the complexity of the remaining tasks. By simplifying the task of authentication, users can more easily form an accurate mental model of the authentication process. Since the burden of completing the task has been reduced to an acceptable level, users will then be less likely to

try to circumvent the security mechanisms, even though security is a secondary task. The optimal outcome of simplification is that the desired actions form the "path of least resistance", meaning it is easier for users to perform the authentication properly than it is to evade it. Simplified and shorter processes are also less susceptible to the deleterious effects on memory when users are interrupted mid-task [151].

A strategy often employed in usable security is to make security interfaces "transparent" [43] by hiding as much as possible from users. In our opinion, this is a misguided goal that often leads to more confusion as it usually translates into insufficient feedback for users. Simplification offers an alternative to transparency that reduces the burden on users without removing vital interface cues.

For example, password managers (Section 2.6.2) reduce the burden on users by having the computer generate and remember complex passwords for them. Users only enter one master password to activate the program, yet each of their accounts is protected by a distinct complex password generated by the password manager.

The simplification principle is based on the PT tools of reduction and tunnelling. PT advocates for the *reduction* of complex behaviour into simple tasks to lower the cost of performing the desired behaviour. Ergo, users can be persuaded to behave securely by making the most secure behaviour the easiest to perform. Developers can also leverage *tunnelling* by guiding users through a process or experience. This implies that breaking down a process into smaller steps not only simplifies the process for the user (making them more likely to perform the overall behaviour by completing the steps), but the smaller steps also provide additional opportunities to influence users' behaviour.

3.2.2 Personalisation

Customised information for individual users typically offers a more personal and engaging experience, which could be more persuasive than generic information. Users are concerned with security and privacy if they understand the implications and consequences of their actions [1]. By offering well-timed personalised advice relating to the individual's needs, preferences, or context-of-use, the system can provide details about why users' current behaviour is insecure and how it can be modified to be

more secure. Because the information is personalised, it is likely to help improve users' mental model of security and help them understand the relevance of behaving securely.

For example, users could provide some general interests (sports, music, finance, etc.) to a system that would customise a mnemonic phrase (Section 2.2.2) to help users remember a system-assigned random password. The given mnemonic phrase could further include something relating to the website or system itself, helping users to mentally link their mnemonic phrase and password to the system. This teaches users a coping strategy for remembering passwords that they can then apply to other randomly-generated passwords as well, thereby encouraging the use of both secure and memorable passwords.

The PT tools of tailoring and suggestion form the basis of the personalisation principle. Information will be more persuasive if it is *tailored* to the individual's needs, interests, personality, usage context, or other factors. Thus, the more relevant and personal an experience is, the more persuasive it is, resulting in a greater likelihood that the user will adopt the persuasive message. The proper timing of a *suggestion* can make it more persuasive. The most opportune moments (referred to as *kairos*) are often when users both are most receptive to a persuasive suggestion and can immediately act on it. In terms of authentication, kairos most obviously presents itself when the user is creating a password and desires to make a secure password that can be remembered. Other opportune moments may also present themselves, depending on the authentication application and context. One example would be when a user learns about a recent attack on passwords, particularly if someone close to them was affected.

3.2.3 Monitoring

When aware that they are being observed, users are more likely to perform the desired behaviour. A system tracking user performance or status can report it directly to the users, who may then adjust their behaviour in accordance with security policies. The system should provide the opportunity for users to learn what should be done to start behaving more securely. This monitoring can be automated and done entirely by the

system or can report to administrators who then take action. Furthermore, events that threaten security often happen in the background, over a long period of time, or as a result of a series of user actions. These events may not be obvious to users. In these cases, monitoring can help the system recognise these circumstances and bring them to the users' attention.

There is also the additional concern of the "barn door" property, where even the briefest insecure action can compromise the system's security [214]. These events may not be perceived by users as a cause for concern, which makes it important that the system raises an alert. It also provides the opportunity for intervention to teach how and why this behaviour puts the system at risk. Users who modify their behaviour can then see the effect as their reported performance improves.

The most widely use form of monitoring in authentication is password strength meters (Section 2.2.2). When the user is creating a text password, a password strength meter illustrates the estimated strength of the text password the user has entered. As users change the characters in their password, most strength meters update in real-time, providing users with an effective motivation for creating a password that is deemed secure by the meter.

Another example of monitoring are context-sensitive guidance polymorphic and audited dialogs (CSG-PAD) [29], which are modified dialog boxes which ask users to state the reason they are opening a potentially harmful e-mail attachment, and forward a copy of the attachment and reason to an auditor responsible for maintaining overall system security. Knowing that someone will be reviewing their choice and reasoning, CSG-PADs may convince users to reconsider opening e-mail attachments of unknown origin or purpose.

The monitoring principle stems from the self-monitoring and surveillance tools described in PT. Users can be persuaded to continue performing a desired task if given technology to support *self-monitoring* their progress. Technology is well designed and positioned to gather and present data. Users can be influenced and motivated to perform desirable behaviours by illustrating the results of their own actions. Similarly, users under *surveillance* by their peers may be persuaded to engage in a desired behaviour. The mere awareness that users' behaviour may be observed by peers can

be enough to motivate them to behave as desired. However, there are ethical concerns regarding the use of surveillance. In particular, covert surveillance may not only be unethical, but may also disenfranchise users from the persuaders (and their messages) when the covert surveillance is discovered.

3.2.4 Conditioning

Computer security is concerned about potential threats and risks to the system. However, most users have little direct experience with the consequences of an attack. When users perform a mental risk analysis, they do not believe that the probability of being attacked outweighs the additional burden of correctly performing the security tasks. In these cases, we need to artificially induce the correct behaviour because the users' natural environment does not support it. With user authentication, we want to convince people to use secure passwords even though it is a secondary task. For users to learn from any conditioning strategy, there should be other techniques at work to help users understand how to create effective passwords in order to receive the rewards for behaving securely. Examples of conditioning inducements in authentication systems include:

- Longer sessions before timing out and requiring users to re-enter their password.
- A customised icon and access to extra features and benefits.
- Providing a faster system response.
- A smiley face with encouraging messages like, "Your password is awesome! Good job!"

The conditioning principle is founded on the PT principle of the same name, states that positive reinforcement (i.e. rewards, praise) can encourage users to perform desirable behaviours.

3.2.5 Social Interaction

Authentication is an activity that typically occurs in isolation; users provide their secret authentication credentials while sitting at their computer. In other areas of

physical security, social norms influence behaviour and encourage users to behave securely. For example, someone may think twice about trying to enter a building without the proper credentials when there is a security guard or others nearby. The social interaction principle advocates repositioning user authentication as a social activity in order to leverage these social norms.

Users are more likely to be persuaded by a system that appears to share similar attitudes, traits, personality, and social membership. Such traits can be conveyed through language that best matches the users' own style, conveying a sense of "team" and encouraging cooperation. Positive and supportive language, such as personally greeting, befriending, and praising users, may further compel them to begin or continue behaving securely. Additionally, the system can be positioned to represent authority, potentially adding more persuasive power for people who respond well to authority figures.

For example, users can be taught that their own insecure behaviour puts others at risk. Through wording and presentation of the security system, users may develop a sense of belonging and duty towards their organisation. For example, organisation members can be told:

- Insecure accounts compromise not only their own account but the entire system.
- Everyone is counting on them to do their part.
- Their efforts at keeping the organisation secure are crucial and appreciated.
- "Other employees have passwords this strong. You don't want to be the weakest link."

The social interaction principle is based on five types of PT cues computing technology can leverage to behave as *social actors*. Devices can emulate *physical* cues, such as a face, eyes, body, and movement. Even simple caricatures can provide sufficient cues for a user to be influences by physical characteristics. *Psychological* cues such as humour, empathy, personality, and feelings can motivate users to behave as desired to please the avatar. Technology can also employ *language* cues to further engage and encourage the user to continue the desired task. *Social dynamics*, such

as turn taking and cooperation, can persuade users to perform desirable behaviours. For example, if the device performs some favour for users, they may feel compelled to return the favour by performing a behaviour requested by the device. Finally, computing technology can adopt one of many *social roles* (such as a teacher, teammate, opponent, guide, pet, etc.) to further persuade users to behave as they would for a live being in a similar social role.

The first known scheme to deliberately use persuasion was our Persuasive Cued Click-Points (PCCP) [40, 46] scheme. In applying PT to click-based graphical passwords, we significantly decreased the likelihood users will choose click-points on hotspots, which are particular areas on an image wherein many users would otherwise choose their click-points. This was accomplished by requiring users to choose their passwords' click-points inside a randomly-positioned *viewport*. Users could *shuffle* the viewport, whereupon it repositioned itself at another random location on the image. The persuasive elements had the positive effect of assisting users to choose more random graphical passwords while still maintaining usability. In Chapter 4, we apply similar persuasive principles to text passwords in the design of Persuasive Text Passwords .

3.3 Memory

There exists a great deal of research on human memory [9,64,93,133,174]. Memory is critical to knowledge-based authentication since its very nature depends on people's ability to process and recall credentials. The research community largely agrees that there are two types of memory; short-term and long-term. We will first discuss short-term memory, followed by the different aspects of long-term memory.

3.3.1 Short-term memory

Short-term memory (STM) or *working memory* is typically defined as the human capacity to mentally retain information in order to perform the current task [89]. The more a piece of information is processed and used in STM, the more likely it is to be encoded and easily accessible in *long-term memory (LTM)* (Section 3.3). Baddeley and Hitch [10] proposed what is currently considered the best model for STM, wherein

they describe a *central executive* responsible for managing attention, decision-making, and correlating information encoded by two slave systems, the *phonological loop* and the *visuospatial sketchpad*. The phonological loop retains verbal, auditory, and textual information through rehearsal. The visuospatial sketchpad holds visual, spatial, and possibly kinesthetic information.

This suggests two working memory modalities in which shared secrets can be represented; lexically or visually. These two representations are of particular importance to authentication. Users currently strongly rely on text passwords (Section 2.2) and PINs (Section 2.2.3), which are lexical KBA systems. However, some research in human memory has suggested that people have a better memory for visual stimuli than text [142, 187]. As a result of this *pictorial superiority effect*, graphical passwords (Section 2.3) have attracted much attention by usable authentication researchers as a possible means to improve password memorability.

The primary role of STM in authentication is to encode credentials into and later retrieve them from LTM. Information is typically encoded into LTM through its *rehearsal* in STM. Research [9, 133] has shown that the more often and deeply information is rehearsed in STM, the more easily it can be retrieved from LTM. Maintenance rehearsal (e.g. repeating a number) is much less effective at facilitating information recall than elaborative rehearsal (e.g. associating a number with a meaningful date).

Long-term memory can be defined as the cognitive functions whereby the brain stores information for later retrieval and use. Clearly, LTM is critical to KBAs. Without the ability to remember information persistently, KBA would be impossible. KBA system developers would be well-advised to be knowledgeable about LTM. Understanding the nature of memory may contribute to providing people with credentials and a user experience that facilitates and reinforces the encoding of credentials into LTM. Unfortunately, much remains unknown about human LTM. However, some general types of LTM can be classified as shown in Figure 3.3.

3.3.2 Explicit memory

Explicit (or *declarative*) *memory* is the deliberate and conscious retrieval of information from long-term memory. Tulving and Donaldson [201] distinguished between two

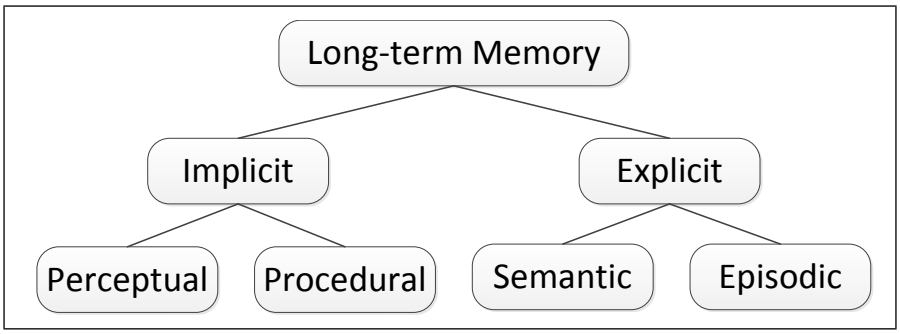


Figure 3.3: Classification of long-term memory types

types of explicit memory. *Episodic memory* represents the recollection of events or situations from the past. Examples of episodic memories include a previous birthday or where someone last remembers seeing their keys. *Semantic memory* represents the storage of general factual knowledge, without remembering where or how such knowledge was learnt. This includes information about people and objects, such as their names and descriptions. However, semantic memory is often supported by episodic memory. For example, when trying to remember someone's name (semantic memory), people sometimes try to remember the previous times and locations they have met said person (episodic memory).

Text passwords are the most obvious example of an authentication scheme that uses explicit memory. Users may choose to leverage semantic memory by basing their password on facts or information of relevance to them. They may also derive their password from a particular event from their past through episodic memory. Episodic memory is usually used in personal verification questions (Section 2.2.4) relating to information about the past (e.g. "What was your first car?"). People may choose to use episodic memory to create a PIN based on a personally meaningful date or year.

3.3.3 Implicit memory

Implicit memory refers to the ability to automatically recall some piece of information without the need to consciously retrieve the information from memory. Implicit memory can be subdivided into two types. *Procedural memory* involves remembering how to perform desired actions, such as driving a car or logging into a website. People do not need to explicitly recall how to perform actions stored in procedural memory;

they "just know" the general procedure for driving or logging in. *Perceptual memory* (or *priming*) is the cognitive function where some recent experience subconsciously influences a person's behaviour in a current task. This effect is often illustrated in experiments where participants are asked to complete a set of words beginning with a few letters (for example "fel"). Participants asked to read a list of words before the task are significantly more likely to unknowingly use words from the list than participants who had not read the list. This priming effect can significantly affect people's behaviour, even over long periods of time [203].

There is relatively less published research on implicit memory than explicit memory. Research has shown that explicit memory is slower and more effortful to access than implicit memory [174]. Thus, it may be advantageous to design authentication schemes whose credentials can easily be stored in implicit memory. How this may be possible is unclear. Weinshall and Kirkpatrick [211] were the first to propose three authentication schemes that leverage implicit memory (priming, in particular). Their most promising scheme was a picture recognition scheme whereby users viewed a hundred images as part of their training (or registration). Over the course of 3 months, users were periodically asked to login, where they were shown a series of image sets, each containing one of their hundred images amongst distractor images that had not been seen. Users authenticated by selecting the correct image in a sufficient number of sets. Users correctly identified the previously-seen image 90% of the time in sets of 6 to 9 images. The authors reported less success with similar text-based recognition systems. However, Denning et al. [210] claim that the aforementioned systems utilised explicit memory, not implicit memory as originally intended. This illustrates the challenge in isolating implicit memory.

Denning et al. [210] proposed their own priming scheme whereby registering users assign a label to each image in a given set. When logging in, users are shown degraded versions of images, some from their original set, and others randomly chosen from the total set (most of which the user had not seen). Again, the user must assign a label to each image. Their user study revealed a small priming effect, such that users more often correctly identified the degraded versions of images they had previously seen than images they had not seen.

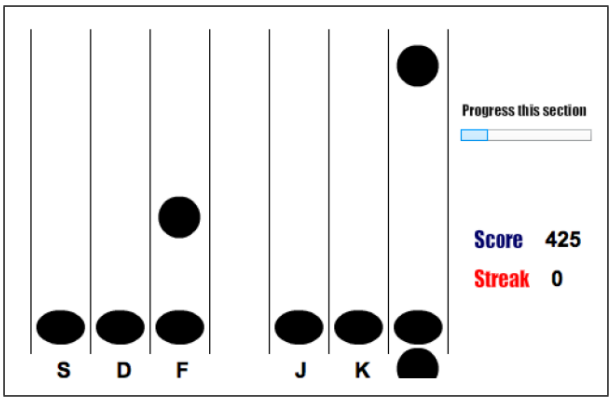


Figure 3.4: Serial Interception Sequence Learning (SISL) task used by Bojinov et al. [22]

Bojinov et al. [22] recently proposed the first scheme using procedural memory. The scheme requires users to perform a Serial Interception Sequence Learning (SISL) task (Figure 3.4), whereby there are a number of on-screen columns, each corresponding to a keyboard button. Objects fall down the columns at a constant speed. The user's goal is to press the key corresponding to the column where a shape reaches near the bottom of said column, thereby "intercepting" the object and being awarded points. The SISL task is conceptually the same as the popular games "Guitar Hero" and "Dance Dance Revolution". During a 30-60 minute registration, users perform hundreds of SISL task trials as described, unaware that in 80% of the trials are covertly embedded repeating sequences which the user is implicitly being trained to perform more accurately (through repetition) than random sequences. Thus, when perform the SISL task to log in, the system will grant the user access if the trained repeating sequences are performed with more accuracy than the random sequences. The author's user study found that users performed significantly more accurately for the trained sequences than random sequences immediately after registration, as well as one and two weeks later. The authors also asked users if they recognised various SISL task patterns, including both random patterns and those they had been covertly trained. The participants were unable to recognise their trained patterns from random ones. Although the registration and login time of the SISL task may be somewhat prohibitive for wide-adoption, this work clearly demonstrates that procedural memory is a promising means for authentication.

3.3.4 Prospective memory and cueing

Psychology research has shown that it is difficult to remember information spontaneously without some form of context-specific assistance [4, 122, 202, 204], or *memory cue*. This research suggests that authentication schemes should provide users with cues to assist with memory retrieval. However, care must be taken when providing cues, since ill-considered cues may reveal password information to adversaries.

Memory cues support both explicit and implicit memory, depending on the context and how the cue is presented [174]. Ellis and Kvavilashvili [65] state that memory cues support *prospective memory* [9], which is the ability to generate, retain, and later recall information in the appropriate context. The authors summarise some prospective memory findings that are relevant to authentication. The saliency of a presented cue does not seem to improve memory any more than a cue that is present but not particularly salient. Thus, there is no need to interfere with the authentication task itself in order to ensure the user saw the cue. Merely presenting the cue within the authentication context should assist users in recalling their credentials. The authors also discuss a curious difference in prospective memory performance between young and older adults. Younger participants outperform older ones in laboratory settings, but the reverse is true in field studies. The reasons for this remain unclear, but some studies suggest that older adults may make better mental use of the environment and context in which the recall takes place, which the unfamiliar settings of a laboratory may not adequately provide. Developers and researchers may wish to consider such factors in the design and analysis of novel authentication schemes, to ensure it provides appropriate memory support for their target users' age.

Personal verification questions (Section 2.2.4), as well as recognition and cued-recall graphical passwords (Section 2.3) are the KBA systems that best utilise cueing thus far. However, different schemes leverage cueing to varying degrees. For example, PassPoints [215–217] shows the user a single image that is a memory cue for their five click-point password. However, in Cued Click-Points [44] (and derivatives thereof), each image is a memory cue for a single click. CCP may form stronger mental associations between the image and click-point location than in PassPoints, where more information (several clicks) are associated with a single image.

Graphical passwords aside, research in KBA memory cueing has been limited to providing inkblots as cues for text passwords. Stubblefield and Simon [193] first proposed showing users a sequence of inkblots as memory cues for users' text passwords. They found that most users in their lab study could successfully log in without error after both one and seven days. The few users who did err only did so for one of ten inkblot associations. Their users were surprised that they could remember such long and strong passwords with the assistance of the inkblots. Renaud et al. [170] tried presenting users with a customisable inkblot and asked them for a description as the password. Their 9-week web study users did not create strong passwords with this implementation. This study may suggest that simply providing a cue is not enough. Users may also need some guidance on how to use the cue to create both memorable and secure password. Otherwise, they may simply fall back on less secure password creation strategies (Section 2.6.3). A good example is the behaviour of our users in a lab study where users were required to create several passwords for different mock systems (bank, e-mail, blog, etc.) [42]. We found that 23 out of 34 users used the system as part of their password. This suggests that users will take advantage of memory cues when possible, whether or not they are explicitly provided. However, an adversary attacking all accounts on a system could incorporate cue-based information in the password attack. This vulnerability can be minimised if each user were provided with their own unique memory cue, such as Stubblefield and Simon's randomly-generated inkblots [193].

3.4 Input and output

In some circumstances, it may be advantageous to vary the secret's entry method to something other than the standard keyboard or keypad. Most work in varying the KBA input modality has been mainly to defend against shoulder-surfing observation attacks. Kumar et al. [127] had some success in a lab study of a text password entry system where users would gaze at on-screen keyboard keys to "type" their password.

Other proposals use a haptic device to provide the user with tactile feedback and input. Sasamoto et al. [175] experimented with contact- and friction-based tactile feedback in the development of their Undercover authentication scheme. Their user

study on the final version found the system to be reasonably usable, but some users' behaviour negated the shoulder-surfing resistance of the haptic device. Bianchi et al. have built and tested a haptic keypad [15] and haptic wheel [16], which uses tactons [27] to generate vibrations that provide user feedback regarding the keys' values or the wheel's current position.

Another innovative KBA input method is thought-based, first proposed by Thorpe et al. [198]. They explored the goals, motivation, and potential design of *pass-thoughts*, which uses a brain-computer interface that can measure users' binary responses to presented stimuli. Further research into thought-based authentication is on-going [121,200], but no proposed schemes present a sufficiently accurate and usable alternative authentication method thus far.

3.5 Obfuscation

A significant concern in KBA are *observation attacks*, when an adversary observes the authentication process by some means. This can be done by any person by shoulder-surfing, watching or recording the authentication within physical proximity of the user. The simplest defence from this attack is with a *obscured feedback*, whereby any sensitive feedback provided to the user is hidden. The most common example is text password systems, which usually echo dots, asterisks, or nothing instead of the user's password characters. Whenever an Apple iPhone [8] user enters a character in a password field on a webpage, the character is shown unmasked either for one second or until another character is typed. Zakaria et al. [227] proposed and studied three methods of obscuring Draw-A-Secret [113] users' password drawings.

A shoulder-surfing attacker could also observe the method of entry, which is commonly done to capture ATM users' PINs by either simply watching or installing a camera to record the PIN entry. An attacker may could also compromise the client machine with malware that records the system's state, input, and output, and sends it to the adversary. An observation attack could immediately compromise users' secrets in KBA systems where users must reveal the entire shared secret to prove their identity (such as providing a text password). *Obscured input* schemes attempt to

make such input observation attacks more difficult. Some ATMs have covered keypads that make visual observation attacks more difficult, but do not protect against a compromised system. An example that provides some protection from malware is VibraPass [55]. During login, VibraPass will cause the user's mobile phone (assumed to be in their pocket) to vibrate whenever the next password character or PIN digit entered by user should be *deliberately incorrect*, in order to confuse observers. The author's user study revealed that whenever users' phone would vibrate, they would visibly pause to think of and enter an incorrect character. An attacker may also hear a vibrating phone, and thus know which characters were incorrect.

Some proposed KBA systems can resist both observation and *social engineering attacks*, where users are deceived into revealing the authentication secret. In so-called *challenge-response* schemes, users prove their knowledge of the shared secret without revealing the whole secret itself to the party requesting authentication, thereby hiding the secret from observers.

A commercial example of a challenge-response system is GrIDsure [209] (Figures 2.6 and 2.7). Users choose a pattern on a 5×5 grid during registration. At login, users are shown a 5×5 grid, each square containing a single randomly-chosen digit (0 to 9). Thus, each digit appears on at least two squares. The user proves knowledge of the secret grid pattern by typing the digits on their squares. Since each digit appears at least twice, the secret pattern cannot be compromised from an observation attack (although repeated observations may compromise the pattern).

The Convex Hull Click Scheme [218] is an obfuscated KBA graphical password system. Users initially select a set of small images among decoys as their password. Over several rounds, a grid of icons is displayed wherein the user must locate a subset of at least 3 of their previously-chosen images, and click somewhere within the (invisible) triangle they form. The author's user study and security analysis concluded that the observation attack resistance comes at the cost of longer login times. This is typical for obfuscated KBA systems, since the user is usually required to perform some mental task in order to respond correctly. Yan et al. [225] provide a recent survey of several challenge-response schemes and proposed five design principles to which such schemes must adhere to be secure.

Some schemes propose using special hardware to provide information to the user in a difficult-to-observe manner. For example, Sasamoto et al. [175] proposed a tactile feedback system whereby the user modifies their authentication response based on information covertly provided through the tactile channel. Their scheme, Undercover, supports all three obfuscation features. An attacker observing the input device (a keypad through which the user selects one of five images) and the output device (the monitor displaying the images) cannot know which image the user selected because the mapping of the images to the keypad is covertly communicated to the user through the tactile channel. Thus, Undercover supports both the obfuscated input and obfuscated output features. Undercover also supports challenge-response because the user responds to the presented challenge differently at every login, depending on the position of the user's image and the tactile feedback.

Sec.	Feature Class	Feature Name	Abbr.
3.2	Persuasion	Simplification	Simp
		Personalisation	Prsn
		Monitoring	Mntr
		Conditioning	Cond
		Social Interaction	Socl
3.3	Memory	Lexical memory	Lex
		Visual memory	Vism
		Semantic memory	Sem
		Episodic memory	Eps
		Procedural memory	Proc
		Perceptual memory	Perc
		Prospective memory / Cueing	Cue
3.4	Input and output	Keyboard	Kbd
		Mouse	Mse
		Touch	Tch
		Eye gaze	Eye
		Haptic	Hap
		Vocal	Voc
		Visual	Viso
		Tactile feedback	Tact
		Auditory	Aud
3.5	Obfuscation	Obscured input	Oin
		Obscured feedback	Ofb
		Challenge-response	C-R

Table 3.1: User-Centred Authentication Feature Framework

3.6 Framework Summary

Table 3.1 summarises the User-Centred Authentication Feature Framework we have described in this chapter. The framework consists of the following four feature classes:

- *Persuasion* features attempt to influence the user to select more secure or memorable passwords than they would otherwise. These features may be particularly useful for authentication schemes where it is unlikely to be obvious to the user how to generate and enter a secure and memorable password. However, these features should be used with caution, since some users may object to persuasion that is too forceful.

- *Memory* features provide different cognitive methods for users to remember their passwords. Some memory features (such as lexical memory and visual memory) rely on users' short-term memory to encode passwords into long-term memory. Other memory features support storing passwords in different formats in long-term memory. Users may have different abilities for particular memory types, and would prefer schemes that support said memory features.

- *Input and output* features provide users with different methods of inputting their password and receiving feedback. These features are an important consideration when designing authentication schemes for devices with novel interfaces or for particular groups of users. Users with disabilities often need alternative input and output modalities, so authentication scheme designers should carefully consider which of these features may support the largest segment of users.

- *Obfuscation* features attempt to make it more difficult for illicit observers to intercept users' passwords. These features can significantly improve security, but they often result in decreased usability. Thus, the scheme's threat model should be carefully evaluated before considering the use of obfuscation features. The implementation and testing of said features should also be carefully executed, to ensure that the scheme meets the expected usability benchmarks.

Scheme	Persuasion	Memory	I/O	Obfuscation
Text Passwords	-	Lex, Sem, Eps, Proc	Kbd, Viso	Ofb
PCCP [46]	Simp, Prsn, Cond	Vism, Sem, Eps, Proc, Cue	Mse, Viso	-
GrIDsure [209]	-	Vism, Sem	Kbd, Viso	C-R

Table 3.2: Features supported by various authentication schemes, as an example of how the User-Centred Authentication Feature Framework can be used.

Each of these classes encompasses multiple features. Any given authentication scheme may support any number of the features in any class. This conceptual framework is intended as a taxonomy of features that schemes may support. Thus, a scheme that supports more features is not necessarily superior to another that supports fewer features. Indeed, it may be difficult to imagine a scheme that supports all features, let alone how usable or practical said scheme would be.

The framework presented in Table 3.1 is a cataloguing of features and classes that we believe are currently most relevant or popular. More features and classes can be added to the framework, such as ones we may have overlooked or ones resulting from novel advances in authentication technology.

Table 3.1 also includes the section numbers where each class' features are discussed, and suggests a shorthand abbreviation for each feature to be used when listing the features supported by a scheme. A benefit of the User-Centred Authentication Feature Framework is the ability to identify and classify the features supported by particular authentication schemes. To illustrate how this is done, we will describe the features found in the various authentication schemes listed in Table 3.2:

Text passwords (Section 2.2) typically assume a keyboard (*Kbd*) will be used as the input modality, and that the process and result of the password entry attempt will be visually (*Viso*) displayed on a monitor. Text password systems were not designed to leverage any persuasive elements, although password strength meters (Section 2.2.2) could add persuasion to text passwords to encourage users to create more secure passwords. Text password systems clearly use the lexical (*Lex*) aspect of people's working memory, given the textual nature of the secret password users must recall. Users have the option of choosing a text password which is based on either some important moment of their past from episodic memory (*Eps*), some factual information from semantic memory (*Sem*), or a combination both. With significant

practice, passwords can often by typed quickly and automatically from procedural memory (*Proc)*, without needing to consciously recall the semantic or episodic contents of the password. Finally, almost all text password systems implement obscured feedback (*Ofb*) by masking or not displaying any characters typed in a password field.

Persuasive Cued Click-Points [46] (PCCP, Section 2.3) was designed for use with a mouse (*Mse*), and a visual output device (*Viso*) is required to render and display the images and allow the user to visually select their click-point locations. PCCP simplifies (*Simp*) the selection of more secure click-points, since it is easiest to select a point in the first randomly-positioned viewport, rather than repeatedly shuffle until the viewport highlights a more obvious and predictable click-point. Users are free to select a click-point that is personally (*Prsn*) meaningful to them, regardless of where it is located. However, users are conditioned (*Cond*) to shuffle as little as possible, since continuously shuffling to highlight one eye-catching click-point can quickly become tedious. PCCP passwords, like most graphical passwords, are rehearsed and recalled in users' visual memory (*Vism*). When selecting click-points, users may choose locations with either memorable semantic (*Sem*) details or are representative of some episodic memory (*Eps*). After repeated logins, users are likely to implicitly recognise their click-point locations through procedural memory (*Proc*). However, even when first logging in, the presentation of the images acts as a cue, triggering their prospective memory (*Cue*). Unfortunately, PCCP does not support obfuscation, since an adversary can observe users click on their password's click-points when logging in.

GrIDsure [209] (Section 2.5) clearly requires some form of visual display device (*Viso*) to show the grid of digits, and a keyboard (*Kbd*) to input said digits (although some form of pointing device to select grid squares would facilitate password creation). GrIDsure passwords are represented visually (*Vism*) as a pattern of squares on a 5×5 grid. When logging in, users are shown a grid of digits, and they must enter the digits that correspond to their grid pattern. However, the placement of digits on the grid changes for every login, users do not need to remember the digits, and therefore do not use lexical memory to remember their credentials. Users are likely to store their chosen grid pattern in semantic memory (*Sem*), either in relation to some shape or

object that is meaningful to them. The primary benefit of GrIDsure is its challenge-response (*C-R*) feature. An adversary observing the shown grid of digits and the user's input digits cannot determine the user's secret grid pattern with certainty, because each digit is shown at least twice in the 5×5 grid.

There are several ways this form of feature analysis can be beneficial for various stakeholders in authentication. In highlighting these supportable features, researchers designing a novel authentication scheme can consider to what degree they wish to incorporate particular features into their new scheme. For example, researchers who focus on designing modifications to existing text password system could benefit from our framework by considering adding visual cues to improve password memorability through visual working memory (in addition to lexical memory inherent in text passwords) and prospective memory and cueing. They may also consider the advantages of altering the input modality (as Kumar et al. [127] used gaze for text password input) or output modality (such as tactile keyboard feedback to guide the user's password entry). Persuasion is a very rich feature with many different tools to guide and influence users in powerful ways. For example, Persuasive Text Passwords (Chapter 4) applies the principles of Simplification, Conditioning, and Personalisation in a particular way. However, other researchers may combine different principles or apply the same principles differently, both of which may yield novel schemes with potential to guide users to create more secure and memorable passwords.

Choosing amongst a wide array of proposed authentication schemes can be time-consuming and confusing. The User-Centred Authentication Feature Framework can also assist people to select a scheme best suited to their requirements and usage context. IT professionals can use our framework to identify which features they and their users require, thereby shortlisting schemes that support said features. For example, an organisation with accessibility concerns could use our framework to focus on authentication schemes with particular input or output modalities. An application targeting users with visual impairments could consider schemes that leverage the output modalities favoured by their users, such as text-to-speech software or tactile feedback devices. An organisation building software for use on a touchscreen may wish to favour schemes designed for a touch interface. In some cases, administrators

may wish to avoid using certain spectrums of a feature type. An obvious example would be that applications for blind people should avoid schemes that rely heavily on a visual output modality. A less obvious example is a website targeting vulnerable groups (such as children) may wish to avoid using certain types of persuasion, since the degree of influence that persuasion can have on vulnerable groups can be considered ethically questionable [77].

3.7 Related Work

In the relatively short history of the usable security field, a vast number of novel authentication proposals have been suggested to solve the Password Problem (Section 1.1). Bonneau et al. [24, 25] have published the most recent survey of many such systems. The authors compare the schemes using their evaluation framework for measuring the benefits provided by any authentication scheme that can be used over the Internet, including biometrics [112] and token-based [146]. Their framework can also be used to evaluate authentication support systems, such as password managers (Section 2.6.2) and single sign-on architectures (Section 2.6.1). The authors use the framework to compare the benefits in usability, deployability, and security of a wide array of proposed authentication solutions. Systems are scored as either fully supporting, partially supporting, or not supporting each benefit. Examples of the usability benefits measured include the burden on users' memory, what users need to physically carry, how hard the scheme is to learn, and how easy users can recover from losing or forgetting their credentials. Some deployability benefits include accessibility, compatibility with web browsers and servers, and whether or not the scheme is proprietary. Security benefits include resistance to physical observation, guessing attacks, phishing, theft, and a required reliance on a trusted third-party.

The authors compared multiple authentication schemes and systems with this framework. They made the unsurprising but important observation that no scheme is perfect. Choosing one scheme over another results only in choosing one set of trade-offs over another. Thus, although text passwords have remained the most popular choice for most applications, it is likely not be the best choice for all requirements, contexts, and threat models.

There are two key differences between Bonneau et al.'s framework and our User-Centred Authentication Feature Framework. First, their framework can broadly be used to evaluate any authentication system, while our framework focuses on knowledge-based authentication (KBA) systems. Second, our framework focuses on the user's experience and interaction with an authentication system as a means of enhancing the system's usability and security. Our User-Centred Authentication Feature Framework considers high-level user-centred design features that are largely grounded in human capabilities identified in psychology and cognitive science research. Thus, our framework simplifies the task for authentication designers who wish to incorporate such advances in psychology and cognitive science into their KBA schemes or support systems to achieve their usability and security goals. In contrast, Bonneau et al. measure the end-result of the authentication system design and deployment. Both frameworks take complementary views of authentication systems, and can be used in tandem to either inform the design of a novel authentication system or evaluate proposed systems. To illustrate this complementary relationship, Bonneau et al.'s framework can quite effectively help designers identify the specific benefits (or lack thereof) in a proposed authentication scheme, while our framework can help guide said designers on *how* to design a scheme that will provide the desired benefits by better supporting the user.

3.8 Conclusion

Proposed authentication schemes have a number of user-centred features in common. In this chapter, we identify these features as part of a conceptual framework called the User-Centred Authentication Feature Framework. The features are currently organised into the four classes of persuasion, memory, input and output, and obfuscation.

The primary use of this framework is to provide a novel and constructive way of viewing the authentication scheme space, which can help researchers discover new useful schemes by combining the features in different ways. We illustrate this usage of the framework in Chapters 4 and 5 by identifying the features supported in existing authentication schemes (Cued Click-Points and Persuasive Cued Click-Points) and isolating beneficial features and recombining them to generate novel schemes.

All authentication stakeholders should be able to benefit from this framework. Scheme designers can identify the features a novel scheme under development should or should not support. Administrators and users can compare schemes features to identify the schemes that support particular features of interest. For example, users or administrators with specific needs, such as support for touchscreen devices, can use the framework (Table 3.1) to identify features best supported by touchscreen devices, namely visual memory, visual output, and the touch input modality.

The User-Centred Authentication Feature Framework is also fully extensible. Any features and feature classes we may have overlooked can trivially be added to expand the framework. Furthermore, advances authentication research and technology will invariably lead to novel features and feature classes, which can be easily be added to the framework to assist security professionals and users identify and choose authentication schemes that best suit their needs. Adding these novel features to the framework can also bring them to the attention of authentication scheme researchers and designers for further study and implementation into novel and useful schemes.

Chapter 4

Persuasive Text Passwords

A crucial component to our proposal of a world with many authentication schemes is of course a variety of available schemes. One of the goals of our User-Centred Authentication Feature Framework (Chapter 3) is to isolate beneficial features in one scheme and implement them into another type of scheme. To this end, this chapter describes Persuasive Text Passwords (PTP), our text-based password implementation of persuasive features that we first discovered in the graphical password scheme Persuasive Cued Click-Points [46]. If using our User-Centred Authentication Feature Framework to identify such beneficial features in one scheme and port them to alternative schemes of varying designs, an ecosystem of multiple secure and usable authentication schemes would provide all users with the opportunity to benefit from the best of modern advances in authentication research. Our work on PTP has been published in full papers at Persuasive 2008 [85], SOUPS 2008 [84], and as a poster in the CHI 2008 Student Research Competition [78].

4.1 Introduction

Text password systems are a ubiquitous form of digital authentication. Unfortunately, users often choose insecure passwords [1,74,212]. Attempts at educating users on creating more secure passwords with advice and password restriction policies (Sections 2.2.1 and 2.2.2) have had mixed success. Some users are willing to surpass imposed password restriction policies [126]. Other users either ignore or misunderstand password creation advice [74]. There have been many proposals for improving password security, such as computer-generated passwords and mnemonic phrase-based passwords, but they are found lacking in either usability or security (Section 2.2).

Recent research into the cued-recall graphical password (GP) system Persuasive Cued Click-Points (PCCP) (Section 2.3) demonstrated how persuasion could influence

users to choose more secure passwords. Despite their advantages, cued-recall GPs could pose significant accessibility problems for users with poor eyesight, lack of fine-motor control, and other challenges. To provide such users with similar persuasive benefits as in PCCP, we explored using persuasion with modern text passwords.

We developed a lightweight password creation mechanism named Persuasive Text Passwords (PTP), a persuasive approach to influencing users to create more secure text passwords. Once users choose a password for creation, PTP improves the password's security by placing randomly-chosen characters at randomly-determined positions. Users may *shuffle* for an alternative improvement they may find more memorable. PTP offers a usable compromise between the memorability of user-chosen passwords and the security of randomly-generated passwords. We hope to improve the security of users' passwords while teaching and influencing users to create more secure passwords on their own, even when PTP is not present. In this chapter, we present an informal pre-test, pilot study, and expanded user study of several PTP variations. We use the term *persuasive* in reference to Fogg's work on Persuasive Technology [77], and because we suggest improvements to the user, but allow them to make the final decision.

PTP serves two purposes in this thesis proposal. PTP's first purpose is as a novel security enhancement to existing text password systems which requires minor changes to the client's user interface and no changes on the server backend. PTP's grander purpose is to demonstrate how lessons learnt from an authentication system that uses a particular memory modality can be adapted to a different system using a different memory modality. In this case, the persuasive benefits of PCCP (which uses visual memory) were adapted to text passwords (which use lexical memory) to create PTP. This demonstrates how the same features (persuasion features, in this case) can be integrated into two schemes of different designs.

The remainder of this chapter is organised as follows. We begin with relevant background in Section 4.2. The Persuasive Text Passwords system and its variants are described in Section 4.3. Section 4.4 presents the overall methodology of our initial pilot and expanded user study. We present the results of our initial study in Section 4.5, while Section 4.6 discusses the resulting modifications to our expanded

PTP study design as well as the overall results. The interpretation and discussion of the results are respectively discussed in Sections 4.7 and 4.8. Finally, we offer some concluding remarks in Section 4.9.

4.2 Background

4.2.1 John the Ripper (JtR)

Security practitioners use *password cracking* tools to evaluate the security of passwords created with a given system. Password cracking is typically understood as an attempt to systematically guess as many account passwords as possible for some system. John the Ripper [56] (JtR) is a popular open-source password cracking tool. JtR has three different attack modes. *Single crack mode* uses available data such as login names, full names, and home directory names as candidate passwords and employs a rich set of customisable word mangling rules (such as adding a digit at either end or replacing "a" with "@"). *Wordlist mode* guesses the character strings provided in a user-specified dictionary as candidate passwords, applying word mangling rules as an option. *Incremental mode* attempts to guess passwords in a brute force manner optimised for the user-chosen character set and length as well as using trigraph frequencies to quickly crack as many passwords as possible. JtR was recently used by Weir et al. [212] to attack the leaked RockYou password set with the Dic-0294 wordlist [152]. In Single crack mode, they quickly cracked ≈20% of the RockYou passwords with 7 or more characters, and after a billion guesses, they had cracked over 35%. Kuo et al. [129] cracked 11% of 146 survey-collected regular passwords using Wordlist mode with word mangling rules, and an additional 8% using Incremental mode for 62 hours. St. Clair et al. [186] cracked 25% of 3500 passwords from their Computer Science and Engineering department in 2 hours with a cluster of 16 computers using Incremental mode and 4 using Wordlist mode. All 20 computers were reported to be AMD Opteron 250 processors. Using the Wordlist mode for 22 minutes and Incremental mode for 24 hours, Proctor et al. [164] used a 400-MHz Pentium II computer to crack 34% of 96 passwords from two experiments on password restrictions.

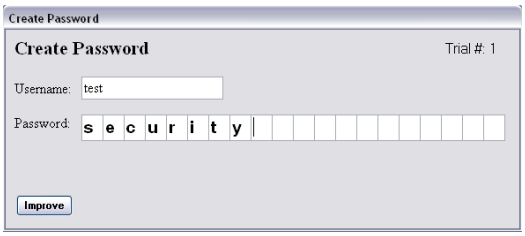


Figure 4.1: PTP password creation before applying the persuasive improvement.

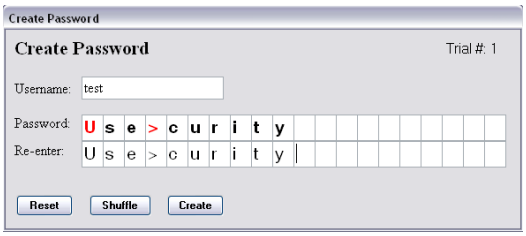


Figure 4.2: PTP password creation after applying the Insert-2 persuasive improvement.

4.2.2 Chunking

Bishop [20] has suggested that passwords should ideally be composed of chunks that have meaning only to the password's creator. Chunking [138] is a well-recognised strategy people use to remember information by grouping several smaller chunks of information into fewer larger ones through some semantic encoding. The 7 ± 2 chunks that Miller [138] suggested could be retained in short-term memory is debatable, as Gobet and Clarkson's [92] survey suggest the number of memorable chunks is closer to three.

4.3 Design

Persuasive Text Passwords (PTP) is a user-chosen text password system which guides users to make their passwords more secure. During password creation, PTP improves password security by placing a few randomly-selected characters at randomly-determined positions in the users' initial password (see Figures 4.1 and 4.2). Users have the option to *shuffle* for an alternative set of random characters and random positions until they deem one is sufficiently memorable. The random characters are chosen with uniform probability across all characters available on English US keyboards.

PTP supports the following features identified in our User-Centred Authentication Feature Framework presented in Chapter 3. Since PTP is largely based on text passwords, the two schemes share many features. Their textual nature clearly relies on users' lexical memory (*Lex*). Users may remember their password with semantic

memory (*Sem*) or episodic memory (*Eps*), depending on the chosen initial password. As with text passwords, we believe that with sufficient practice, users' procedural memory (*Proc*) would facilitate password entry without requiring the conscious retrieval of the password's contents from memory. Both schemes also obscure feedback (*Ofb*) by masking the echoed password on the monitor (*Viso*) during login, which users input with the keyboard (*Kbd*). Text passwords and PTP differ in their use of the persuasion class of features. While text passwords do not leverage any form of persuasion, PTP leverages the following persuasive features (which our framework first identified in PCCP):

- ***Simplification*** (Section 3.2.1). Since the PTP system takes on the responsibility of ensuring the password is secure, users can focus on making their password memorable, thereby simplifying the password creation task. Furthermore, the users' *path-of-least-resistance* is to comply with the system's initial suggestion, which is more secure than shuffling until a weaker set of characters (such as all lowercase characters) are found. Thus, when creating a new password, PTP makes the most secure choice the least burdensome.

- ***Personalisation*** (Section 3.2.2). Since the system-assigned characters are placed in a user-chosen password, users are likely to feel a kinship towards their password and thus are more likely to comply with the system's suggestions. Furthermore, we expect users are most likely to be open to password suggestions when creating one. Thus, PTP applies its persuasion at the most opportune moment. The persuasion may also develop their mental model of secure passwords, potentially leading them to apply the PTP random-character placement scheme to their other passwords.

- ***Conditioning*** (Section 3.2.4). Shuffling repeatedly to find a specific set of system-assigned characters can be tedious. The PTP system makes less secure choices less attractive, hence guiding users away from poor security decisions.

PTP offers a middle-ground between the usability of purely user-chosen passwords and the security of system-assigned passwords. We hypothesise that the PTP strategy

of adding random elements to user-chosen passwords will increase their security while maintaining sufficient memorability. Furthermore, we believe the visibility and user involvement in PTP's strategy may teach users how to improve the security of their passwords for systems that do not implement PTP.

4.3.1 PTP Variations

We devised a number of PTP variants in order to explore this approach to improving password security. Below, we outline several variations which we have implemented and studied. Many others exist that we have yet to examine.

- ***Preload.*** Users are given the system-assigned characters before creating their password. The characters are positioned randomly within the first eight character slots. Users create their password around the system-placed characters.

- ***Replace.*** Users first choose an initial password as they would for a typical password system. The system replaces characters in the users' passwords at random positions with randomly-chosen characters.

- ***Insert.*** After users select an initial password as usual, the system inserts randomly-selected characters at random positions between user-chosen password characters, lengthening the password. See Figure 4.2 for an example of Insert with two system-selected characters.

We ran informal user pre-tests with these three variants. The pre-tests revealed the following problems which we subsequently corrected before running the pilot phase of our study:

- ***Repeating Characters.*** Users would often repeat the system-assigned characters when creating a Preload password. For example, if the system presented users with `__B__#_8`, they were likely to choose a password similar to `BBB###88`. Since the lack of distinct characters makes such passwords very insecure, we chose not to test the Preload variant in our study.

- ***Character Minimisation.*** Users would shuffle until the system placed only two randomly chosen characters into the password. Memorability seemed to suffer the few times users kept three or four system-assigned characters. Therefore, we chose to examine only placing two characters for the pilot study.

- ***Indistinguishable Characters.*** Certain system-chosen characters were difficult for users to tell apart, such as the grave accent (`) and the apostrophe (’), or the lowercase L and the vertical line (|). Users also were confused by the blank space character. To avoid such confounds in our pilot study, we removed the grave accent, vertical line, and space from PTP's set of system-selectable characters, reducing the total number from 95 to 92. We believe the resulting small loss in security is worth avoiding possible user error, confusion, and frustration.

- ***Character Memorability.*** Users had difficulty identifying the position and capitalisation of their password's system-chosen characters when confirming and logging in. Although their password was visible during creation, it was masked with asterisks (*) when confirming and logging in. To assist users in learning their passwords, the pilot study asks users to re-type their password unmasked (shown in Fig. 3) during the password creation phase. This additional step allows users to practice typing their entire password while visually verifying that the characters they type are correct. This assumes PTP passwords are created only in environments free of shoulder-surfing, where potential attackers cannot observe users entering their passwords.

4.4 Methodology

We conducted our study to evaluate the usability and security of PTP in two phases: a smaller pilot and expanded lab study. This section begins by describing the overall methodology used in both phases of the user study. Later sections will elaborate on the methodology and analysis unique to each particular phase. Our procedures for both phases were evaluated and approved by the Carleton University's Ethics Committee for Psychological Research.

Condition	Males	Females	Total
Control	9	10	19
Replace-2	7	9	16
Insert-2	7	9	16
Insert-3	7	9	16
Insert-4	7	9	16
Total	37	46	83

Table 4.1: Number of males, females, and total participants in each condition testing a PTP variant.

Our PTP between-subjects study included a total of 83 participants (Table 4.1). Participants in a given condition used the PTP variant by the same name. The number after the hyphen in the condition name represents how many system-assigned characters were placed into users' passwords. For example, four characters were inserted into Insert-4 participants' passwords. Each participant performed 10 trials. For each trial, users created, confirmed and logged in with a password. Users filled out a demographics and post-test user-opinion questionnaire. The materials used in this study can be found in Appendix A.

All times and user actions, pre-improvement initial passwords, and improved passwords were logged by the system. The experimenter took note of all participant behaviour and comments throughout the session. Before beginning the experiment, participants were asked to pretend the study's passwords were going to protect their online bank accounts, and they should create memorable passwords that would be also hard for others to guess. Participants were also told that PTP's password improvement would help them create more secure passwords, but they could shuffle as often as desired to find a character arrangement they felt was memorable. Participants familiarised themselves with creating PTP passwords by performing a practice trial (which are excluded from the 834-trial data). A trial consisted of five steps:

1. ***Create.*** Users would first enter a password of their choice. The system then placed random characters appropriate for the PTP variant into the users' password (see Section 4.3.1). Users were allowed to *shuffle* the characters as much as they liked. Users could press the *Reset* button if they wished to change their

initial password. Once they found a password and system-assigned character combination that they liked and felt they could remember, they then re-typed the improved password on the second row and pressed the *Create* button. As shown in Figure 4.2, the participants' passwords were visible during password creation. We learned from the pre-tests that users found it beneficial to see the system-assigned characters and re-type their improved password, ensuring they could correctly identify and type the extra characters.

2. ***Confirm.*** To confirm their password, users re-entered their improved password, which was masked with asterisks. If they made a mistake but felt they knew their password, they could retry to confirm. However, if they thought they forgot their password, they could end the trial.

3. ***Questions.*** Users were asked how easy they felt it was to create their password and how difficult they thought it would be to remember in one week. Participants responded on a 10-point Likert scale, from very easy to very difficult.

4. ***Distraction.*** For 45 seconds, users would count down in threes from a randomly chosen four-digit number. This type of distraction flushes their textual working memory [162] and simulates a longer passage of time by focusing participants' attention on a separate cognitively-difficult task.

5. ***Login.*** Participants attempted to login with their improved password, which was echoed with asterisks. If they made a mistake, they could try to login again, or end the trial if they forgot their password.

Before performing any statistical tests on the data, the data from a participant's trials is aggregated into an average (or counts, in the case of Boolean data, such as confirm or login successes). This ensures that each participant only contributes a single *independent* data point in each test. We use a variety of statistics to determine if two or more distributions are significantly different. For normal distributions, we use t-tests and one-way ANOVAs. When the data is not normally distributed, we use Mann-Whitney U and Kruskal-Wallis (KW) tests instead. For categorical data, we primarily use chi-squared or Fisher's Exact tests depending on category counts. The

Bonferroni correction is applied as appropriate when doing multiple comparisons. In all tests, we accept $p < .05$ as statistically significant.

4.5 Pilot Study

The purpose of our pilot study was to determine whether Replace or Insert PTP variants better helped users create more secure passwords without sacrificing too much usability. The two specific conditions were Replace-2 and Insert-2, wherein two characters would either replace two existing characters in the user's password or be inserted in between. We recruited 15 participants; 7 participants tested the Replace-2 PTP variant and 8 others tested Insert-2. All were university students; 8 studying Computer Science (CS) and others from various disciplines (non-CS). All used computers, the Internet, and passwords regularly. Data was collected from a total of 154 trials. When choosing an initial pre-improvement password, the only password requirement was a minimum of 8 characters, since many text password systems in use have a similar length requirement [74, 75, 87].

4.5.1 Results

In considering the pilot study's results, we adhere to the twin goals of usable security. We must care for usability; the system should be easy to use for the purpose intended. In this case, created passwords must be memorable, and the creation process should not be time-consuming. However, the system should also accomplish the security goals. In this case, the resulting passwords should be sufficiently stronger than those originally chosen by the user. We now address both these issues.

Memorability: *How memorable were the passwords?* Users created 154 passwords, and successfully confirmed 94% of the time, 86% of the time on the first attempt. Of the 145 successful confirmations, 97% also resulted in a successful login, of which 88% were on the first attempt. Fisher's Exact Test reported no significant distinction in confirm or login successes between the Insert-2 and Replace-2 conditions. Although our method did not test long-term memorability, the cognitive load was substantial due to the many passwords being created. Overall, we feel these results suggest a reasonable level of memorability, but warrants further testing.

Times: *How long did it take to create, confirm, and login?* Users took a mean and median of 66 and 58 seconds to create a password, with a standard deviation of 31 seconds. This time includes constructing an initial password, shuffling as much as desired, and re-entering the improved password. The mean times to confirm and login were 15 and 14 seconds respectively. A regression analysis showed login times became shorter as users completed more trials. T-tests reported no significant difference between Insert and Replace for create, confirm, or login times. We believe these times are an acceptable starting point and are likely to decrease with daily use.

Shuffles: *How often did users shuffle improved password suggestions?* Over all 154 trials, the mean number of shuffles per trial was 8, and the median was 3. Our observations suggest that most users shuffled until they found an improved password they felt was memorable. However, users were often persuaded to comply with the first system-suggestion, as they did not shuffle at all in 41 of 154 trials. A Mann-Whitney U test found no significant difference in shuffles between Insert-2 and Replace-2.

Perception. *What opinions did users report about the usability and security of PTP?* Our post-test questionnaire asked users to rate on a 10-point Likert scale how much they strongly disagreed (1) to strongly agreed (10) with several statements. Some statements were reversed to avoid bias. These results present the median responses such that higher scores are always in favour of PTP. Users felt that the new system was slower (4), they would prefer a normal system if they were in a hurry (4), and they would find the passwords more difficult to remember (4). However, they felt the improved passwords were more secure (8) and more difficult to guess (8) than ordinary passwords, and with practice, they could get used to the system (8). These results seem reasonable for a new authentication system, and reflect a trade-off we would expect. However, participant responses may be biased from being told the system would improve their passwords' security. Chi-squared tests revealed no significant distinction between Insert and Replace.

Security. *Did the system help users create more secure passwords?* Table 4.2 shows the system's effect on password security by describing the nature of both the pre- and post-improvement passwords. For example, our improvement process reduced the proportion of very weak lowercase alphabetic passwords from 20.8% to

Character Subsets	% of Original Pwds	% of Improved Pwds
All lowercase	20.8%	4.5%
All uppercase	0.6%	0.0%
All numeric	0.0%	0.0%
All special	0.0%	0.0%
Lowercase & uppercase	1.3%	5.8%
Lowercase & numeric	26.6%	5.8%
Lowercase & special	5.9%	7.2%
Uppercase & numeric	2.0%	0.0%
Uppercase & special	2.6%	0.0%
Numeric & special	0.6%	0.0%
Lowercase, uppercase, & numeric	14.3%	16.9%
Lowercase, uppercase, & special	3.2%	9.7%
Lowercase & numeric, & special	7.8%	16.9%
Uppercase & numeric, & special	0.0%	3.3%
All four character subsets	14.3%	29.9%
Passwords in 1 subset	21.4%	4.5%
Passwords in 2 subsets	39.0%	18.8%
Passwords in 3 subsets	25.3%	46.8%
Passwords in 4 subsets	14.3%	29.9%

Table 4.2: Original and improved password distributions across character subsets.

4.5%. The proportion of very strong passwords that included lowercase, uppercase, numeric, and special characters rose from 14.3% to 29.9%. Most passwords were improved to include characters from additional character subsets.

$$[tb]H = l \times log_2(b) \tag{4.1}$$

As a rough estimate of *bits of security*, we evaluate password security strength using the formula shown in Equation 4.1, where l is the password length and b is the size of the alphabet from which the password's characters were chosen. This metric was used by Florêncio and Herley [75] in their evaluation of password restriction policies. Although this metric is useful as a starting point for relative comparison, it over-estimates password security. It assumes that individual password characters are chosen randomly and independently from one another. Thus, the metric does not account for user-biases towards English words, predictable character positioning, or relationships between adjacent characters. For example, password containing 5

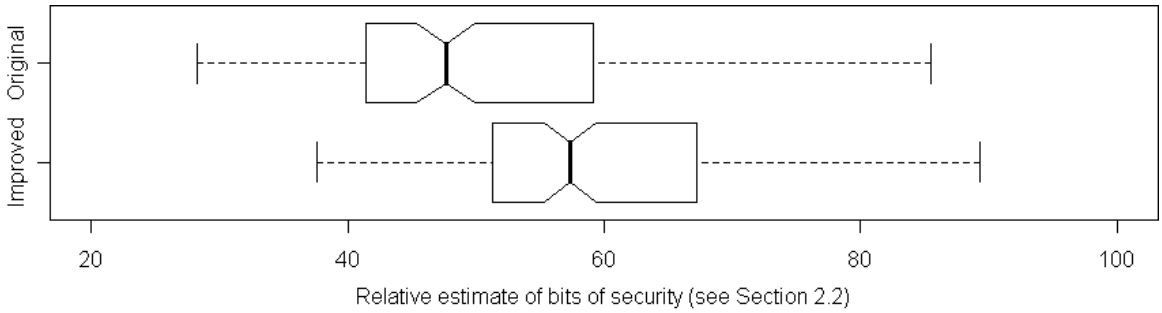


Figure 4.3: Distribution of bits of security of PTP pre- (original) and post-improvement (improved) passwords.

lowercase letters (each with 26 possible choices) and 1 uppercase letter (with 26 other possible choices), is assigned the password a measure of $6 \times log_2(52)$ bits. Nonetheless, we use this model for its simplicity to provide a preliminary security evaluation, and there is no accepted measure that is more promising.

The boxplot in Figure 4.3 compares the distribution of users' original and improved passwords' bits of security (Section 2.2). The bold line down the middle of each box is the median, and the left and right halves of boxes denote the 2^{nd} and 3^{rd} quartiles. The length of this box is known as the *interquartile range* or IQR. The whiskers at either end of the plot denote the 1^{st} and 4^{th} quartiles. Any circles beyond the whiskers are outliers which are at least $1.5 \times IQR$ away from the median. Mann-Whitney U tests confirm the improved passwords were statistically stronger across all participants ($U = 10, p < .0001$), and within both the Insert-2 ($U = 4, p < .005$) and Replace-2 ($U = 0, p < .001$) conditions. Thus, PTP influenced users to create passwords more difficult for attackers to guess.

Our user sample consisted of approximately half Computer Science (CS) students, who appear to have formed a distinct group. For example, 12% of CS students chose original passwords in all lowercase, while 30% of the other students did the same. Thus, we believe the possibility for improvement was reduced for CS students. Since such technical expertise was over-represented in our sample, we expect that the general impact of our approach would likely be greater than our study results suggest.

Correlations. In addition to the results above, we wished to determine if there were any correlations between said results. We believed correlations between results

(or a lack thereof) may suggest directions for the expanded study or improvements to PTP. In examining the correlations, we found that the sum of time taken to confirm and login (confirm + login time) was a suitable password usability measure since a linear regression test revealed a strong correlation with the number of confirm and login errors ($F(1, 152) = 339.3, p < .001$), and the linear model showed the recall time increased as the confirm and login errors increased. Admittedly, it is not surprising that users spent more time entering their password when they had to recover from errors. However, this emphasises the need to minimise authentication errors to provide a timely password entry, as well as avoid user frustration.

PTP's actual security improvement can vary, since the added characters will come from varying password spaces (Section 2.2). Thus, we first examined whether the amount of shuffling had any effect on the difference in bits of security between their original and improved passwords. A linear regression test found no correlation ($F(1, 152) = .04914, p = .825$). We also found no correlation between the amount of shuffling and confirm + login time ($F(1, 152) = .3144, p = .576$). These results show that users could chose memorable characters without sacrificing password security.

We next compared confirm + login time to the difference in bits of security between improved and original passwords, finding no correlation ($F(1, 152) = 1.011, p = .3164$). We also found no strong correlation between confirm + login time and the improved passwords' bits of security ($F(1, 152) = 3.532, p = .06211$). These results suggest PTP did not affect the improved passwords' memorability.

To our surprise, we found a strong correlation between confirm + login time and users' original passwords ($F(1, 152) = 9.191, p < .005$). The linear model revealed that stronger original passwords were correlated with longer confirm + login times, implying the improved passwords' memorability suffered. This matches comments from participants who created more secure original passwords, stating that PTP made their passwords less memorable by interfering with their password creation method.

4.5.2 Summary

Our 15-participant pilot study explored the Replace-2 and Insert-2 PTP variants, wherein two characters would either replace or be inserted in between two existing

characters in the user's password. Half of our participants were Computer Science (CS) students, who usually created more secure initial passwords than participants studying other disciplines (non-CS). However, CS students had more difficulty with PTP, since the memorability of their PTP-improved passwords declined as they chose more secure pre-improvement passwords. We had expected non-CS users to have some difficulty with PTP, but their success rates were reasonably high.

4.6 Expanded Study

Our pilot study showed that PTP successfully helped people choose secure and memorable passwords. However, the pilot's small sample size was one half Computer Science students, which is a much larger proportion than we would expect in the general population. Thus, our expanded study's participants totalled 83 university students studying across various disciplines. Since both study phases were executed following identical procedures, the data from the 7 Replace-2 and 8 Insert-2 pilot participants are included in this 83-participant study. All expanded study users were familiar with using computers, the Internet, and passwords.

Some anecdotal evidence from our pilot study suggested that Replace-2 was slightly more difficult for users than Insert-2. Also, adding characters increases password security more than replacing characters. Since the Insert variant seemed superior to Replace in terms of both memorability and security, our expanded study includes Insert-3 and Insert-4 conditions to test users' password memory when more (three or four) characters are inserted, but not replaced. We also added a Control condition, wherein users' initial passwords were not modified by PTP, to serve as a baseline comparison between PTP and standard text passwords. The number of participants in each experimental condition for the expanded study are shown in Table 4.1.

Another difference between conditions was the minimum length requirements of users' pre-improvement initial passwords. Participants in the Control and Replace-2 conditions were required to enter a minimum 8-character initial password while Insert-2 participants had to choose a minimum 6-character initial password, which would become an 8-character improved password (post-improvement password). This facilitates comparisons between each of these conditions, since all their improved

	Success %		Significant differences versus Control	
Condition	Confirm	Login	Confirm	Login
Control	99.5	98.4	-	-
Replace-2	91.9	92.6	n.s.	n.s.
Insert-2	93.9	99.3	n.s.	n.s.
Insert-3	81.9	97.7	$U = 232.5, p < .01$	$U = 229, p < .05$
Insert-4	92.5	93.9	n.s.	$U = 244, p < .01$

Table 4.3: Confirm and login success rates (shown as percentages) and significant differences versus Control across PTP conditions.

passwords are at least 8 characters long. We likewise considered setting the minimum initial password length to 5 and 4 characters for the Insert-3 and Insert-4 conditions. However, we felt the three Insert conditions would be more comparable if they all required at least six characters, as well as better emulating contemporary password policies.

Other than the differences mentioned above, the experiment was carried out in the exact same manner as the pilot study for all conditions. The 83 participants collectively completed a total of 834 trials.

4.6.1 Results

We now present the results and a brief interpretation.

Successes

Table 4.3 shows the percentage of trials wherein participants were able to successfully confirm and login. The table also shows the Mann-Whitney U significance test results (with the Bonferroni correction) comparing each condition to Control. Control participants logged in significantly more often than Insert-3 and Insert-4 participants, but not Insert-2 or Replace-2. This suggests that Insert-3 and Insert-4 passwords may have been too difficult for participants to remember. However, Insert-2 and Replace-2 passwords seemed to be as memorable as Control passwords. We will later discuss reasons for the anomalous result of the Insert-4 confirm success rates being higher than those for Insert-3, despite the former being the more difficult condition.

	Mean			Median			Standard Deviation		
Condition	Create	Confirm	Login	Create	Confirm	Login	Create	Confirm	Login
Control	33.2	8.7	11.4	27.9	7.3	7.8	20.5	5.6	12.0
Replace-2	67.0	14.5	16.3	56.2	9.8	11.7	37.2	15.7	15.5
Insert-2	65.0	20.0	17.1	55.3	11.8	12.9	34.0	25.5	16.4
Insert-3	65.0	21.6	20.8	57.9	12.7	13.3	26.7	22.2	35.4
Insert-4	98.3	25.1	28.4	81.5	16.4	16.7	66.1	23.3	44.4

Table 4.4: Seconds taken per trial to complete the experiment phases across PTP conditions.

Timings

Table 4.4 shows the time participants took to complete each step of a trial. Participants in the Control group took a mean of 33 seconds to create a password. By shuffling for acceptable system-chosen characters, all PTP users took approximately 65 seconds on average to create their passwords, twice as long as the Control group. The notable exceptions are Insert-4 users who took about 98 seconds, over three times as long as Control participants. We noted that Insert-4 participants would stare at their password at length after they had finished shuffling, in an attempt to memorise it. We believe that 65 seconds is an acceptable creation time for long-term passwords. Furthermore, as we discuss later in the *User Perception* section, the passage of time is not noticeable when creating a password, presumably because users are cognitively active when choosing a password and shuffling.

Shuffles

Table 4.5 shows the amount participants shuffled per trial in each condition. Participants in more difficult conditions shuffled more. Regarding Insert-4, the high standard deviation and median lower than the mean shows that the mean is inflated by a small number of trials where participants shuffled an exceptional number of times.

Errors

Table 4.6 demonstrates confirm and login errors committed by participants per trial. The medians of 0 errors show that participants successfully confirmed and logged

	Shuffles		
Condition	Mean	Median	StD
Control	-	-	-
Replace-2	5.7	3.0	9.3
Insert-2	8.4	2.0	19.3
Insert-3	10.2	6.0	10.0
Insert-4	18.0	6.5	51.8

Table 4.5: Mean, median, and standard deviation of number of shuffles per trial. *StD* is short form for *standard deviation*.

	Errors					
	Mean		Median		Standard Deviation	
Condition	Confirm	Login	Confirm	Login	Confirm	Login
Control	0.1	0.1	0	0	0.4	0.6
Replace-2	0.3	0.3	0	0	0.7	0.9
Insert-2	0.4	0.1	0	0	1.0	0.6
Insert-3	0.6	0.3	0	0	1.2	1.7
Insert-4	0.4	0.3	0	0	0.9	1.1

Table 4.6: Number of errors made per trial during confirm and login across PTP conditions.

in on their first attempt for the majority of trials. Therefore, when users did commit an error, they were likely to retry to confirm or login multiple times before either succeeding or giving up. Kruskal-Wallis tests revealed a significant difference amongst the number of confirm ($KW\chi^2(4) = 16.60, p < .005$) and login errors ($KW\chi^2(4) = 12.17, p < .05$) across conditions. Mann-Whitney U tests revealed that Insert-2 ($U = 55.5, p < .05$) and Insert-3 ($U = 49.5, p < .01$) users made significantly more confirm, but not login, errors than Control participants. Users sometimes had trouble confirming, but once they did so successfully, they typically were able to recall their password to login. The exception is Insert-4 participants, who made significantly more login errors than Control participants. These findings support our conclusions regarding Insert-4 participants' significantly lower success rates.

Password Space

The set of all characters can be split into four classes: lowercase, uppercase, numeric, and special characters (or symbols). Many users choose passwords from a single class, usually lowercase letters [74]. A password containing characters from fewer classes will be easier for attackers to guess than a password spanning more classes. These character classes can be combined to form distinct *password spaces*. To illustrate, the password space of all lowercase and numeric passwords contains all passwords with at least one lowercase and one numeric character, but no uppercase or special characters. Given all possible combinations of the four classes mentioned above, there are a total of 15 distinct password spaces.

Although PTP places randomly-selected characters into users' passwords, users may shuffle to obtain new characters. Thus, a user with an all-lowercase password may shuffle until the system randomly selects only lowercase letters. Such a password would be vulnerable to a brute force attack on all-lowercase passwords. To evaluate the effectiveness of PTP in defending against such attacks, we examined the types of characters found in users' initial and improved passwords.

The password spaces of initial and improved passwords of each condition are shown in Figures 4.4, 4.5, 4.6, and 4.7. Each graph shows the percentage of passwords in each password space for each PTP variant condition. The x-axis represents all character class combinations of lowercase letters (L), uppercase letters (U), numeric characters (N), and symbols (S). The y-axis represents the percentage of passwords from the given condition that fall into the password space on the x-axis. The line connecting the points corresponding to a password type (either initial or improved) is only for ease of comparison and does not represent a continuum.

Figures 4.4, 4.5, 4.6, and 4.7 show that PTP users predominantly chose initial passwords containing either only lowercase (L), lowercase and numeric (LN), or lowercase, uppercase, and numeric characters (LUN). Insert-3 and Insert-4 participants more often created initial lowercase-only passwords. However, for all conditions, we see that there are very few PTP-improved passwords containing only lowercase or only lowercase and numeric characters. This shows that users were usually influenced to create passwords containing characters from more than one class, since PTP's

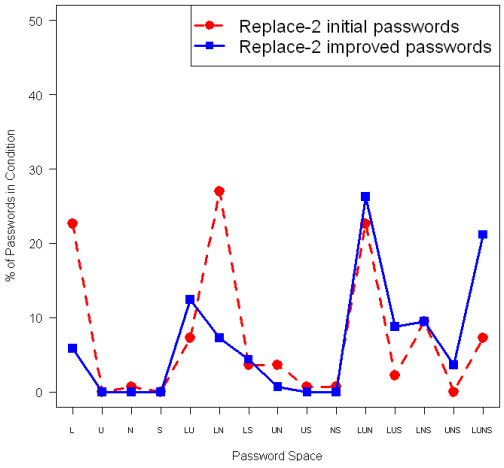


Figure 4.4: Percentage of Replace-2 initial and improved passwords that fall into various password spaces.

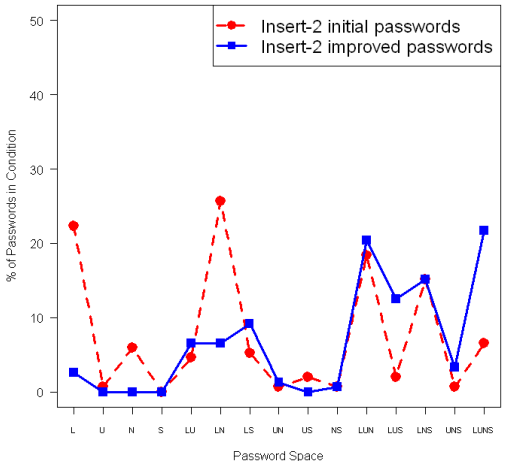


Figure 4.5: Percentage of Insert-2 initial and improved passwords that fall into various password spaces.

random character selection algorithm was unbiased and would allow users to create passwords consisting of characters from a single class. This also suggests that accounts protected by PTP passwords would most likely not be compromised in an obvious and theoretically efficient attack by guessing all passwords containing only lowercase and/or numeric characters.

Furthermore, in all four figures, we can see a significant increase between the number of initial and improved passwords in the larger spaces of lowercase, uppercase, and special (LUS) ($U = 895, p < .0001$), and lowercase, uppercase, numeric, and special ($LUNS$) characters ($U = 944.5, p < .0001$). Thus, all PTP variants improved the security of passwords by adding characters that promoted the users' passwords into larger password spaces.

Estimate of Bits of Security

To evaluate the security of users' initial and improved passwords, we use the same metric as in the pilot study (Equation 4.1). The initial and improved passwords' estimated bits of security H for the five conditions are shown in Table 4.7. The *Delta* columns show the increase in H as a result of PTP's improvement. Although Insert initial passwords had to be at least six characters, Insert-2 users often chose longer initial passwords. This is why the estimated security bit values for the Insert-2 initial

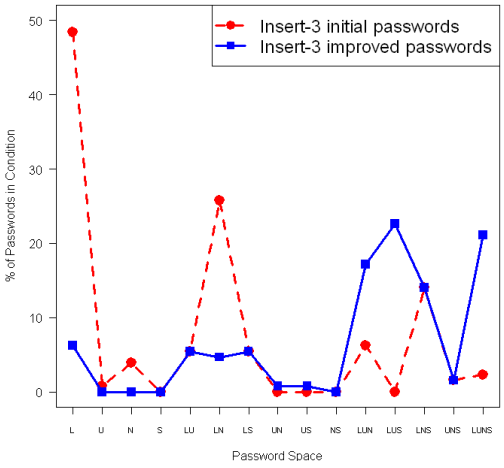


Figure 4.6: Percentage of Insert-3 initial and improved passwords that fall into various password spaces.

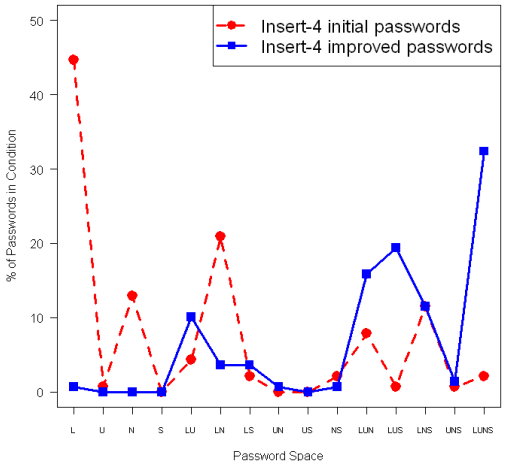


Figure 4.7: Percentage of Insert-4 initial and improved passwords that fall into various password spaces.

passwords is not much lower than for the Control initial passwords, which had to be at least 8 characters long.

All three Insert conditions resulted in improved passwords having significantly more estimated bits of security than the Control and Replace-2 conditions ($U = 1418, p < .0001$). However, the three Insert variations produced improved passwords of similar estimated security ($KW\chi^2(2) = 1.5045, p = .4713$). In fact, participants who had more characters inserted into their password chose to create initial passwords with noticeably fewer estimated bits of security ($KW\chi^2(2) = 8.6505, p < .05$). Insert-4 participants in particular created initial passwords with significantly lower estimated security as they completed more trials ($F(1, 158) = 8.17, p < .005$). Reasons for this phenomenon are further discussed in Section 4.7. Similar but weaker correlations between estimated initial password strength and completed trials were found for Insert-2 and Insert-3, but not for Control and Replace-2.

John The Ripper (JtR)

We ran various John the Ripper [56] dictionary attacks on the initial and improved passwords. We used the Bartavelle-patched version of JtR [12], to work with SHA-1 encrypted passwords. Our first dictionary attack, *All+Rules*, used the largest free word list available on the JtR website (`all.lst`), containing about 4 million entries.

	Bits of Security								
	Mean			Median			Standard Deviation		
Condition	Initial	Improved	Delta	Initial	Improved	Delta	Initial	Improved	Delta
Control	51.6	-	-	47.6	-	-	13.7	-	-
Replace-2	51.5	56.7	5.2	48.9	53.6	4.9	10.8	11.4	5.8
Insert-2	49.3	67.8	18.5	46.0	61.1	17.8	20.5	23.4	7.2
Insert-3	42.1	68.1	26.1	36.7	64.1	26.1	14.6	15.9	6.1
Insert-4	35.5	69.3	33.8	31.0	65.6	32.9	12.6	13.2	5.5

Table 4.7: Initial, improved, and delta (the difference between improved and original) estimated bits of security H for the five conditions.

We also enabled JtR's built-in word mangling rules, which guesses predictable variations of the words in the provided list. Our second dictionary attack, *Mangled*, used the `mangled.lst` word list purchased from the JtR website. Containing over 40 million entries, this list contains all the words in `all.lst`, in addition to pre-mangled variations and various extra Latin alphabet-based dictionary words.

We ran two rounds of attacks with each dictionary. We attacked passwords from each condition separately and together (*All*). Table 4.8 displays the number of total, cracked, and percentage of cracked initial passwords from these attacks. The Mangled attack cracked slightly more initial passwords than the All+Rules attack in every condition except Replace-2. We believe this difference is because Replace-2 and Insert original passwords were respectively 8 and 6 characters long. Examining the two dictionaries, it appears that `All.lst` contains a larger proportion of longer words than `Mangled.lst`. Thus, it is possible the All+Rules attack may be more effective at cracking longer passwords in general than the Mangled attack.

None of the dictionary attacks cracked a single password improved by any PTP variation. This is likely because the JtR dictionaries do not include words formed by randomly inserting characters in common words. Of course, considerably larger dictionaries could be built to specifically crack such passwords.

User Perception

Participants rated statements on our post-test questionnaire on a Likert scale of 1 (strongly disagree) to 10 (strongly agree). To cross-validate the users' responses, the questionnaire was composed of four statements for each general topic of user

		All+Rules		Mangled	
Condition	Total # of Passwords	Cracked	Percent	Cracked	Percent
All	1668	202	12.1	251	15.0
Control	190	18	9.5	36	18.9
Replace-2	161	24	14.9	21	13.0
Insert-2	163	23	14.1	28	17.2
Insert-3	160	65	40.6	75	46.9
Insert-4	160	54	33.8	55	34.4

Table 4.8: Pre-improvement passwords cracked by various John the Ripper dictionary attacks. None of the attacks were able to guess any password improved by any PTP variation (see discussion in text).

perception we measured. To control for bias from the statements' structures, each of the four statements for each topic were constructed to be either posed in a normal or reversed manner, and to be either comparing PTP to normal passwords or have no comparison. The statements of all topics were randomly ordered on the questionnaire.

Figures 4.8, 4.9, 4.10, and 4.11 are notched box plots describing the distribution of participants' Likert score responses to four aggregate questions measuring users' perception regarding ease of password creation, improved password guessability, how much PTP helped them create more secure passwords, and login speed. Whenever the notches, which are the angled sections of the box plots, for two groups overlap relative to the x-axis, there are no statistically significant differences between the two groups. Since the notches overlap, Figure 4.8 shows that PTP users and Control participants felt they had the same degree of difficulty creating passwords. The PTP variant notches in Figures 4.9 [1] and 4.10 do not overlap with the Control group notches, showing that the participants felt their PTP improved passwords were significantly less likely to be guessed by an attacker, and that PTP helped them create significantly more secure passwords than participants in the Control group. Finally, the overlapping box plot notches in Figure 4.11 show that PTP users in all conditions did not feel that the login time was longer than those in the Control group. Oddly,

[1] The Replace-2 box plot in Figure 4.9 appears different because all participant responses in the 2nd quartile were equal to the median. The notch is drawn in the conventional manner, indicating the significant difference with the Control group, due to the lack of overlap between the two groups' notches.

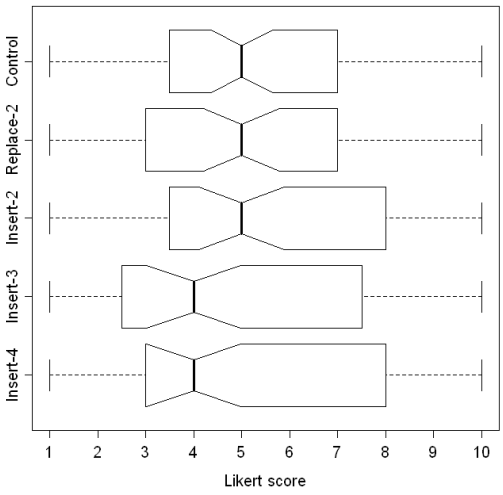


Figure 4.8: Participants' perception of ease of password creation (1 is very difficult, 10 is very easy).

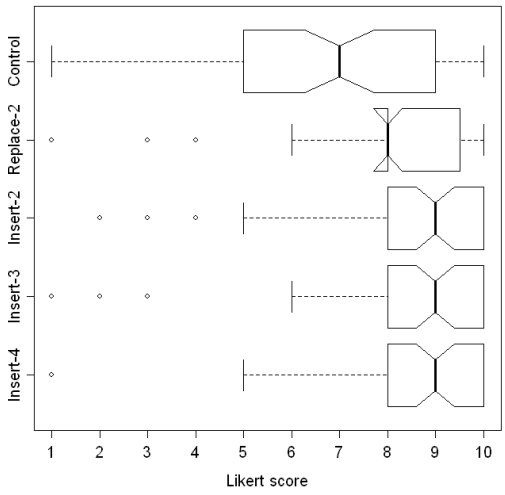


Figure 4.9: Participants' perception of PTP password guessability (1 is very guessable, 10 is very difficult to guess).

Insert-2 users perceived that the login time was *faster* than Control participants, since the two box plot notches do not overlap. These results suggest that the longer login times for PTP did not have a negative impact on users' perception of the system.

4.7 Interpretation

We now interpret the results by examining the effects PTP had on participants.

How does PTP affect password security? In all conditions, PTP improved the security of users' passwords according to our metric. Users' initial passwords characters were mostly lowercase characters, and the odd numeric or uppercase character. The majority of users' improved passwords included at least three classes of characters. All three Insert conditions' improved passwords to similar estimated security strengths, despite some PTP variants adding more characters than others.

How does PTP affect users as the memory load is increased? The mean number of errors in all conditions shows users sometimes had difficulty confirming their password. However, the medians of 0 show that the majority of those errors occurred on a small number of trials. This means that participants were usually able to successfully confirm without error. Furthermore, Replace-2 and Insert-2 participants we able to confirm and login as easily as Control participants, suggesting

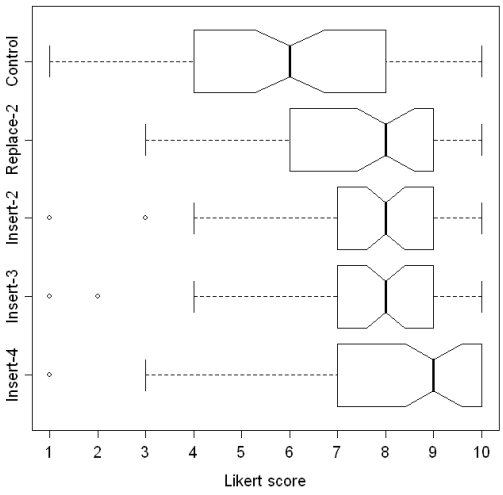


Figure 4.10: Participants' perception on PTP's assistance in creating more secure passwords (1 is not helpful, 10 is very helpful).

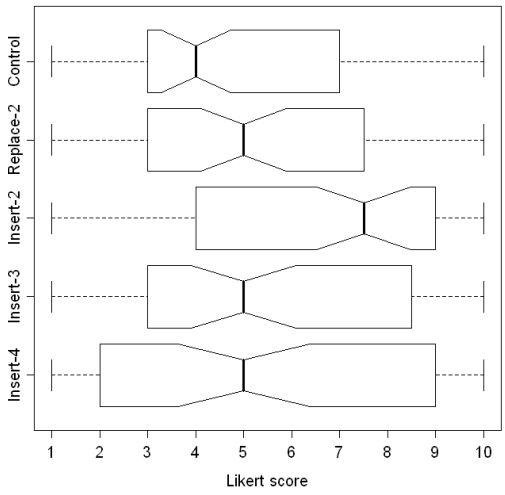


Figure 4.11: Participants' perception of login speed with PTP passwords (1 is very slow, 10 is very fast).

that users' memory may be able to handle 2 random characters, but no more. These memorability results should be confirmed with a field study as well as testing for long-term password memory and interference from multiple passwords.

The median login time in our most difficult condition (Insert-4) is 16.7 seconds. While this is 9 seconds longer than the Control group, we believe it is still acceptable, considering the increase in password security. Furthermore, all PTP users (including Insert-4) reported perceiving the login time as equivalent as or faster than Control group participants reported. Most users also felt that they would be able to login more quickly with practice, and no one mentioned login time during our observations.

Figure 4.12 contains three plots showing various phenomena across the Insert conditions to directly compare the effect of inserting more characters. The graph on the left shows the mean confirm and login errors committed by participants per trial, as previously shown in Table 4.6. Although we expected the number of errors to increase for more difficult conditions, we see a significant drop in the number of confirm errors per trial between Insert-3 (approx. 0.6 errors) and Insert-4 (approx. 0.35 errors). The two reasons for this lie in the other two graphs. The second graph shows the mean times to create, confirm, and login. Notice the steep increase in creation time between Insert-3 (approx. 60 seconds) and Insert-4 (approx. 100 seconds), indicating

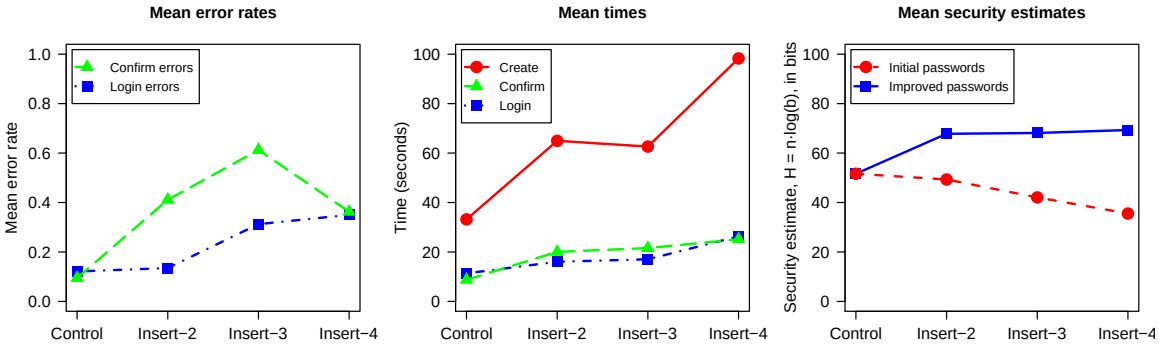


Figure 4.12: Summary of mean times, error rates, and a crude relative security estimates across conditions. The lines are present for ease of interpretation and do not represent continuous values.

that users in this condition needed much more time to memorise their password. Observations during the session support this, as Insert-4 users would gaze at their password for long periods of time before completing the password creation. We believe that this intense effort taken by Insert-4 users to memorise their passwords during creation is the first reason they made significantly less confirm errors. The second reason lies in the graph on the right, which shows the mean estimated bits of security (using the model from Section 4.6.1) for the initial and improved passwords. This shows that participants created less secure initial passwords as more characters were inserted, especially in Insert-4 (under 40 bits, according to our metric). Among the more interesting results to us, the strength of improved passwords throughout all three Insert conditions is relatively constant (approx. 68 bits, according to our metric). Therefore, the additional burden of extra inserted characters did not result in greater password security, because users increasingly compensated for the cognitive burden of the additional random character insertions by creating weaker initial passwords. This suggests that there may be no benefit to inserting more than two characters.

How does PTP affect users' understanding of how to create secure passwords? Results from our questionnaire indicated that users in all PTP conditions felt that the system was helping them create more secure passwords. Furthermore, experimenter observations and other post-test questionnaire responses suggest that most users learnt new ways of making their passwords more secure. Many of them also mentioned that they would try to employ the random character insertion

strategy to their own passwords for systems where PTP is not implemented. Many participants had never previously considered using symbols in their passwords and seemed previously unaware such symbols were available on their keyboard.

As previously discussed, users create less secure passwords if cognitively overburdened. This suggests a possible danger; users may become dependent on PTP and create even less secure passwords than usual with non-PTP systems. We must be careful that users do not rely solely on PTP to make their passwords secure.

4.8 Discussion

We now consider the implications and overall effect of PTP. Since PTP is a password creation scheme, we address only its usability and the resulting passwords' security. We do not address the security of any systems between and including the client and server, nor social engineering attacks.

A benefit of PTP is its lightweight design. Implementation of PTP should only require minor changes to the client-side password creation module, and no modifications are necessary on the server side. PTP requires changes made to the password creation process, but none to the login process. PTP is also highly resistant to password reuse, which may be viewed as both a security advantage and usability disadvantage. It is highly improbable that PTP will place the same characters in the same positions in two identical initial passwords. This may increase memory interference effects from multiple passwords if users choose the same initial password for multiple passwords.

A drawback of PTP is password visibility during creation. This makes PTP suitable only when users will be creating passwords in environments free of shoulder-surfing. On the other hand, since passwords are never displayed after their initial creation, PTP is no more vulnerable to shoulder-surfing than regular passwords during confirm or login. This is a notable advantage over PTP's graphical password predecessor PCCP, where logins are vulnerable to shoulder-surfing.

The main idea of PTP is a middle-ground approach between system-generated and user-chosen passwords. The addition of random elements to users' passwords may create more unpredictable passwords than other methods, such as mnemonic phrase-based passwords. Our study has also highlighted some limitations in our approach.

Most importantly, we found that in our stronger conditions, users compensated by selecting less secure initial passwords. We conclude that we must be careful to not overburden users as we help and show them how to behave more securely. For PTP, we recommend inserting no more than two random characters into users' passwords.

In our pilot study, we found that people with stronger initial passwords (such as those already containing special characters) had trouble remembering their improved passwords. By expanding the pilot study, our larger sample population had a smaller proportion of computer experts ($8/83 = 9.6\%$), and was more representative of typical users. As such, we cannot verify the previous result of the pilot study. However, PTP could be designed to accept sufficiently strong initial passwords without attempting to improve them further. PTP could also insert only characters from classes that are not present in the initial password, which are usually symbols, numbers, or uppercase letters. This character selection bias would result in more PTP passwords including more character sets. However, it would reduce PTP passwords' theoretical (and also effective) security, since password crackers may narrow the set of candidate passwords based on the smaller character sets from which random characters are selected, making the passwords easier to crack. Further examination of the security and usability implications of these modifications is left for future work.

We expect that the amount of user-choice in PTP would contribute to password memorability. While completely random passwords would be more secure than those created with PTP, we believe that PTP helps users create more memorable passwords, though this has yet to be tested. Of course, knowing that a system uses PTP and knowing how PTP works will allow attackers to refine their cracking strategies. For example, a potential modification to John the Ripper would be to search for all words in a list, with two random characters inserted in all possible places. Thus, as a novel scheme, inserting only two random characters seems sufficient for foiling enhanced dictionary attacks. However, should PTP gain popularity, attackers are more likely to build dictionaries and password crackers to specifically target PTP passwords. In such a case, inserting additional random characters and/or increasing the minimum initial password length may be necessary in order to make PTP password cracking more computationally expensive.

We believe that the general idea of PTP would be most effective, not as a single scheme for password improvement, but rather in allowing many different variations of several password schemes (in addition to PTP). This would limit the number of assumptions attackers can make about the authentication system, potentially increasing the work involved in guessing attacks.

4.8.1 Cued Text Passwords

Admittedly, users may have difficulty remember multiple PTP passwords, given the random nature of the added characters. Fortunately, Persuasive Cued Click-Points (PCCP) [46] has another feature that may increase password memorability: Cueing (Section 3.3.4). Cued-recall graphical passwords [19], such as PCCP, have found an increased password memorability as a result of presenting users with the same cue(s) at login that users saw during password creation. We will now explore how an analogous cueing feature could be implemented for text passwords. Our goal in this exercise is provide users with the support to more easily remember their credentials (without compromising security), so that users could hopefully remember more secure credentials with little to no more additional effort than they use to remember their existing cue-less passwords.

Cued Text Passwords (CTP) would incorporate cueing by presenting the user with a different textual cue for every character they type, just as PCCP users are shown a different image for every click. We hope the visual stimuli of the cue will have two primary benefits:

1. ***Independently-selected characters:*** During registration, users will use the cues to choose password characters that are independent of each other, which should result in stronger passwords than users create without cues.

2. ***Cued-recall:*** The cues will trigger users' memory of the character they pressed when they first saw the cue during registration, which facilitates password recall.

Before describing CTP in detail, we will walk through a CTP password registration and login. When first registering a new CTP password, the user will be asked to rate their level of expertise on a few categories for which the system has textual cues.

```
Love is too young to know what conscience is,
Yet who knows not conscience is born of love?
   Then, gentle cheater, urge not my amiss,
   Lest guilty of my faults thy sweet self
                  prove.
     For, thou betraying me, I do betray
  My nobler part to my gross body's treason;
    My soul doth tell my body that he may
   Triumph in love; flesh stays no farther
                  reason,
  But rising at thy name doth point out thee
As his triumphant prize; proud of this pride,
    He is contented thy poor drudge to be,
  To stand in thy affairs, fall by thy side.
   No want of conscience hold it that I call
   Her 'love' for whose dear love I rise and
                  fall.
```

(a) Shakespeare's 151st Sonnet.

```
NAME
man - format and display the on-line manual
pages
manpath - determine user's search path for
man pages
SYNOPSIS
man [-acdfFhkKtwW] [--path] [-m system] [-p
string] [-C config_file] [-M pathlist] [-P
pager] [-S section_list] [section] name ...
DESCRIPTION
man formats and displays the on-line manual
pages.  If you specify section, man only
looks in that section of the manual.  name is
normally the name of the manual page, which
is typically the name of a command, function,
or file.  However, if name contains a slash
(/) then man interprets it as a file
specification, so that you can do man ./foo.5
or even man /cd/foo/bar.1.gz.
```

(b) UNIX Command `man` manual page.

```
          BLACK
     a b c d e f g h
   -----------------
 8 |R|N|B|Q|K|B|N|R| 8
 7 |P|P|P|P|P|P|P|P| 7
 6 | | | | | | | | | 6
 5 | | | | | | | | | 5
 4 | | | | | | | | | 4
 3 | | | | | | | | | 3
 2 |p|p|p|p|p|p|p|p| 2
 1 |r|n|b|q|k|b|n|r| 1
   -----------------
     a b c d e f g h
          white
```

(c) A text chess board.

```
West of House
You are standing in an open field west of a
white house, with a boarded front door.
There is a small mailbox here.
```

(d) A text adventure room description [6].

Figure 4.13: Examples of possible textual cues.

After stating that they have sufficient expertise in at least one of the categories, users are then shown a random cue from one of their categories of expertise. Many such cue categories should be available. Figure 4.13 contains examples of cues from possible categories such as works of Shakespeare, technical manual pages, text-based chess boards, and text adventure games. A cue category can encompass virtually any domain with significant quantities of content that can be represented as text. Each cue will be accompanied by a set of suggested responses. For example, if the cue is part of the text adventure game category, the cue could describe a room and provide a list of different keys (either numbers for PINs or pseudo-random characters for text passwords) the user could press for each possible action. In the case of the room described in Figure 4.13d, possible options could be:

```
0. Go north
1. Go east
2. Go south
3. Go west
4. Look at house
5. Look at mailbox
6. Look at front door
7. Open mailbox
8. Take mailbox
9. Cry for help
```

Optionally, CTP could also incorporate the same persuasive features as PTP and PCCP. *Persuasive Cued Text Passwords (PCTP)* could persuade users to select more random (and hence more secure) passwords by only allowing the user to select from a randomly-determined subset of the presented options. The available options would be visually distinguishable from the unavailable options, but both will still be clearly legible. As in PTP and PCCP, users may *shuffle* for PCTP to make a different random subset of options to be available. This extra step of shuffling will encourage users to select from the randomly-determined subset of choices, thereby generating a more secure password. This applies the *path-of-least-resistance* principle whereby the most secure behaviour is the easiest for users to perform. It would take a lot of shuffling for users to make obvious character choices or reuse one of their existing passwords for another system.

Whether using the persuasive or non-persuasive version of CTP, upon making a selection by pressing a key, the user will be shown a new cue with a new set of suggested responses. Each cue presented to the user will be pseudo-randomly chosen based on the user's response to the previous cue, as is done with CCP [44]. The user will again respond to the second cue, and be shown a third. This process will continue until a preset password length is reached. CTP will then set the user's password as the concatenation of all the user's responses, which can be stored in the usual manner for text passwords (Section 2.2).

When logging in, the user will be shown the same first cue as during registration. Thus, upon entering the correct response, the user will be shown the same following cue. However, if the user's response differs from their original response, a different cue will be shown. As previously described, this is because the shown cue is selected based on the user's previous response to the previous cue. Thus, users should realise the cue

is unfamiliar and know that their previous response was incorrect, at which point the user may retry entering their password. This *implicit feedback* property (originally part of Cued Click-Points [44]) provides the user with confirmation that they are correctly entering their password, without revealing their password to an observing adversary. Another advantage of this property is that users are informed precisely which character in their password is incorrect. Finally, users can retry entering their password immediately after making the mistake, rather than wait until entering the entire password without knowing where the mistake was made.

When beginning the registration process, the system will ask the user to rate their expertise on a random subset of all the cue categories. If the user claims to know little about all presented categories, the user will be asked to rate their expertise on a different random subset of cue categories. This process will be repeated until the user has reported some minimum level of expertise in some minimum number of cue categories. The minimum level of expertise and number of cue categories will be determined by the system administrator. We recommend at least one category with a moderate level of expertise is necessary. However, we will note that further studies will be needed to determine if a single category would provide enough security to make cue-based dictionary attacks difficult. These future studies should also determine if a self-reported moderate level of expertise in a category is sufficient to support stronger passwords that are also memorable due to the user's expertise in the subject matter.

CTP is similar to personal verification questions (PVQs, Section 2.2.4) and inkblot authentication [170,193] in that a cue (and question or an inkblot) is shown to the user and they respond by entering a response. However, PVQs and inkblot authentication are limited to questions and inkblots, while PCTP is a generalised form of these systems and could include questions or inkblots (in some form of pseudo-random text). PVQs and inkblot authentication also do not assist users in choosing more secure passwords as PCTP does using persuasion.

A useful variant of CTP is *Cued PINs (CPINs)*, whereby users' responses are restricted to digits. CPINs can support standard computers as well as mobile devices, ATMs, and other systems where number pads are the default input method. This could be ideal for websites that users log in to from both computers and mobile

devices, where typing a standard text password can be a difficult task. The portability of CTP and CPINs is their primary advantage, for the following reasons:

- ***Textual representation is universal.*** Almost all electronic systems can display text, while some systems may require additional software or hardware capabilities to display arbitrary images.

- ***Easy to display.*** Most devices (e.g. monitors, tablets, mobile phones) have different display sizes and orientations, which can make moving and displaying an image in a recognisable manner challenging. However, these devices' operating systems can automatically reformat text to fit whatever its size or orientation. Thus, textual cues are likely to display more clearly and consistently across devices and require less programming for the scheme developers than images.

- ***Less storage required.*** Text requires much less storage space and can be better compressed than images, if storage space is limited.

Restricted to digits, CPINs could only offer users a maximum of ten possible responses per cue. This offers a much smaller password space than is possible with standard CTP, which supports all keyboard characters. However, it may be unreasonable to expect users to choose from over 10 possible responses for each cue, which suggests that the full keyboard might not be utilisable in CTP. Furthermore, CPINs may be able to compensate for its small per-cue space by showing the user more cues and thus requiring more responses, resulting in longer CPINs than the typical 4-digit PIN. We would not expect users to easily choose secure or memorable long standard PINs, but it remains unclear what the security or length of CPINs users could remember, with the help of the provided cues. CPINs could also use a persuasive variant to influence users to select more secure PINs.

4.9 Conclusion

In this chapter, we have presented the design and user study of Persuasive Text Passwords, a password creation approach using Persuasive Technology by randomly

positioning randomly-chosen characters into users' passwords. Our study involved 83 participants, each using one of several PTP variations. An important result of this study is identifying the point where the limits of human memory lead users to employ coping mechanisms when dealing with randomness in passwords. Users in the stronger conditions of our study compensated for added complexity by choosing simpler passwords before applying the system's improvement. Even so, in order to remember the improved passwords, they needed to mentally rehearse the password for longer periods of time to be able to successfully use it. Our study suggests that PTP has merit in usability and security, and that 3 randomly-chosen and inserted characters is the most users can remember without exerting an unreasonable amount of mental effort. However, this result is relative to this particular study and must be verified by future studies before generalising the result for all circumstances. Admittedly, we suspect users may have difficulty recalling multiple PTP passwords for different accounts. To compensative for this potential drawback, we highlighted the *cueing* feature of PCCP which may be leveraged in text password schemes to increase memorability. We concluded with a discussion of a possible design of a *Cued Text Password (CTP)* scheme, which optionally could also support persuasion. Furthermore, CTP could use digits for a text-based *Cued PIN* scheme, which would support users in creating more memorable and secure PINs on ATMs and mobile devices.

PTP illustrates how our User-Centred Authentication Feature Framework (Section 3) can be used to identify some features in one scheme (Persuasive Cued Click-Points [46], in this case) and apply those same features to a different scheme design. We imagine a similar process can be used to develop a myriad of usable and secure authentication schemes by identifying useful features from one scheme and applying them to a novel scheme design. This cross-pollination of sorts would ultimately lead to an ecosystem of distinct authentication schemes, whereby all users, regardless of their capabilities or preferences, would be able to benefit from significant usability and security advances in authentication research and development. In the following chapter, we present another example of a novel authentication scheme, which adapts the advantages of the Cued Click-Points [44] scheme for a gaze-based interaction that does not require fine-motor control over a mouse.

Chapter 5

Cued Gaze-Points

This chapter provides two contributions to this thesis proposal. First, we will introduce and evaluate Cued Gaze-Points (CGP), a novel graphical authentication scheme that leverages the usability and security advantages of cued-recall graphical passwords, but is less susceptible to shoulder-surfing. Second, CGP is an example of how our User-Centred Authentication Feature Framework can be used to vary an existing authentication scheme's input modality from the standard keyboard and mouse. This demonstrates how a single feature can be modified to provide the original scheme's usability and security attributes to users who otherwise could not have benefitted from them. CGP and its contributions to eye tracking technology have been respectively published as a CHI 2010 short paper [81], and a CHI 2010 extended abstract [80].

5.1 Introduction

Graphical passwords (GPs) are proposed as more memorable and secure authentication methods that leverage the human ability to more easily recognise and recall images over text [142]. One disadvantage to most graphical password schemes is their susceptibility to *shoulder-surfing*: attackers may observe or record users as they enter passwords and subsequently log in with the observed credentials. Text passwords and PINs may also be vulnerable to shoulder-surfing [173, 196].

It may be possible to make some authentication systems resistant to shoulder-surfing by varying the input modality. In particular, we are interested in exploring eye-gaze as an alternative input modality for graphical passwords. Recent research on different authentication schemes [54, 61, 127] suggests that gaze-based authentication shows promise. Some other shoulder-surfing resistant graphical password schemes rely on obfuscation or challenge-response [125, 196], but these suffer from longer login times since they require more cognitive effort than traditional authentication systems.

In this chapter, we present *Cued Gaze-Points* (CGP): a cued-recall graphical password scheme using eye-gaze as an input mechanism. Cued-recall (or locimetric) password systems show a graphical cue that triggers the user's memory of their password, and therefore facilitates memory of multiple distinct passwords. Unlike similar click-based schemes, CGP is shoulder-surfing resistant since there is no on-screen indicator revealing users' gaze-point locations. Our 45-participant user study is the first evaluation of gaze-based password entry with user-selected points on images, rather than pre-defined regions.

CGP is a variant of Cued Click-Points [44] that an eye tracker as input instead of a mouse. We identify the following features in CGP with our User-Centred Authentication Feature Framework (Chapter 3) By design, graphical password schemes like CGP leverage visual memory (*Vism*) for locations or objects on images. Users may choose their gaze-points based on semantic details (*Sem*) or episodic recollections (*Eps*). With some practice, we believe users would not need to consciously recall the locations of their gaze-points, but would unconsciously remember them from procedural memory (*Proc*) and the presentation of the cue (*Cue*). An attacker observing the login process would have difficulty determining where users' input (*Oin*) gaze-point is precisely located on the image. Furthermore, the feedback given to the user is also obfuscated (*Ofb*), since there is no feedback highlighting the correct location of the gaze-point. Obviously, users' eye gaze (*Eye*) is the intended input modality, and a visual output modality (Viso), such as a monitor, is required. CGP does not support any persuasive features.

Note that the obscured feedback (*Ofb*) feature was not present in CCP, where an observer could see the mouse cursor move towards the click-point. By changing the input modality from the mouse (*Mse*) to eye gaze (*Eye*), CGP automatically gains an additional feature without additional design effort on our part. This exemplifies how our framework can provide surprising but positive results when used to isolate and modify authentication features.

This chapter is organised as follows. We will first cover the relevant background in Section 5.2. We will next describe the methodology (Section 5.3.1) and results (Section 5.3.2) of our user study on the first version of CGP. The subsequent design

improvements in the second version of CGP will be discussed in Section 5.4, followed by our hypotheses (Section 5.5) and results (Section 5.5.1) of our second user study. Section 5.5.2 will discuss the results, and we will offer concluding remarks in Section 5.6.

5.2 Background

Gaze-point selection can use a *gaze-trigger* method, where users gaze at the desired location while pressing a button to explicitly indicate their gaze-point selection. Another method is *gaze-dwell*, where users' gaze dwelling on an object implies its selection. Kalman filters have been explored as a gaze-dwell method of continued-stream user input [128]. In our system, users' gaze must first be used to search for the correct location to enter the login point, which makes the more explicit gaze-trigger method preferable.

Jacob and Karn [111] suggest that, in spite of many challenges in eye tracking technology (calibration, accuracy, etc.), eye trackers have provided many human-computer interaction insights. Vertegaal [207] found that eye tracking has properties relating to Fitt's Law, with some speed and accuracy advantages over mouse-based interaction. Eye tracking applications could be widely-deployed since most modern computers have a webcam, and eye tracking research using webcams conducted over the Internet [68] is already possible.

Some recent authentication scheme proposals use eye-gaze input, which should become affordable in the near future. Kumar et al. [127] first implemented a gaze-based authentication system. Their EyePassword scheme displays an on-screen keyboard whereupon users gaze at the letters of their password. However, EyePassword remains vulnerable to guessing attacks due to the predictability of text passwords. De Luca et al. [54] have proposed eye-gesture methods for shoulder-surfing resistant authentication. Dunphy et al. [61] tested gaze control with PassFaces (Section 2.3).

5.3 Cued Gaze-Points Version 1 (CGP-1)

Cued Gaze-Points (CGP) is an eye-gaze version of Cued Click-Points (CCP) [44], where users select points on a sequence of images with their eye-gaze instead of the mouse cursor. As in CCP, CGP uses Centered Discretization [45] for gaze-point identification. In our studies, we used a 17" Tobii 1750 eye-tracker. Our first version of CGP (CGP-1) used the gaze-trigger method of gaze-point selection.

5.3.1 Methodology

To evaluate CGP-1's usability, we conducted a 16-participant user study following the published methodology for the CCP [44] study, including the same image set. Participants completed a 1-hour in-lab session of 6 trials, each including the following steps:

- ***Create.*** Participants first created a password by gazing at five points of their choice on five different images. To select a gaze-point, users would stare at the desired point and press either the left-mouse button or space bar

- ***Confirm.*** Participants confirmed their password by selecting the same gaze-points as during creation. If users could not successfully confirm their password, they could retry as many times as they wished or skip the trial.

- ***Questions.*** Users were then asked two 10-point Likert scale questions on how easy they felt it was to create this password, and how difficult they thought it would be to remember in a week.

- ***Distraction.*** Users took at least 30 seconds to perform a mental rotation test [161] to clear their visual working memory and simulate a longer passage of time.

- ***Login.*** Users logged in with their password to verify it could be recalled. If users could not successfully log in, they could retry as many times as they wished or skip the trial.

The materials used in this study can be found in Appendix B. Eye-gaze is less precise than mouse input, so we adjusted the system configuration from CCP's 1024×768 resolution to 800×600, resulting in a $\frac{1024}{800} = 1.28$ linear increase in our study's image size. People have full vision acuity within $\sim 1°$ of their gaze's centre [60]. Thus, a 1° radial target on a 17" monitor with a 800×600 resolution that is $64\,cm$(25" [7]) away from the user forms a circular target with a *51 pixel* diameter. Kumar et al.'s [127] on-screen keys were a similar size. Although the full vision acuity area forms a circular target on the screen, we used square tolerance regions because CGP needs a grid system to store passwords through discretisation [45].

For analysis of our study's results, we use a variety of statistics to determine if two or more distributions are significantly different. For normal distributions, we use t-tests and one-way ANOVAs. When the data is not normally distributed, we use Mann-Whitney U and Kruskal-Wallis (KW) tests instead. For categorical data, we primarily use chi-squared or Fisher's Exact tests depending on category counts. The Bonferroni correction is applied as appropriate when doing multiple comparisons. In all tests, we accept $p < .05$ as statistically significant.

5.3.2 CGP-1 Results

CGP-1's usability was much poorer than its click-based predecessor, CCP. CGP-1 participants had lower error-free confirm and login success rates (36% and 50% versus 83% and 96%), more mean confirm and login errors per trial (3.03 and 2.95 versus 0.39 and 0.05), and longer mean create and login times (42.23 s and 38.87 s versus 24.7 s and 7.4 s) than CCP participants. We found two main problems when examining users' gaze patterns and behaviour.

First, users' gaze sometimes jittered away from their desired target when pressing the record button, causing the system to store the incorrect point. Second, users naturally moved their head and body throughout the study. This caused the system to increasingly misinterpret users' gazes and record all gaze-points as though they had *drifted* from the intended target.

This problem of calibration and drift are somewhat different in CGP than for other usages. Eye tracker use in CGP is brief and subsequent logins may be widely separated

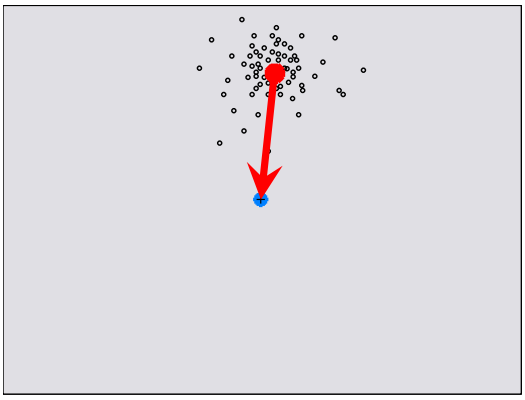


Figure 5.1: Illustration of how 1-point calibration calculates the offset before password creation.

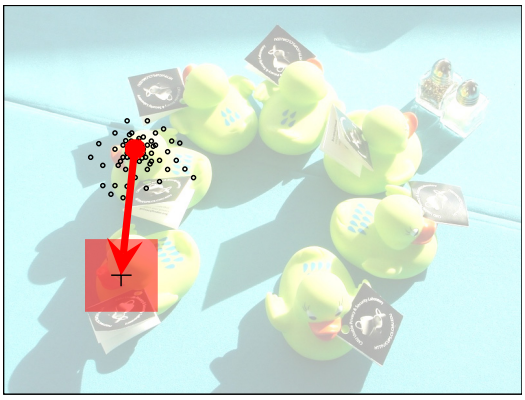


Figure 5.2: Illustration of how 1-point calibration corrects drift during password creation.

in time. Any calibration in CGP must be very quick, otherwise it takes longer than the password entry. Clearly, some sort of calibration is needed since users are unlikely to sit in the exact same position for each login and they need maximum precision to accurately select gaze-points. Furthermore, with greater precision, smaller tolerance squares can be used without compromising usability. This in turn makes for a larger number of possible distinct gaze locations on each image [45], which increases the theoretical password space and system security (against password guessing attacks).

5.4 Gaze Accuracy Enhancements

We developed the following enhancements we hoped would improve gaze accuracy.

5.4.1 Nearest-Neighbour Gaze Aggregation

Instead of recording users' current gaze-point when a button is pressed, the first enhancement has users record multiple gaze-points as they hold the space bar. When it is released, the system uses a nearest-neighbour algorithm to calculate the user's gaze-point for this image as the point with the most neighbours within its tolerance. If there are several such points, their average is selected. We hoped that this method of gaze-point selection would more accurately reflect the user's intended gaze-point. However, we expected this might slow down gaze-point selection, since users hold down the space bar for a few seconds with each point.

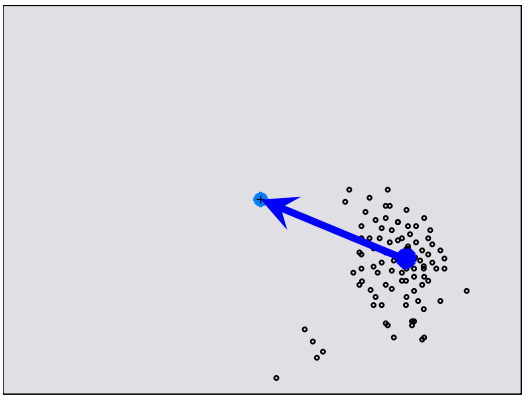


Figure 5.3: Illustration of how 1-point calibration calculates the offset before login.

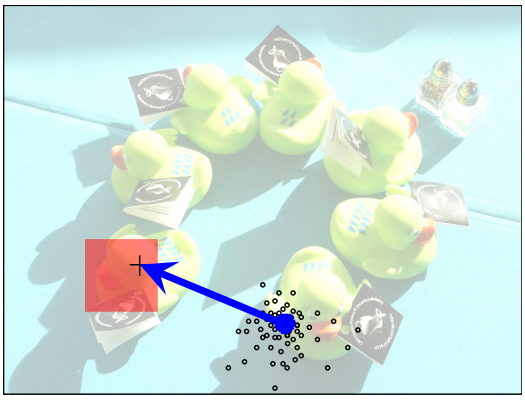


Figure 5.4: Illustration of how 1-point calibration corrects drift during login.

5.4.2 1-Point Calibration

The second enhancement is a *1-point calibration* that users perform before they are about to enter a password (before creating, confirming, or logging in with their password). A blank image containing only a blue dot in the centre is displayed. Users stare at the dot and hold the space bar as though selecting a gaze-point. Upon releasing the space bar, the system calculates the offset distance between users' gaze-point (calculated using the nearest-neighbour gaze aggregation algorithm) and the actual location of the blue dot. This offset is then applied to all gaze-points recorded during that phase. We hoped that these quicker, more frequent calibrations would eliminate the drifting problem.

The 1-point calibration is illustrated in Figures 5.1 to 5.4. The blue circle in Figures 5.1 and 5.3 denotes the centre of the calibration image. The red arrow in Figure 5.1 and the blue arrow in Figure 5.3 illustrate the offsets calculated during the corresponding create and login calibrations. Figures 5.2 and 5.4 show these offsets being respectively applied to a create and login gaze-point. The transparent red square represents the *tolerance region* around the password gaze-point. Without 1-point calibrations, this login would have failed, since the login point would have been outside the acceptable tolerance of the create point.

Users' foveal vision roughly covers a circular area when gazing straight at the screen. However, when gazing at the edges, the coverage is more elliptical. Eye

trackers typically perform a 5-point calibration (centre and four corners) to ensure full-screen coverage. However, in CGP, users' gaze-points are only recorded within the displayed image (roughly centred on the screen), so a quicker calibration with only one point may provide sufficient coverage. In our setup (see CGP-1 section above), the image dimensions are $19 \times 15\,cm$, so distance between the image's centre and corner is $12.1\,cm$. This leads to an angle of $\sim 10.71°$ between the centre of gaze from the image's centre to the image's corner. This projects an elliptical foveal area onto the monitor, centred at the corner of the image, with a major (longest) diameter of $\sim 2.31\,cm$. Comparatively, a circular foveal target at the centre of the image has a diameter of $\sim 2.23\,cm$, so there is at most a $\frac{2.31-2.23}{2} = 0.04\,cm < 1\,pixel$ radial margin of error when recording image gaze-points. Therefore, we believe that our 1-point calibration should not result in errors at the image edges, given the current system parameters (e.g. screen size and image size and position).

5.5 Cued Gaze-Points Version 2 (CGP-2)

We implemented these two enhancements in a second version of CGP (CGP-2). Following the same methodology as for CGP-1, we repeated the study to test CGP-2 with 25 new participants. We made the following hypotheses about CGP-2:

H1. CGP-2 will achieve higher success and lower error rates during password re-entry than CGP-1.

H2. CGP-2 will have longer login times than CGP-1.

H3. CGP-2's 1-point calibration will retain gaze accuracy at the edge of the images and not result in more errors at the edges of the images.

5.5.1 CGP-2 Results

Table 5.1 lists performance comparisons between CGP-1 and CGP-2 with significance test results. Table 5.1 shows that CGP-2 users successfully logged in without error (on their first attempt) more often and committed fewer mean login errors than CGP-1 users. The table also shows no significant differences in the time CGP-1 and CGP-2 users took to create or log in. These results support hypothesis H1, but H2 cannot

System	CGP-1	CGP-2	CGP-1 vs CGP-2 Sig. Tests
# of Participants	16	25	-
# of Trials	127	169	-
Successful Logins on 1st try	50%	73%	$t(39) = -2.43, p < .05$
Mean Login Errors (per trial)	2.95	0.51	$U = 300.5, p < .005$
Mean (SD) Create Time (s)	42.2 (22.4)	44.2 (22.0)	n.s.
Mean (SD) Login Time (s)	47.9 (64.2)	36.7 (35.9)	n.s.

Table 5.1: Performance comparisons between CGP-1 and CGP-2.

be accepted. This means our enhancements resulted in higher success rates and fewer errors, without lengthening login times.

Figure 5.5 compares the x-coordinate of users' creation gaze-points to the horizontal distance (Delta X) between the respective creation and login points. Blue circles represent login points within tolerance, while red ×s are outside. We see no more erroneous gaze-points at the figure's top and bottom than near the centre. The analogous y-coordinate graph is similar. This suggests our 1-point calibration did not introduce errors at the images' edges, which supports our earlier calculations and hypothesis H3.

Apart from the 25 participants who tested CGP-2 with a tolerance square size of 51×51 (T-51), 20 additional participants tested a smaller tolerance size of 31×31 (T-31) instead of 51×51.

Table 5.2 compares the number of participants, trials, and general results from our CGP-2 study conditions, T-51 and T-31. In the right-hand column, we show the published results from the CCP study for comparison.

Successes. The success rates and chi-square significance tests in Table 5.2 show that larger tolerance squares (T-51) resulted in significantly easier logins than smaller tolerance squares (T-31). In comparison with CCP, T-51 participants logged in 73% without error, while CCP users did so 96% of all logins. Note T-51's 3-try login rate (93%) and CCP's 1-try login rate (96%) are comparable, as are the confirm success rates of T-51 3-try (82%) and CCP 1-try (83%). Thus, CGP users can successfully re-enter passwords, but may attempt more than once.

Errors. CGP participants using larger tolerance squares (T-51) were significantly less prone to login errors. Despite larger tolerance squares leading to higher confirm

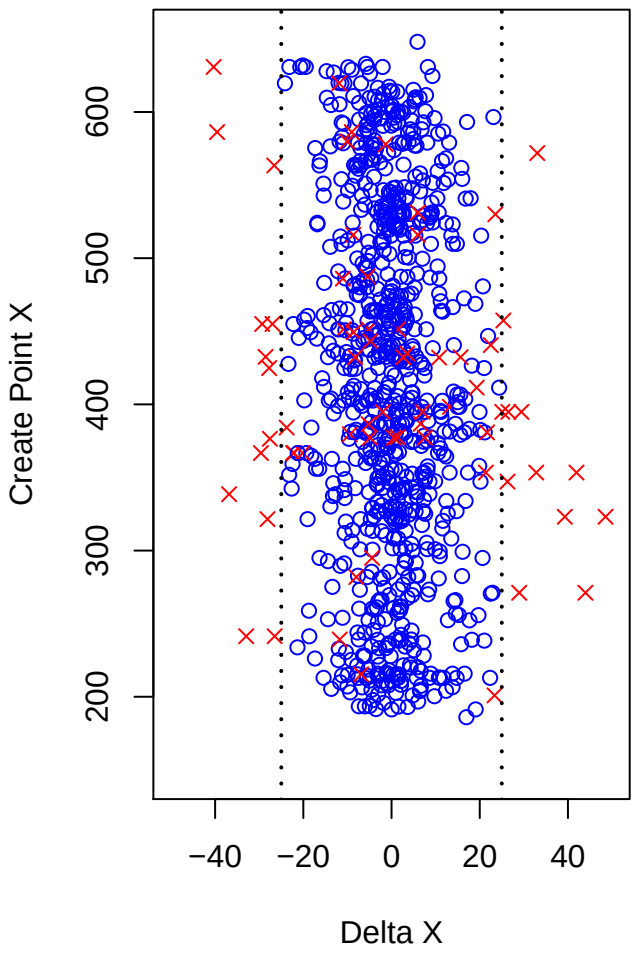


Figure 5.5: Scatterplot of CGP-2 login gaze-points. The x-axis represents the horizontal distance between the creation point and the login point, and the y-axis represents the x-coordinate of the creation point. Blue circles are correct gaze-points and red Xs are incorrect. The dotted vertical lines show the tolerance region.

System	CGP-2 T-51	CGP-2 T-31	T-51 vs T-31 Sig Tests	CCP T-19
# of Participants	25	20	n/a	24
# of Trials	169	141	n/a	257
Confirm on 1st try	67%	50%	n.s.	83%
Confirm in ≤ 3 tries	82%	68%	n.s	n/a
Login on 1st try	73%	54%	$t(43) = 2.67, p < .05$	96%
Login in ≤ 3 tries	93%	79%	$t(43) = 2.35, p < .05$	n/a
Avg confirm errors per pwd	1.21	1.68	$U = 162, p < .05$	0.39
Avg login errors per pwd	0.51	1.11	$U = 102.5, p < .005$	0.05
Avg (SD) create time (s)	44.2 (22.0)	50.1 (23.2)	n.s.	24.7 (16.4)
Avg (SD) confirm time (s)	47.1 (78.5)	64.3 (85.3)	$U = 147, p < .05$	10.9 (13.1)
Avg (SD) login time (s)	36.7 (35.9)	53.5 (45.9)	$U = 107, p < .005$	7.4 (5.5)

Table 5.2: Results from our CGP-2 Tolerance-51 and -31 (T-51 & T-31) study conditions and Chiasson et al.'s CCP study [44].

success rates, no difference in confirm errors was observed. CGP users generally committed more errors per trial than CCP users. This is not surprising, since users are probably more practiced at pointing with a mouse than with their gaze.

Times. CGP passwords were re-entered for confirm and login more quickly with larger tolerance squares (T-51). Participants were slower on average to create, confirm, and log in with T-51 than with CCP. This is not surprising because CGP times include typing their username, calibrating, and recording a few seconds of gaze for each point, while mouse-clicking is rapid and CCP times begin at the first click-point [44]. CGP users also committed far more errors than CCP users, and time elapsed during errors and re-tries is included.

Accuracy. Figure 5.6 plots the frequencies of the Euclidian distances between the creation and login points for passwords created with T-51 and T-31 respectively, scaled by natural logarithm for greater detail. We use a line graph instead of overlaid histograms to make comparisons between the conditions easier. Figure 5.6 shows that T-51 login gaze-points were farther from the corresponding creation gaze-point than were those of T-31 ($t(41) = 2.63, p < .05$). This shows that CGP participants gazed more precisely when using a smaller tolerance square. We suspect this occurred because T-31 users required greater precision to re-enter their passwords. Although eye-tracking is limited by eye physiology, somewhat better precision may be achieved with additional care. We also examined the x- and y-coordinates of incorrect gaze-points to see if CGP participants committed more errors at the edges of the image, due to our 1-point calibration. We found no such evidence, since the errors were evenly distributed across the image.

User perception. Figures 5.7a and 5.7b show notched box plots of post-test responses to two 10-point Likert scale questions. Higher scores favour CGP. Figure 5.7a shows that T-51 users felt they could easily create CGP passwords, while T-31 users were neutral. Figure 5.7b demonstrates that most CGP users felt they could quickly enter passwords with practice. This suggests that participants felt that any difficulties were because they lacked familiarity with eye-tracking, which reflects our belief as well.

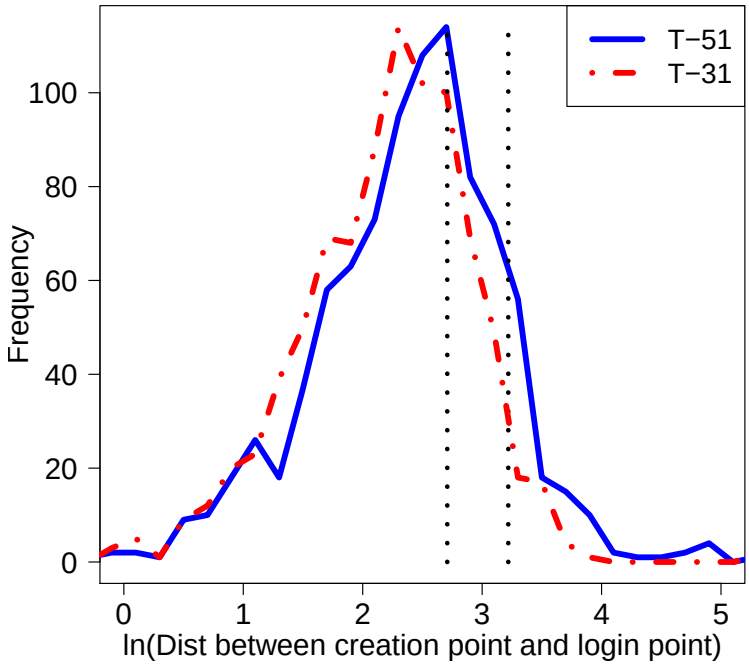
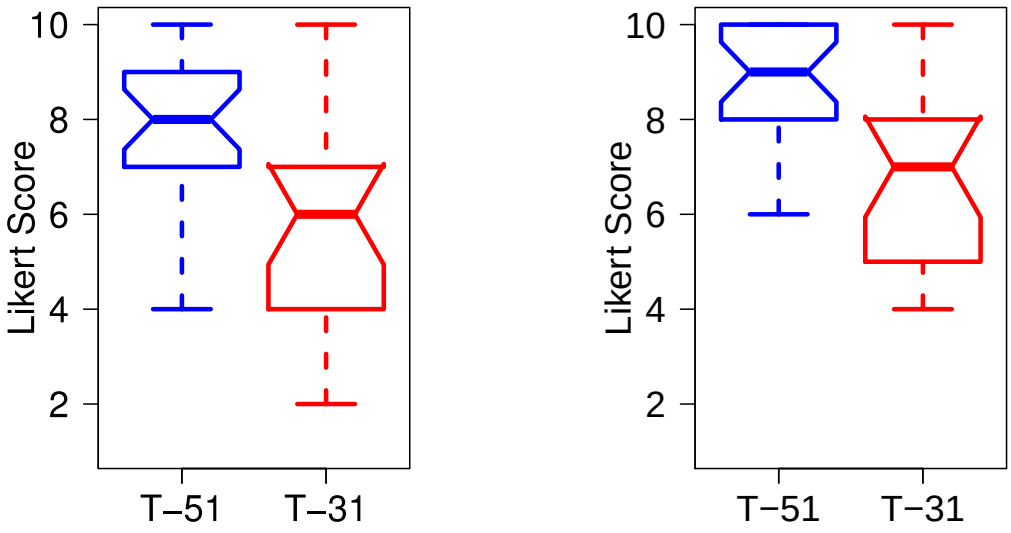


Figure 5.6: Frequencies of Euclidian distances between the creation and login points for passwords created, scaled by natural logarithm (ln). The vertical lines denote the conditions' tolerance square boundaries.

5.5.2 Discussion

Results of our CGP-2 study show a clear trade-off between usability and security. We found the smaller tolerance size too difficult to use with eye-tracking technology. The larger tolerance size proved considerably more usable. However, this configuration's smaller theoretical password space (Section 2.2) makes it more vulnerable to password guessing attacks. This would be an acceptable trade-off in certain environments. For example, T-51 is much more secure than ATM PINs, because of a larger password space. TPSs grow exponentially, and are typically compared in log_2. An image size of 451×331 gives a grid of approximately 63 51×51 squares. With 5 gaze-points per password, the password space of T-51 is $log_2(63^5) \approx 29.9\,bits$, while 4-digit PINs only offer $log_2(10^4) \approx 13.3\,bits$. CGP can also offer an even larger TPS by using larger images and/or adding gaze-points. For example, using an 800×600 image and 7 points, T-51 would have a password space equivalent to an 8-character password using a full 95-character US keyboard ($log_2((16 \times 12)^7) \approx 53.1\,bits$ vs $log_2(95^8) \approx 52.6\,bits$). TPS is only an estimate of security against guessing attacks, since not all points on



(a) Ease of creation

(b) Speed of entry

Figure 5.7: Box plots of Likert scale responses. High scores favour CGP.

an image or text character combinations are equally likely to be chosen by users. This issue is addressed by Persuasive Cued Click-Points (Section 2.3).

We next compare our CGP-2 T-51 (hereafter abbreviated to simply CGP) results (Table 5.2) to other gaze-based authentication systems. Precise comparisons are not possible due to differences between study methodologies, tasks, and assessment. For example, CGP success rates and times include username entry, calibration time, failed login attempts, and password recall time, in order to present more ecologically valid results. Conversely, Kumar et al. [127] and De Luca et al. [54] only reported the mean password entry time from the first to last gaze, and their participants received more training than CGP participants. We also count a login failure when users either enter an incorrect password or choose to re-enter their password if they see an unfamiliar image (by having previously gazed at the wrong point). This latter re-entry is analogous to erasing and re-typing a text password, which Kumar and De Luca do not count as failed logins. CGP users chose their own passwords, while Kumar and De Luca did not test memorability by assigning passwords to users and showed them their passwords before entry. CGPs success rates are lower as a result of using a stricter definition of "success", which is more representative of real-world usage.

Kumar et al. [127] tested their EyePassword system with two conditions: Dwell (when the users stare at their desired target) and Trigger (where users press the space bar to record their gaze-point). EyePassword participants took on average of 10.7 s and 9.2 s to enter Trigger and Dwell passwords respectively. The 1-try success rates for Trigger and Dwell were 85% and 97% respectively. EyePassword performance appears better than CGP, but EyePassword users saw their password immediately before entering it, while CGP users relied solely on memory. Each EyePassword on-screen keyboard key had a focus point in the centre to help users focus their gaze. Obviously, CGP cannot show users their gaze-point, as this would reveal the password to an attacker. EyePassword and text passwords also share the same password space weaknesses: weak user-chosen passwords.

De Luca et al. [54] reported that EyePassShapes users took an average 12.5 seconds to log in, and achieved a 1-try login success rate of 86%. This suggests that EyePassShapes is quicker and easier to use than CGP. However, we believe EyePassShapes may be more vulnerable to shoulder-surfing, since attackers could simply watch users' eye movements to capture passwords. Furthermore, EyePassShapes passwords were 7 gestures long, each of which could be one of 8 possible directions, resulting in a password space of $log_2(8^7) = 21 bits$, which is smaller than CGP ($29.9 bits$, see above).

Login times were not reported in Dunphy et al.'s [61] gaze-based PassFaces study. Their 1-try and 3-tries login success rates were 40% and 65% respectively, which are lower than either CGP condition. CGP also has a larger password space ($29.9\,bits$, see above) than PassFaces ($log_2(9^5) \approx 15.8\,bits$).

5.6 Conclusion

Graphical passwords offer a number of security and usability advantages over text passwords. Although click-based graphical passwords (such as Cued Click-Points) are potentially vulnerable to shoulder-surfing, Cued Gaze-Points offers a gaze-based alternative that is resistant to such attacks. Even with a recording of users' eyes during login, we believe it would be very difficult to align and synchronise this data with the screen with sufficient accuracy.

An initial user study on CGP revealed that its unique use of eye tracking required special techniques to enhance users' gaze accuracy. We developed two novel gaze-accuracy enhancements: a quick 1-point calibration and a nearest-neighbour gaze-point aggregation algorithm. We implemented and user tested these enhancements in a second version of CGP. The two enhancements significantly improved users' gaze accuracy and the system's usability. It is encouraging to see that 93% of login attempts in the T-51 condition were eventually successful, indicating that users are capable of using the system with additional practice. Participants also indicated confidence in their ability to improve with practice.

CGP is the first implementation of a shoulder-surfing resistant cued-recall graphical password system using eye-gaze. This approach has a number of advantages over similar gaze-based schemes and with sufficiently large tolerance squares, the system's usability is potentially acceptable. CGP's password space is larger than similar gaze schemes, and hence more secure against password guessing attacks. Moreover, CGP's cued-recall nature can help users remember multiple distinct passwords, as distinct images on different systems will help users remember their different gaze-points.

Most monitors and laptops today have built-in cameras, and we expect eye tracking technology to become more affordable in the near future, so it could have a place in everyday user interaction. Given the choice and availability of eye-tracking, users may prefer a shoulder-surfing resistant scheme for systems they wish to access in public areas.

Eye tracking could be used by a variety of applications with properties similar to CGP: a very short time period of use with potentially long time periods between uses. We hope such applications would benefit from the two enhancements described in this chapter.

A significant benefit to supporting multiple authentication schemes is accessibility, whereby all users can benefit from novel usable and secure authentication research. This chapter demonstrates how our User-Centred Authentication Feature Framework (Chapter 3) can be used to isolate and modify a single feature of an authentication mechanism. This process can generate a novel scheme that retains most of the original scheme's benefits, but makes them accessible to people who may have been

unable to use the original scheme. In this chapter's example, the input modality (Section 3.4) of Cued Click-Points (CCP) [44] was modified from mouse to eye gaze. The resulting scheme, Cued Gaze-Points (CGP), retains most of the benefits of CCP, and makes them accessible to users either lacking fine-motor control or are concerned with shoulder-surfing attacks.

The ultimate motivation of CGP and Persuasive Text Passwords (Chapter 4) is to exemplify how our User-Centred Authentication Feature Framework can propagate the advances in authentication research into multiple schemes that provide similar benefits in a variety of ways. This enables users with different preferences and abilities to access said usability and security benefits. However, in order to reap the benefits provided by a myriad of schemes, we must provide an appropriate form of guidance to teach users how to use said schemes.

Chapter 6

Authentication Scheme Learnability

6.1 Introduction

Authentication is a necessary process in order to protect users' assets. Given the difficulties users have with creating memorable and secure plaintext passwords, introducing a more secure novel authentication scheme is a likely possibility. In fact, an ecosystem of novel authentication schemes could present all users with the security and usability advances in authentication research. However, these schemes, including Persuasive Text Passwords (Section 4) and Cued Gaze-Points (Section 5), are typically tested in experiments where users are given an in-person walkthrough of the scheme by an expert [15,19,35,175] or asked to read a page of instructions [11,129,224]. However, should any of these schemes be widely deployed, traditional in-person training would be time consuming, expensive, and may be ineffective [32]. Instead, newly-deployed schemes would need to be accompanied with some form of self-training material.

This chapter contributes to usable security research by evaluating different tutorial methods, in the hopes of easing users' transition from plaintext passwords to a novel method of authentication. We present the design of four types of training material; a plain page of instructions with a single image, a sectioned text tutorial with several images, an interactive demo, and a narrated video tutorial. We performed a between-subjects user study to determine which tutorial best helped users to learn a novel authentication scheme. We hypothesised that interactive demo users would spend more time on their tutorial than all other users, but that demo users would spend less time registering and create more secure and memorable passwords than all other users, due to superior learning from the demo tutorial's richer interaction.

Over one week, participants recruited both locally and from Amazon Mechanical Turk [3] learnt to use the graphical authentication scheme Persuasive Cued Click-Points (PCCP) [40] to register with and log in to three different websites to perform a

typical task. We chose PCCP over other password schemes because it offers reasonable security and usability [46], but we believed it may be complicated and challenging for users to learn with only a tutorial to guide them. This work has been published as a full paper at E-Learn 2012 [83].

We found only two clear differences between conditions. As expected, text and hypertext users spent less time on their respective tutorials than video and demo users. Curiously, hypertext and video users more easily recalled their passwords than text and demo users, despite demo being more engaging. Overall, users seemed to benefit most from our hypertext tutorial.

Section 6.2 provides background on user training methods. Section 6.3 describes the tutorial contents, while Section 6.4 describes our study's methodology. We present our hypotheses in Section 6.5, and the results in Section 6.6. Our hypotheses are tested and results interpreted in Section 6.7. Finally, we present our concluding remarks in Section 6.8.

6.2 Background

Any novel authentication scheme or enhancement should be accompanied by some kind of primer on how users can create and login with memorable and secure passwords. Barton and Barton [13] may have been the first to advocate for the availability of material on "user-friendly password methods", both before and while using the password system. They advise such material contain descriptions and illustrations for creating memorable passwords, as well as more technical information, such as minimum password length and other restrictions.

Carroll et al. [32] advocate a minimalist approach (known as the Minimal Manual) to training users for new computing tasks. They suggest structuring the training material to facilitate users' desire to begin the task as soon as possible, present only material that is essential to performing the task, and support error recognition and recovery. We applied these principles in the design of our tutorials. For example, the first half of the tutorials contained only material essential to use PCCP, while supplementary material was placed in the second half. Halfway through the tutorial, users were explicitly told they could skip the supplementary material if they desired.

Grossman et al. [97] provide an excellent summary of the learnability literature, and present 25 learnability metrics that have been used. In related work [96], they found that online videos helped users complete seven times more unfamiliar tasks in the same amount of time given to participants learning from traditional text-based information.

Some of the design of our tutorials used Persuasive Technology [77] (PT). PT is a framework for building technological solutions to support users in learning new behaviours. The framework states numerous principles that are based upon established psychological research on human behaviour and motivation. PT has been successfully used in numerous domains to assist people in learning new or improving existing behaviours. We think using PT is especially appropriate for teaching users PCCP, since PCCP itself is an outgrowth of PT applied to usable security [82]. Weirich and Sasse [213] also recommend using persuasion in tutorials and training material to guide users towards more secure password selection and behaviours.

To our knowledge, there is no published research on teaching users new authentication methods. However, there are commercial authentication products that are approaching training with demonstrations. An online demo of Passfaces [156] emphasises the memorisation of users' assigned faces, and walks users through the process of confirming and logging in. Although they were developed independently, the PassFaces demonstration is similar in form to the demo tutorial presented in this chapter. Unfortunately, any evaluation of the PassFaces demonstration's effectiveness has not been published.

6.3 Tutorials

We drew upon several sources when designing the tutorials. The Minimal Manual [32] principles ensured the tutorials contained only the minimum material users would need to create PCCP passwords that were both memorable and secure. The Persuasive Technology (PT) [77] tools advised us how to deliver suggestions at the most opportune moments and to tunnel users through the tutorial. Finally, the tutorials' sections were "dynalinked" [171] to make it easier for users to move through them freely and help them form correct mental models of the password system's usage.

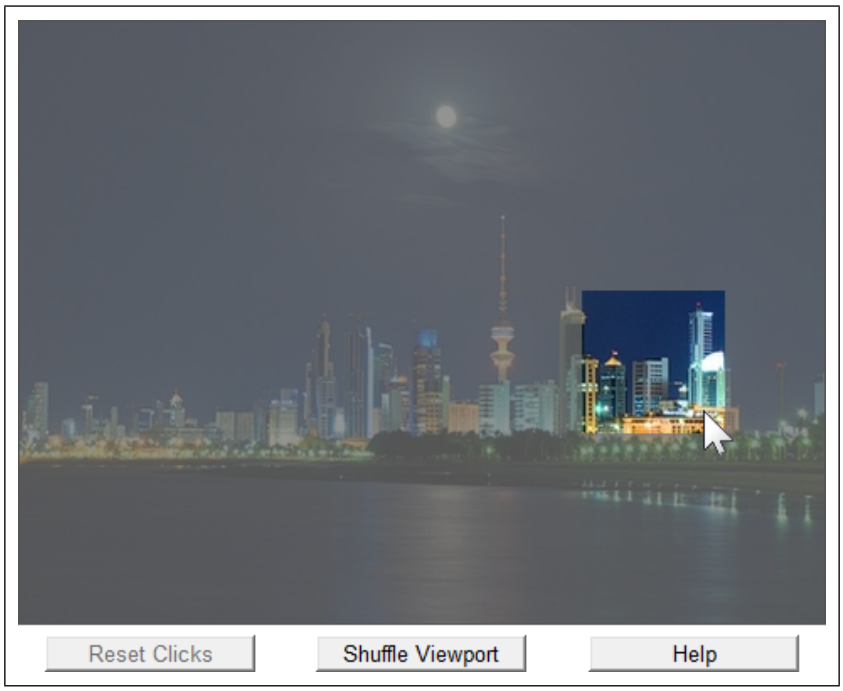


Figure 6.1: PCCP interface for password creation.

The tutorial material was divided into the following seven sections:

1. *Basic Usage*: The tutorial first introduced PCCP, the persuasive viewport, and the shuffle button (see Section 2.3 and Figure 6.1).

2. *Choosing a Good Password*: Users were encouraged to choose click-points that were "easy to click on precisely, but don't stand out very much", and were given an example.

3. *Accuracy for Login*: The tolerance region was described and illustrated, and a related example was provided.

4. *Number of Click-Points*: The tutorial told users their password would consist of 4 click-points and that they now knew all they needed to use PCCP. Users were explicitly told that they could either start using PCCP right away, or continue the tutorial.

5. *Why Am I Seeing the Wrong Image?*: This section discussed PCCP's implicit feedback property, whereby each image shown to the user is chosen based on

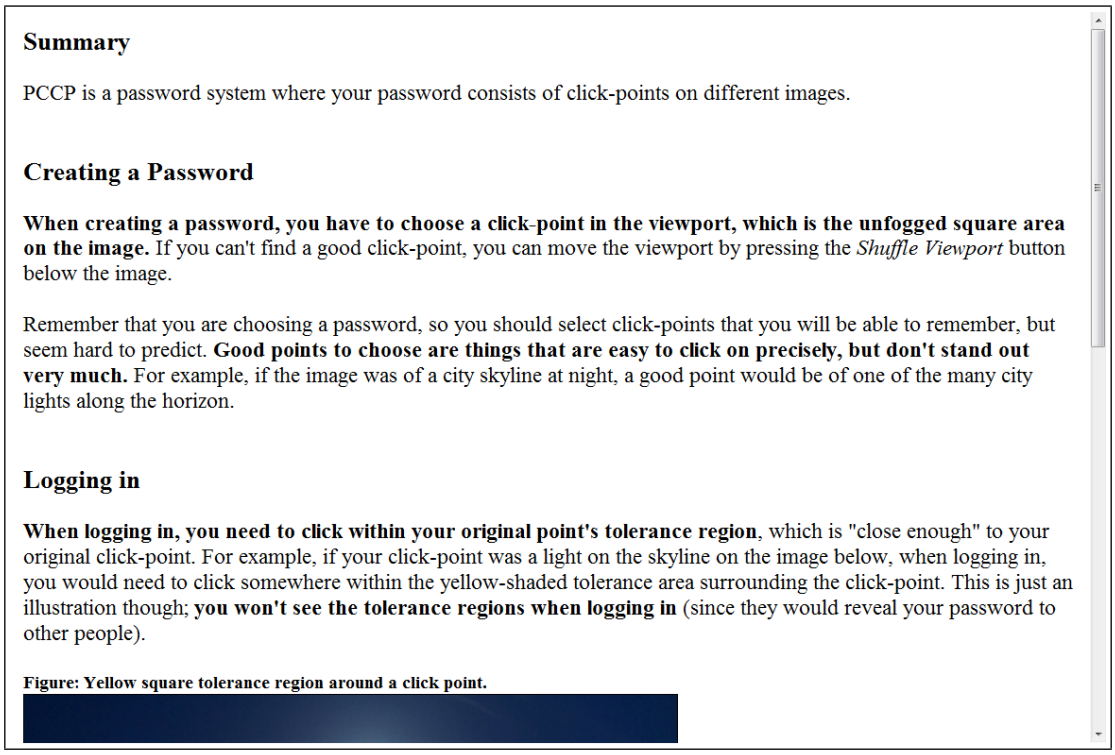


Figure 6.2: The *text* tutorial.

their previous click-point. Thus, when logging in, if the user sees an image they do not recognise, they know immediately that they made a mistake and should re-enter their password.

6. *Avoiding Bad Passwords*: The tutorial explained that objects that are either very obvious or difficult to click on precisely are poor click-points selections, because they are either insecure or hard to remember. An example of each was given, and users were advised to shuffle the viewport when only poor click-points were in the viewport's current position.

7. *Why the Viewport?*: The tutorial described how the randomly-positioned persuasive viewport helped users select more random (i.e. more secure) click-points.

Based on this material, we designed four different tutorial types:

Text: This was the simplest tutorial; one page of instructions with an image (Figure 6.2). This low-tech tutorial omitted sections 5, 6, and 7 (see above), which were not absolutely necessary in order to get started with PCCP. These omissions

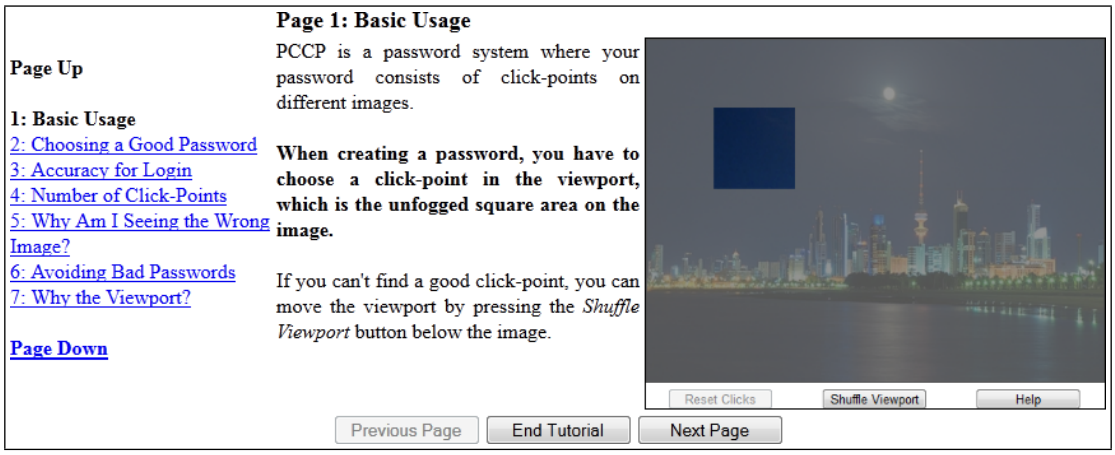


Figure 6.3: The *hypertext* tutorial's opening page.

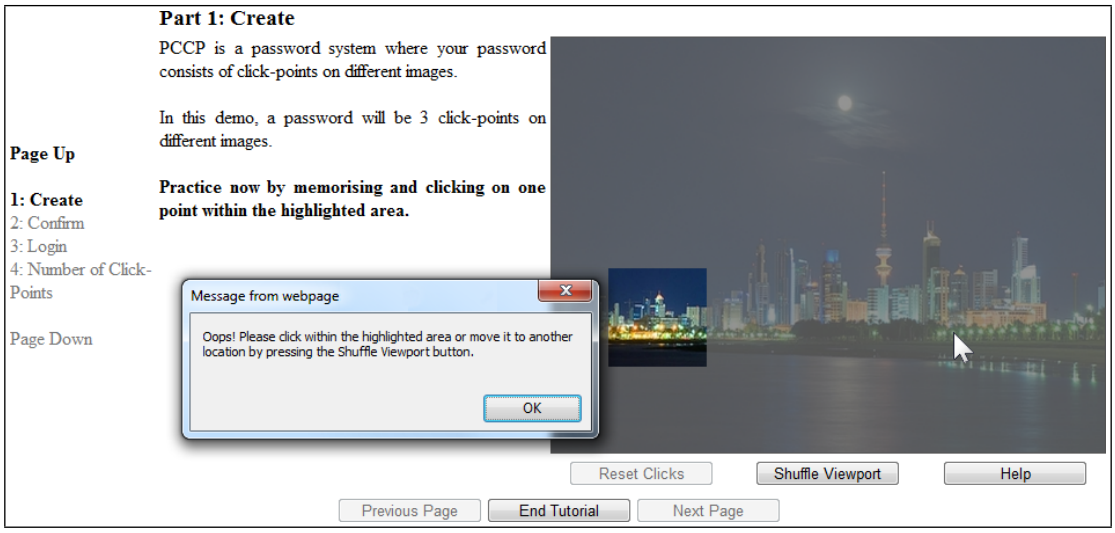


Figure 6.4: The *demo* tutorial providing immediate feedback.

followed the Minimal Manual approach, as well as an attempt to emulate the tutorial design of a developer without expertise in usability.

Hypertext: We split the tutorial material into seven hyperlinked sections of instructions and images (Figure 6.3). We made the material easier to parse with links and buttons to move between page-like sections, each of which included an illustration to make information easier to access understand. This condition is a richer use of web technology than the text tutorial.

Demo: We constructed an interactive demo with which users could practice creating, confirming, and logging in with a three-click password, supplemented with

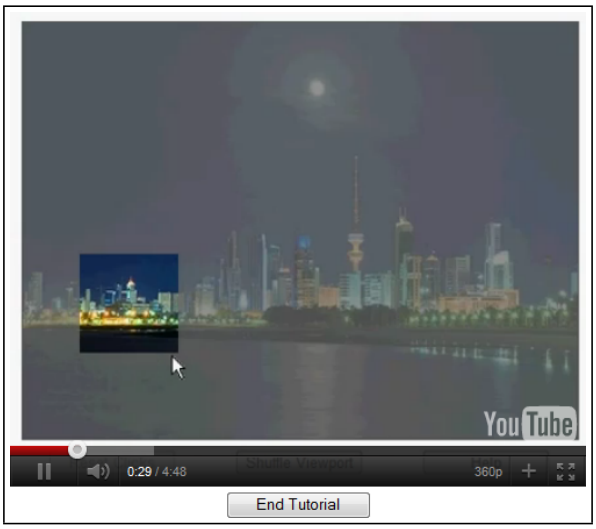


Figure 6.5: Screenshot of the *video* tutorial playing.

textual guidance (Figure 6.4). Links and buttons similar to the *Hypertext* condition aided navigation through the tutorial. Users received immediate feedback whenever they interacted with the demo. We leveraged Persuasive Technology principles [77] in designing this tutorial, since PT has been successfully used to educate users in other domains. PT principles used in the demo include:

- *Tunnelling*: The tutorial stepped users through the process of creating, confirming, and logging in using a PCCP.
- *Suggestion*: The tutorial delivered key advice at opportune moments, such as suggesting secure but memorable click-points when creating a demo password.
- *Self-monitoring*: Immediate feedback on the user's progress was provided after every click-point selection.
- *Conditioning*: The user was informed when a mistake occurred, and the tutorial advised how to avoid future mistakes.

Video: We recorded and uploaded a five-minute YouTube [95] video of screen-captured interactions with PCCP, with narrated instructions and advice (Figure 6.5). Users could pause, play, and seek as with any typical online streaming video. The video began with an index of the sections with timestamps, providing an overview of

the content and allowing quick access to desired information. Numerous benefits have been ascribed to streaming media tutorials [197], which take advantage of humans' cognitive abilities described by dual coding theory[1] [153], the modality effect[2] [91], and others. There is also evidence that videos assist users more with unfamiliar tasks than traditional textual help [96].

The content of the tutorials was based on our several years' experience tutoring participants in using novel authentication schemes. We attempted to make the tutorials as effective learning tools as we could within the constraints of the modality. For example, we tried to make the tutorials as easy to navigate as possible, whether participants were reading through the tutorial sequentially, or browsing for particular information. In our experience, ease of navigation is important, since otherwise users may become frustrated and give up on the tutorial and be unable to create a password. Users were always shown one of the tutorials before creating a password, and could navigate the tutorials freely, either by clicking on a particular section or viewing them in the prescribed order. The tutorial could be ended at any time. While using PCCP, users could press a *Help* button to display the tutorial in a new window, so it may be referenced alongside the password system.

6.4 Methodology

To compare the four types of tutorials, we performed two between-subjects user studies; one locally and one online using Amazon Mechanical Turk [3] (MTurk). We chose to run these two studies since they each provide different qualities of information. By observing local participants, we can collect qualitative information on participants' behaviour and expressions during the study, which can provide a deeper understanding of the quantitative results. However, MTurk participants perform the entire study in their normal environment, without an experimenter who may otherwise influence users' behaviour unintentionally. Thus, MTurk participants are more likely to provide more realistic quantitative data, at the cost of poorer qualitative data.

[1] Dual coding theory postulates that information is cognitively represented in both textual and visual representations, and both representations can be processed in parallel.

[2] The *modality effect* asserts that, when presented with visual information, people learn better from accompanying auditory information rather than textual information.

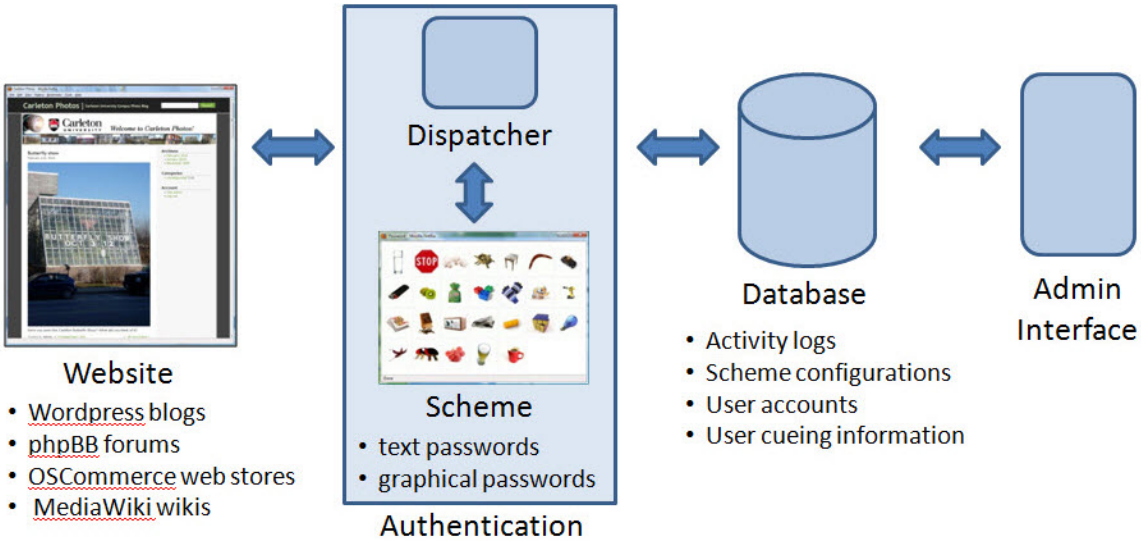


Figure 6.6: MVP architecture. [38]

Our user study was performed with Multiple Versatile Passwords (MVP) [38] (Figure 6.6). MVP is an the authentication evaluation system we currently use in experiments with involving users and authentication schemes. Participants can remotely access live realistic websites that are similar to ones users typically visit. Experimenters can implement and upload authentication scheme prototypes to MVP, and assign schemes to particular users and websites. MVP provides functions for prototypes to output logging data to a MySQL database on the server. MVPis the latest evolution of our continually-improving user study methodology. The flexible architecture of MVP was also instrumental in the design of CYOA (Chapter 7).

In this study, users created their first password when registering for the *Student Life* website (Figure 6.7), where people share advice and experiences related to post-secondary studies. Users' second password was registered with the *World Vacation* website, where photos and stories of places users have visited or would like to go are posted for discussion. The third and final website users registered for was *Vote for You*, where users may post polls for others to vote on and discuss the results.

Participants were told the study was about overall website usability (including registering and logging in), in order to focus their goal on performing the specific website-related tasks, and make the password-related activities a secondary task. Over the course of one week, participants performed the following tasks (Figure 6.8):

Figure 6.7: PCCP login window overtop the MVP Student Life blog site.

Day 0: Users created their first password and account on Student Life. They logged in, performed a task, and answered a questionnaire. MTurk participants were e-mailed the experiment's instructions, while local participants visited our lab. To minimise outside influence, the experimenter left the room while local participants reviewed the tutorial and created their account.

Day 2: All users were e-mailed a request to visit the second website, World Vacation, and create an account, login, and perform a task.

Day 4: All users were sent an e-mail requesting they create an account, login, and perform a task on the third website, Vote for You.

Day 7: Local participants returned to the lab to log in to the three websites, perform a task on each, and complete a final questionnaire. MTurk participants were e-mailed the same instructions.

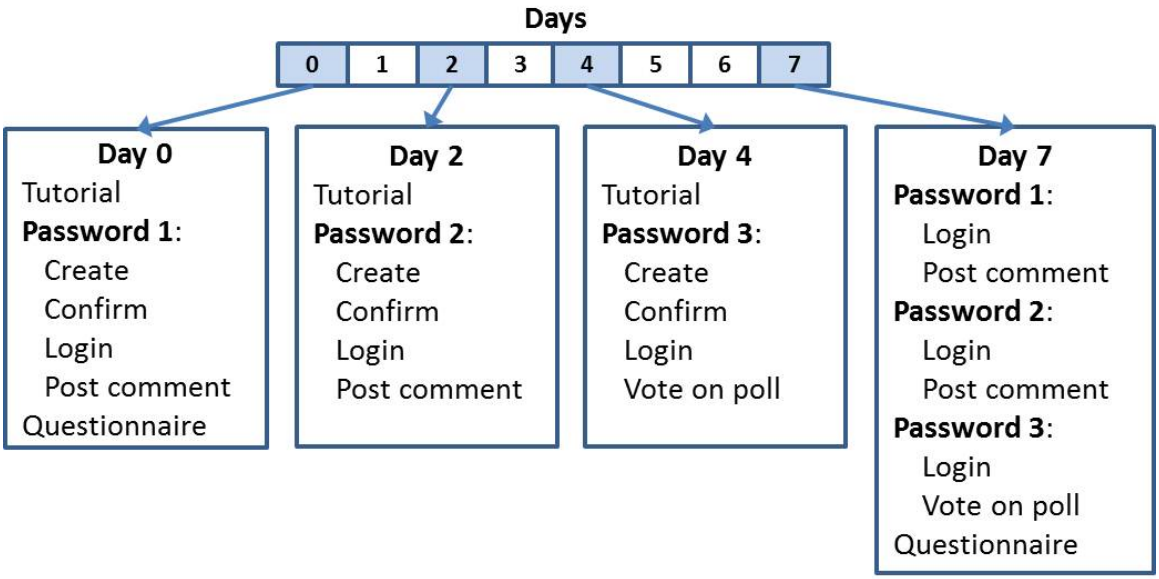


Figure 6.8: Timeline of study tasks over the course of one week.

The materials used in this study can be found in Appendix C. Participants were e-mailed the instructions for tasks completed at home (days 2 and 4 for local users and all days for MTurk users). We label the start of the study as day 0 (rather than day 1) so that conceptually "day n" is n days since beginning the study. Local participants were compensated with $20 or course credit at the end of the study. MTurk participants were paid $1 for their participation on each of the days listed above, as approved by our ethics review board. Rates differed in the two studies to follow established norms. MTurk rates range from $0.01 to $1.00 [47].

A total of 134 people participated in our study. Table 6.1 summarises the demographics for local and MTurk participants in each condition. The numbers of participants in each condition are slightly unbalanced due to untimely drop-outs. Two-thirds of our local participants were female, while thirty percent of MTurk participants were female. Our participants ages ranged from 17 to 55 for local and 18 to 50 for MTurk. We also asked participants to rate their computer skills on a scale of 1 (novice) to 10 (expert). Participants in both groups rated themselves a median of 8, which suggests they felt very confident in their computer abilities. All local and about 40% of MTurk participants were university students from various disciplines. The non-student MTurkers' occupations varied, from unemployed to a company's

	Local	MTurk
Text	12	25
Hypertext	9	24
Demo	10	20
Video	10	24
Males	14	62
Females	27	31
Age (Min)	17	18
Age (Median)	20	26
Age (Max)	55	50
Computer Skills (Min)	5	3
Computer Skills (Median)	8	8
Computer Skills (Max)	10	10

Table 6.1: Number of participants and their demographics.

"Vice President - Materials". No participants reported studying or working in computer security.

6.5 Hypotheses

We initially thought that the demo tutorial would be most successful, since people generally learn better with greater engagement [77]. This is the case for demo users, who actively interact with the tutorial in creating, confirming, and logging in with a practice password. Using Grossman et al.'s [97] *categories of learnability metrics*, we used their task and documentation metrics to formulate our hypotheses along four pillars:

Investment: Demo participants will spend more time on the tutorial than participants in other conditions.

Learnability: Demo participants will spend less time registering their password than participants in other conditions.

Security: Demo participants will shuffle (Section 2.3) less than participants in other conditions.

Memorability: Demo participants will successfully login more often than participants in other conditions.

6.6 Results

We analyse our study's results along these same four pillars: *Investment* measures the time invested in looking at tutorial materials. *Learnability* evaluates the effectiveness of the tutorials at imparting their material to users. *Security* evaluates password strength. *Memorability* examines how easily users could recall their passwords.

We use various statistics to determine if two or more distributions are significantly different. For normal distributions, we use t-tests and one-way ANOVAs. For data not normally distributed, we use Mann-Whitney U and Kruskal-Wallis (KW) tests instead. For categorical data, we use chi-squared or Fisher's Exact tests, depending on category counts. The Bonferroni correction is applied as appropriate when doing multiple comparisons. In all tests, we accept $p < .05$ as statistically significant.

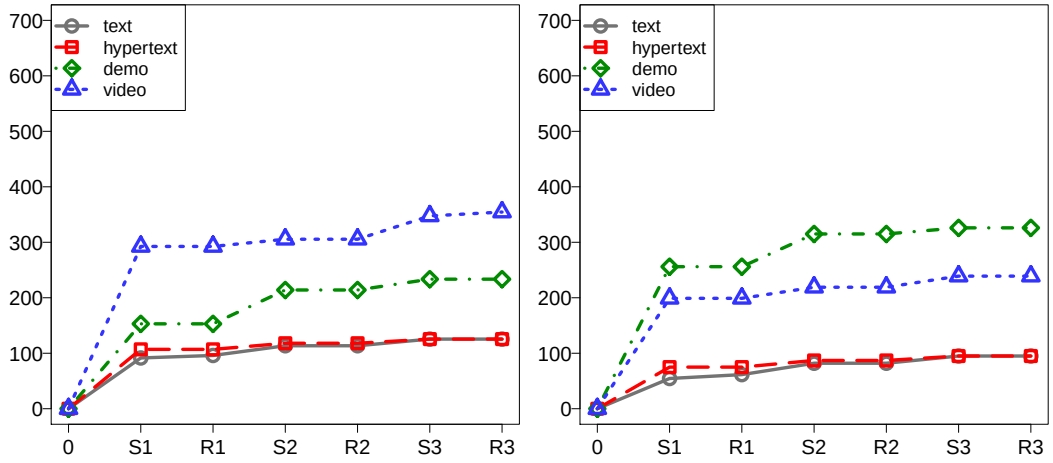
We analyse the local and MTurk data in parallel rather than together because factors such as Internet speed, language or cultural barriers, or other unknowns may have affected results in ways that our experimental design was not intended to measure. Where appropriate, we informally identify differences between both studies, but they should be further explored in experiments specifically measuring cross-study effects.

6.6.1 Investment

We measure investment by the time users spent reviewing the tutorial and user perception of the tutorial. The less time users need to spend on the tutorial, the sooner they can register and resume their primary tasks. All other factors being equal, we believe less time invested into a tutorial would be a positive result. The time spent viewing the tutorial before and during password creation are equivalent to Grossman et al.'s [97] documentation metrics *D2 (time taken to review documentation until starting a task)* and *D1 (help commands used over certain time interval)* respectively.

Tutorial times

We first examined the time users spent with the tutorial. Figure 6.9 plots the cumulative median time participants spent looking at tutorial material. The medians are a better representation of the tutorial times than means, since a few outliers skewed the

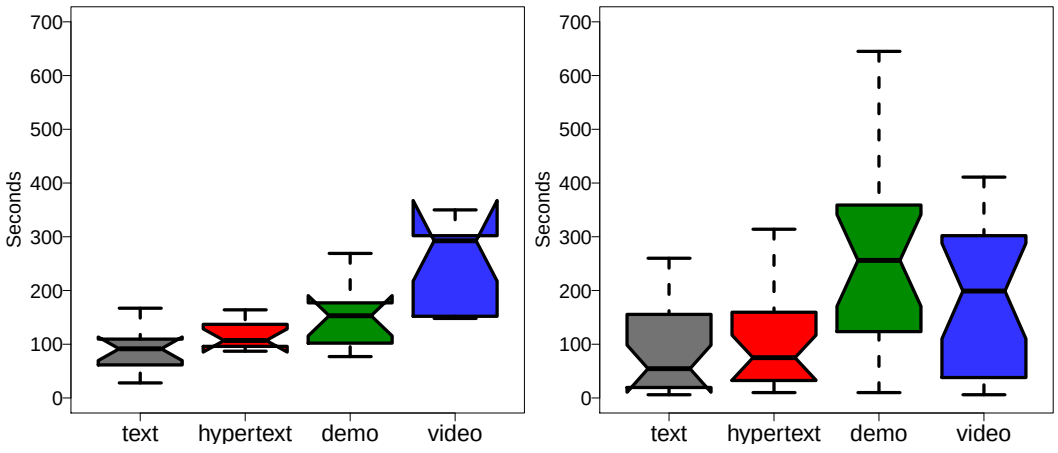


(a) Local (b) MTurk

Figure 6.9: Cumulative median tutorial times (in seconds) per condition. (S)tart and (R)evisit time spent on tutorial material when registering the 1st, 2nd, and 3rd time.

averages. The x-axis represents the occasions when participants could have viewed the tutorial: at the start of a session (*start, S*) and revisits during registration (*revisit, R*), for each of their three passwords separately. The slopes between points illustrate the median amount of time users spent at each tutorial viewing. We include the origin (0,0) to illustrate the amount of time users spent when first viewing the tutorial (*S1*). For example, as we might expect, users spend a lot of time with the tutorial when they first see it (*S1*), before ever using PCCP for the first time. This time spent on the tutorial is shown by the steep slopes between the *0* and *S1*. Conversely, after having just viewed the tutorial, users may spend little to no time on the tutorial when registering their first password (*R1*). The time spent on the tutorial while registering for their first password is show by the slope from *S1* and *R1*. Similarly, users spend little to no time on the tutorials thereafter throughout the study, shown by the very shallow slopes between the points of *R1*, *S2*, *R2*, *S3*, and *R3*.

We compared start and revisit tutorial times across conditions. Participants spent significantly different times on their tutorials during the first visit (*S1*), but not thereafter. As reported in Table 6.2, Kruskal-Wallis tests show that overall differences exist between conditions for *S1*. In our post-hoc analysis, six Bonferroni-corrected Mann-Whitney tests identify where these differences were found. Specifically, video participants initially spent longest on the tutorial for the local study, while demo



(a) Local (b) MTurk

Figure 6.10: Time (in seconds) initially spent on the tutorial at the first visit (*S1*).

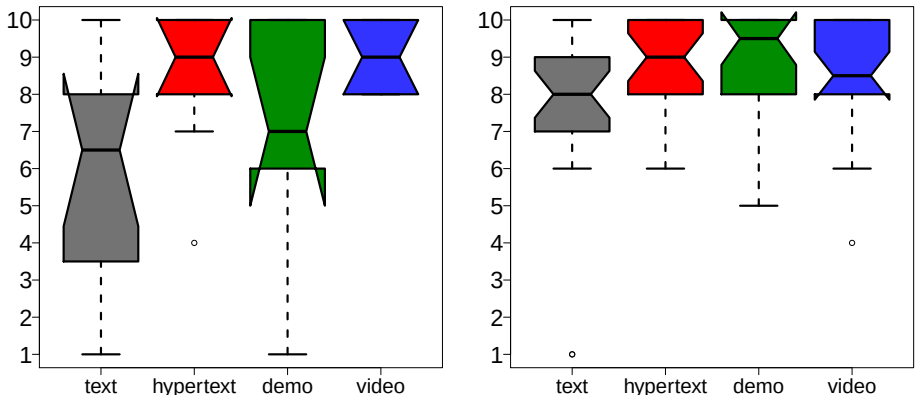
Users	Conditions	Test Result
Local	All	KW $\chi^2(3) = 21.37, p < .0001$
	Video vs Text	$U = 117, p < .0005$
	Video vs Hypertext	$U = 86, p < .005$
	Video vs Demo	$U = 85, p < .05$
MTurk	All	KW $\chi^2(3) = 14.93, p < .005$
	Demo vs Text	$U = 361, p < .01$
	Demo vs Hypertext	$U = 338.5, p < .05$

Table 6.2: Differences between conditions on the time initially spent on the tutorial during the first visit (*S1*). Only statistically significant results are reported.

participants spent the longest for the MTurk study (although the difference between demo and video was not significant, due to high variance). The boxplots in Figure 6.10 illustrate these start times spent on the tutorial at the first visit (*S1*). Boxplots show the distribution with the median at the centre, the box showing the central quartiles, the whiskers showing the outer quartiles, and the notch showing the 95% confidence interval.

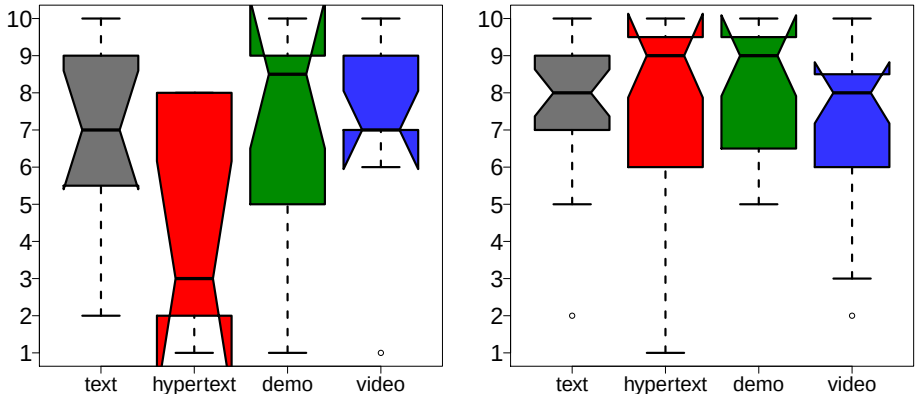
User perception of the tutorial

On day 0, users completed a questionnaire after registering their first password. Among other topics, the questionnaire included 10-point Likert scale questions about the tutorial, where 1 represents *strongly disagree* and 10 represents *strongly agree*. All



(a) Local (b) MTurk

Figure 6.11: Day 0 questionnaire response for how easy the tutorial was to understand.



(a) Local (b) MTurk

Figure 6.12: Day 0 questionnaire response for how quickly users felt they could complete the tutorial.

user perception Likert-scale topics had both a positively- and negatively-phrased question for counter-balancing. However, in the analysis, we eliminated questions whose results showed evidence that users may not have understood the wording. Figure 6.11 shows responses regarding how easy participants found their tutorial to understand. Figure 6.12 illustrates responses regarding how quickly they felt they could complete the tutorial. In all Likert-scale results reported, higher numbers represent positive responses.

No significant differences were found between conditions for either question. This suggests that participants in any one condition felt their tutorial was no harder to understand nor slower to navigate than participants in any other condition.

Local responses show high variance. This is possibly a result of a small sample size, since the largest local condition (text) only had 12 participants. MTurk responses appear more consistent. In particular, MTurkers rated their respective tutorials very easy to understand and quick to navigate across all conditions.

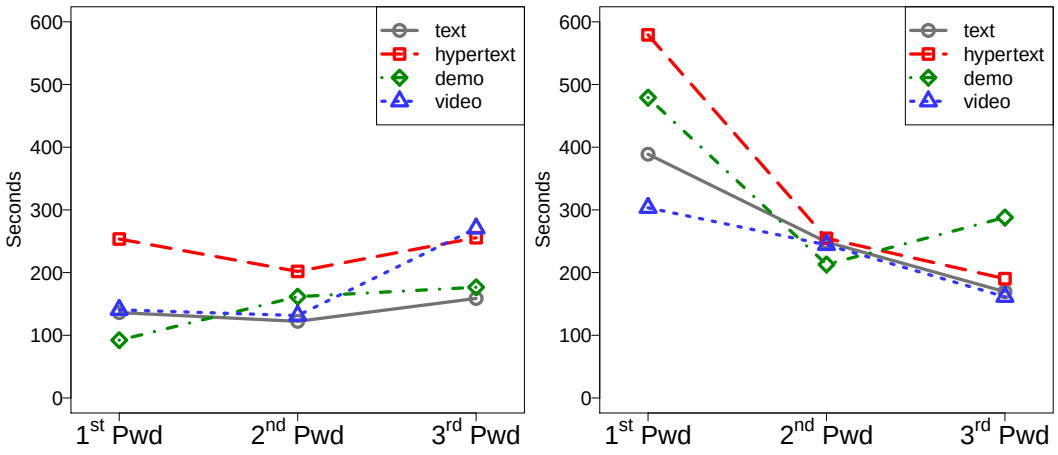
6.6.2 Learnability

We define *learnability* as the tutorial's effectiveness in enabling users to create and login with the password system efficiently. If participants understood the tutorial and learnt to use the scheme, then they should be able to quickly create and login with few to no errors. Thus, we measure learnability by the time taken to successfully register and log in to a new account, and by the number of times users restarted the registration process. Respectively, these measurements are the same as Grossman et al.'s [97] task metrics *T5 (time until user completes a certain task successfully)* and *T4 (task errors made over a certain time interval).*

Registration time

Our first learnability metric is *registration time*; the time users took to successfully complete password registration (create, confirm, and login). Figure 6.13 displays the mean registration times by condition. In this and subsequent line graphs, the connecting lines are only included for readability. No significant differences in registration time between the conditions were found for either local or MTurk participants.

Local participants registered their first password more quickly than MTurk users (Figure 6.14). We believe that this may be due to the lab environment where local participants were paying full attention to the tutorial. Conversely, MTurk users may have been multi-tasking in less focused environments while performing the study tasks, thus requiring more time to register. This difference in registration time between local and MTurk participants was not visible for the second and third passwords (Figure 6.13), presumably because users needed less cognitive effort for subsequent registrations, having already successfully completed the process at least once.



(a) Local (b) MTurk

Figure 6.13: Mean registration times (in seconds) to register each of the three passwords.

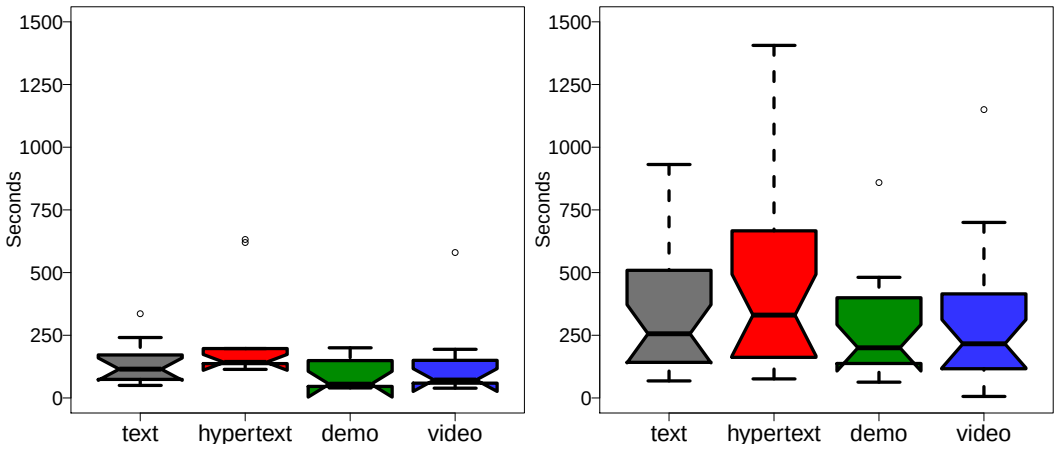
Registration restarts

We next examined how often users restarted the password registration process. We use this as a learnability metric because restarting a task is an indication of misunderstanding or insufficient instruction. Ideally, the task and instructions should be easy enough to understand, so users can avoid irrecoverable errors that require restarting. We found no significant differences between conditions for any of the three passwords.

6.6.3 Security

The tutorials included advice on choosing secure PCCP passwords. The *security* dimension is measured by the randomness of users' chosen passwords. In PCCP, the less users shuffle, the more random their password (see Section 6.2). PCCP is designed to evenly distribute users' click-points across the image. Thus, we measure security by the number of shuffles and users' perceptions of whether the tutorial material helped them choose a more secure password. Shuffles could be considered analogous to Grossman et al.'s [97] task metric *T7 (quality of work performed during a task)*.

We considered performing click-point distribution and hotspot analysis on participants' click-points. However, we believe this is out of this chapter's scope, since we are not evaluating the authentication scheme, which has already been done [46].



(a) Local (b) MTurk

Figure 6.14: Time spent on the 1st password's registration.

Shuffles

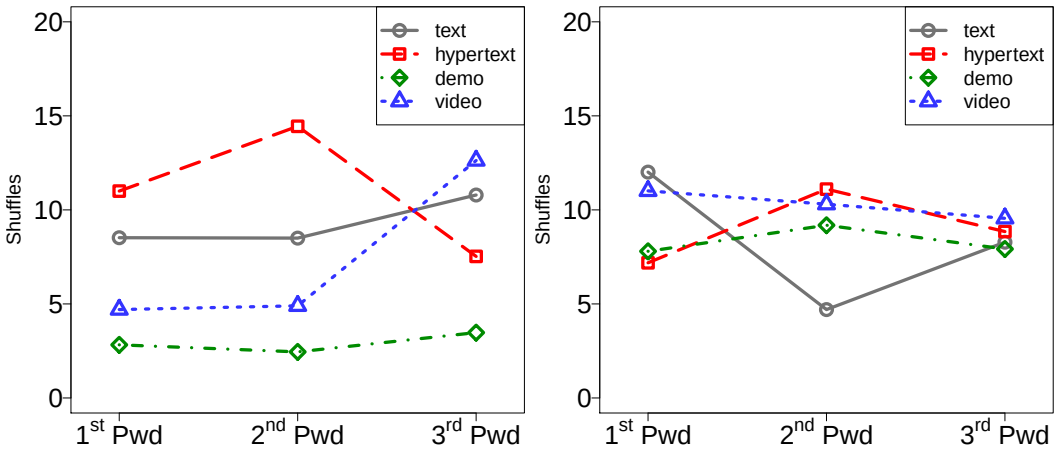
Our tutorials were designed not only to teach how to use PCCP, but also how to select *secure* passwords that would be difficult for an attacker to guess. Thus, our tutorials contained material explaining shuffling, and encouraged users to shuffle only when necessary.

Figure 6.15 plots the mean number of shuffles per click-point for each of the three passwords. There was a large variance in the number of shuffles, but on average users shuffled moderately. There were no significant differences in shuffling between passwords or conditions for either local or MTurk participants.

All four MTurk conditions seem to converge on 10 shuffles per click-point by the third password. This shuffling rate per image is similar to those in the online study by Chiasson et al. [46]. We are less concerned that our shuffling rates seem higher than Chiasson et al.'s in-lab studies [46], since it is possible their participants may have been unintentionally influenced to behave more securely by the laboratory setting.

User perception of security

We asked participants to rate on a 10-point Likert scale how strongly they agreed or disagreed with a statement that the tutorial helped them to create more secure passwords. As shown in Figure 6.16, local participants seemed neutral towards the



(a) Local (b) MTurk

Figure 6.15: Mean number of shuffles per click-point.

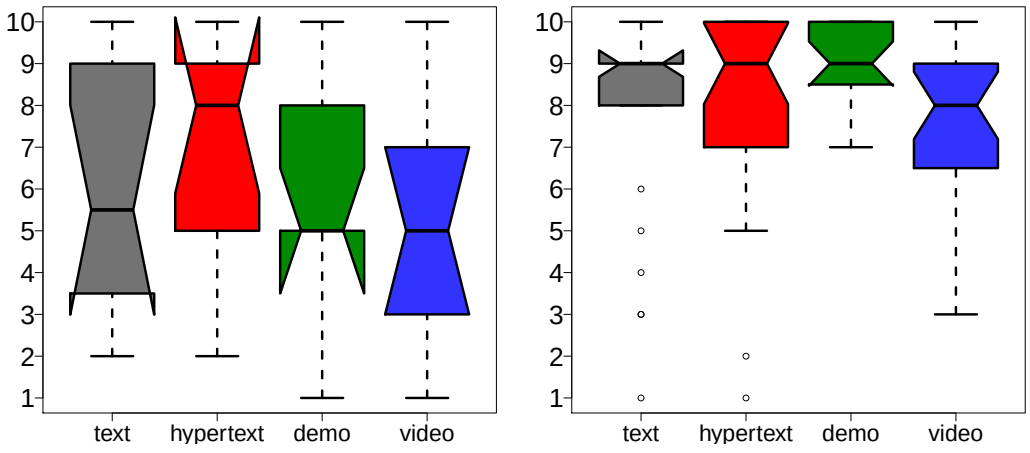
statement, while MTurk users generally felt the tutorial did help them in choosing secure passwords. There were no significant differences between conditions in either the local or MTurk responses.

6.6.4 Memorability

We evaluated *memorability* by looking at whether users were able to remember their passwords on the last day of the study (day 7). Thus, we measure memorability by the proportion of users who successfully logged in at the study's end. Successful logins are closest to Grossman et al.'s [97] task metric *T2 (percentage of users who complete a task without any help)*, considering password reset request as "help".

Success rates

Figure 6.17 shows the login success rates on day 7. Local participants returned to the lab for this final login to each website while MTurk participants received an email requesting that they log in to each site. We considered the login successful if the participant was able to login on day 7 without having to reset their password registered on day 0. We found no significant differences in success rates across conditions. We speculate that the local study's small sample size accounts for its higher variance.



(a) Local (b) MTurk

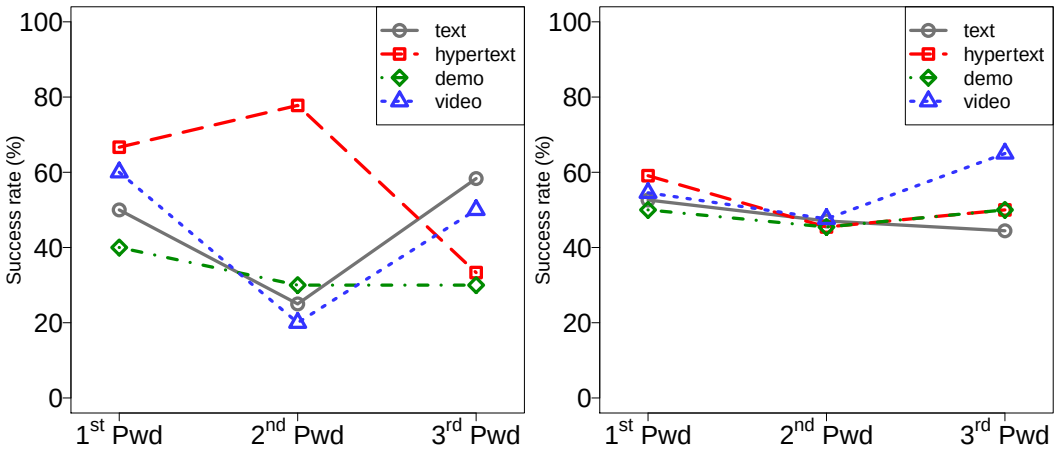
Figure 6.16: Day 0 questionnaire response for whether the tutorial helped them create secure passwords.

End-of-study no-shows & immediate resets

The success rates in Figure 6.17 exclude 22.5% of MTurk participants who did not return to complete day 7's tasks (*no-shows*). Table 6.3 shows the distribution of MTurk no-shows across conditions. We see significantly more no-shows in the text and demo conditions ($\chi^2(3) = 8.20, p < .05$).

We further examined the number of local and MTurk participants who chose to reset their password immediately on day 7, without first attempting to login. Results are shown in Table 6.4. As with the no-shows, it appears that text and demo participants immediately reset their password more frequently than those in other conditions. However, this difference is only significant for local participants ($Fisher, p < .05$), and not MTurk ($\chi^2(3) = 7.44, p = .06$).

We believe that the most likely explanation is that these users had forgotten their passwords, and decided not to try at all. This may provide evidence that text and demo participants' created less memorable passwords than hypertext and video participants.



(a) Local (b) MTurk

Figure 6.17: Login success rates for the three passwords on day 7.

Users	Cond	Returned	No-shows (%)
MTurk	Text	16	9 (36%)
	Hypertext	21	3 (13%)
	Demo	11	9 (45%)
	Video	20	4 (17%)

Table 6.3: Number of day 7 no-shows for the MTurk participants. All local participants returned for day 7.

User perception of memorability

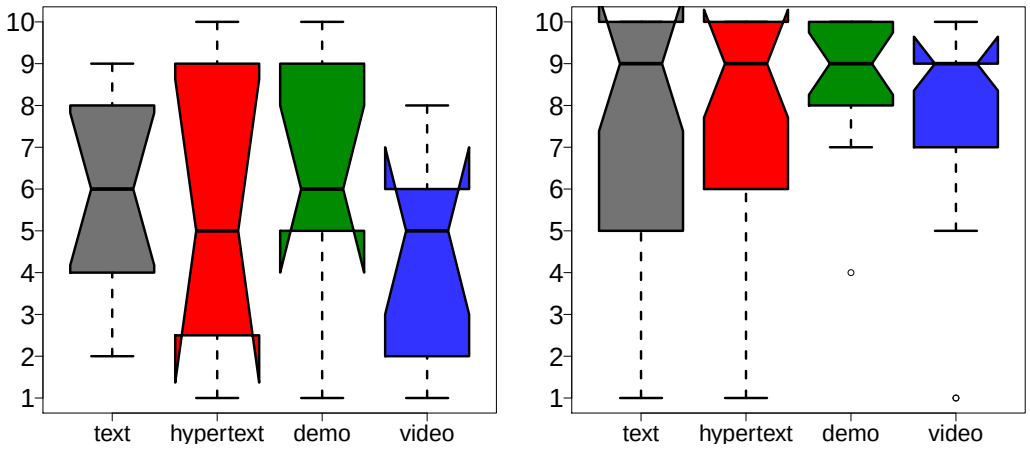
At the end of day 0, users were asked to rate how much they felt their tutorial helped them create more memorable passwords (Figure 6.18). Statistical tests show no significant differences in responses between conditions. Responses from local participants were neutral while MTurkers felt their tutorial was helpful in creating more memorable passwords.

6.7 Interpretation

We revisit our hypotheses in light of the results of the studies, which demonstrate that the demo tutorial did not have the effect we predicted. We will then discuss possible reasons why the other tutorials were surprisingly more effective than the demo. We propose a simple evaluation framework based on our four evaluation pillars and rank the tutorials accordingly. Finally, we discuss differences between the two studies.

Users	Condition	Attempted login	Immediately reset
Local	Text	8	4 (33%)
	Hypertext	9	0 (0%)
	Demo	4	6 (60%)
	Video	9	1 (10%)
MTurk	Text	13	3 (19%)
	Hypertext	18	3 (14%)
	Demo	8	3 (27%)
	Video	17	3 (15%)

Table 6.4: Number of day 7 attempted login and immediate resets.



(a) Local (b) MTurk

Figure 6.18: Day 0 questionnaire response for whether the tutorial helped them create memorable passwords.

6.7.1 Hypothesis Testing

Our hypotheses favoured the demo tutorial to best educate users, at the cost of requiring more time to complete the tutorial. We will now test each hypothesis.

Investment: *Demo participants will spend more time on the tutorial than participants in any other condition.* We found little support for this hypothesis since it holds true only when compared to the MTurk text and hypertext conditions. As reported in Section 6.6.1, local video participants spent significantly more time than demo participants with the tutorial when first seeing it. MTurk demo participants spent significantly more time initially on the tutorial than text and hypertext, but not video.

Learnability: *Demo participants will spend less time registering their password than participants in any other condition.* We found no support for this hypothesis, as there were no significant differences in registration times between conditions (Section 6.6.2).

Security: *Demo participants will shuffle less than participants in any other condition.* We found no support for this hypothesis, as there were no significant differences in the number of shuffles between conditions for either study (Section 6.6.3).

Memorability: *Demo participants will have higher login success rates than participants in any other condition.* We found no support for this hypothesis because demo participants avoided re-entering their password because they had forgotten them more often than hypertext and video participants. We also found no significant differences in success rates between conditions in either study (Section 6.6.4).

Contrary to our expectations, the richer interaction and immediate feedback provided by the demo tutorial showed no greater benefit over the other tutorials.

6.7.2 Demo Tutorial Retrospective

There are a number of reasons we believe the demo tutorial did not outperform the other tutorials, contrary to our hypotheses.

Memory Interference. The demo tutorial had participants create, confirm, and login with a practice password, while users in other conditions did not. This practice password may have interfered with users' memory of their real passwords, making the latter more difficult to remember.

Deviations from the Minimal Manual. Of all the tutorials, demo deviated the most from the Minimal Manual recommendations [32]. For example, the Minimal Manual suggests indexing the material, so users can find the particular information they want easily. However, the demo required that users create a practice password before allowing access to the information available in later steps. We had purposefully designed the demo tutorial this way to leverage the Persuasive Technology *tunnelling* principle of guiding users through a process and providing them with information and feedback throughout. This study's result may be evidence that PT principles may not be as effective for authentication system tutorials as they are in other domains.

Security is always a secondary task. We had given users a website-related task to perform, which required them to create an account and login. Their primary task was to perform the website-related task, not learn how to use a new authentication mechanism. Users may not have wanted a rich, interactive demo that was time-consuming and keeping them from completing their primary task. Therefore, we believe that when users are in the process of registering a password, they may gain more benefit from *less* interactive media, such as our hypertext or video tutorials.

6.7.3 Evaluation

Overall, participants were able to learn how to register and login with PCCP with any one of the tutorials. Although no tutorial was consistently more effective than the others, we wished to derive some insights on which modality may be most promising to pursue in future work. Based on our hypotheses' topics, we ranked the tutorials along the four pillars of Investment, Learnability, Security, and Memorability. Table 6.5 shows relative scores we gave to each condition for each dimension, ranging from 0 (worst) to 3 (best). When no differences were found between conditions, we divided the points evenly.

Investment. The video tutorial received the lowest investment score (0) because local video participants spent the most time on their tutorial compared to participants in any other local condition. Demo received the second lowest score (1) because MTurk demo users spend more time on the tutorial than text or hypertext participants. We found no differences between text and hypertext, so we awarded each condition the average of the top two scores (2 & 3 = 2.5).

Learnability. All conditions were given the average of all possible learnability scores (0, 1, 2, & 3 = 1.5), as no differences in learnability were found between the participants in different tutorial conditions.

Security. We awarded all conditions the average of all possible security scores (0, 1, 2, & 3 = 1.5), because we found no difference in the security of the passwords chosen by users in different conditions.

Memorability. We averaged the two lowest memorability scores (0 & 1 = 0.5) for the text and demo tutorial, since those participants more often avoided attempting

	Text	Hypertext	Demo	Video
Investment	2.5	2.5	1.0	0.0
Learnability	1.5	1.5	1.5	1.5
Security	1.5	1.5	1.5	1.5
Memorability	0.5	2.5	0.5	2.5
Total	**6.0**	**8.0**	**4.5**	**5.5**

Table 6.5: Comparative scores for each condition along our evaluation pillars (Investment, Learnability, Security, and Memorability). Each dimension's score ranges from 0 (worst) to 3 (best).

to login at the end of the study. Hypertext and video shared the two highest scores (2 & 3 = 2.5), as there were no differences in success rates or user opinion.

Summing the scores for each condition in Table 6.5, the hypertext tutorial ranks the best overall modality of the four we tested, as hypertext always tied for the highest possible score for each dimension. We emphasise that this is only an informal ranking and that our study results found few statistically significant differences between conditions. However, Stobert and Biddle [189] ran several experiments where over 300 participants successfully learnt to use an novel authentication scheme with nothing more than tutorial similar to our text tutorials. Together, our studies suggest that simpler text or hypertext tutorials are sufficient for teaching users novel authentication schemes.

6.7.4 Low Success Rates

Login success rates were much lower than expected, given the higher rates reported in the *PCCP Web* study by Chiasson et al. [46]. Our participants seemed to only able to login about 50% of the time (without considering any errors they may have made), while 54% of PCCP Web participants [46] were able to login without a single error a week after creating the password.

However, there are noteworthy differences between these studies. Firstly, the PCCP Web study participants were asked to login to their accounts three more times during the week than our participants. This prompted PCCP Web participants to practice their password more often, which is likely why they could better recall their passwords at the end of the study.

Secondly, in all of Chiasson et al.'s user studies, users had one-on-one training with PCCP from an experimenter, as well as the opportunity to practice with the scheme before creating the passwords used throughout the study [46]. This may have given their users more opportunity to learn or develop strategies for creating memorable passwords. Of course, this comes at a higher cost of one-on-one training and time required to practice with PCCP.

6.7.5 Local vs MTurk

We noted several differences between our two studies. We are reluctant to statistically compare the two user groups since this was not the purpose of this study. We did not expect to see such differences given that previous research comparing the two user study methods found them to provide equivalent results [123]. This dichotomy suggests further research is required to determine when the two types of studies may be interchangeable.

We also questioned whether other factors may have influenced MTurk participants' user perception Likert-scale responses because they seemed *too* consistently positive. It was possible that participants blindly agreed with all statements to quickly complete the questionnaire, or to give us the positive reviews they believed we wanted. However, only two (out of 111) MTurk users gave the same response (10) for all questions, giving some indication that MTurkers actually considered their responses.

We feel the MTurk results may be more ecologically valid than the local results. MTurk users may have had distractions while performing the study tasks. We believe the MTurkers' situation and motivations probably match those of real end-users registering for an account to perform a task or access a resource.

6.8 Conclusion

Before creating a password, our participants were presented with one of four tutorials; text instructions with an image, smaller hyperlinked sections of text with several images, an interactive demo with immediate feedback, or a narrated screen-captured video of interactions with the password scheme. Over one week, users were asked to create and recall three different passwords to perform primary tasks on three websites.

Among the alternative authentication schemes, we chose Persuasive Cued Click-Points (PCCP) for our study since it seems to offer reasonable security and usability, but is not straightforward to learn.

We hypothesised that the demo tutorial would help users quickly create more memorable and secure passwords than the other conditions, at the cost of more time spent on the tutorial. Although MTurk demo users did spend the most time on the demo tutorial, we did not find evidence to support our four hypotheses.

Users were able to independently learn to create passwords and login to their accounts. In fact, users surprisingly did not need an interactive demo; a hypertext or video tutorial was sufficient, and even preferred. This is good news for authentication researchers and developers, since they may not need to spend extra time and effort building elaborate demos to get users started with new password systems. For researchers, assessing the learnability and user investment in their novel authentication schemes should be part of the design and proposal.

This chapter presents the first research work on teaching people to use unfamiliar authentication schemes. Our user study demonstrates that users can learn to authenticate with a novel scheme with a tutorial, preferably a simple text or hypertext tutorial. This result was confirmed in work performed in parallel by Stobert and Biddle [189]. Over 300 participants in their studies learnt to use a different novel authentication scheme with a tutorial similar to our text tutorial. Together, these studies suggest that users can cope well with learning novel authentication schemes. The next step is to discover a way to provide each user with an authentication scheme that best suits their abilities, preferences, and usage context.

Chapter 7

Choose Your Own Authentication

7.1 Introduction

A wide variety of authentication schemes have been proposed (Chapters 2, 4, and 5). Each scheme has strengths and weaknesses. For example, Persuasive Cued Click-Points [40, 46] (PCCP) has considerable security and usability advantages over other graphical password schemes, but PCCP would only be appropriate for accessing systems in private locations with little risk of shoulder-surfing. Gaze-based schemes like Cued Gaze-Points (Chapter 5) may compensate with shoulder-surfing resistance at the cost of some security and usability, as well as requiring the presence of eye tracking hardware. Another example is Persuasive Text Passwords (PTP, Chapter 4), which helps users create more secure text passwords. However, multiple PTP passwords may be too difficult to remember, so PTP is best suited for high-risk and high-value accounts requiring a text password. It seems unlikely that a single scheme will ever be universally usable and sufficiently secure for all possible users and applications. Indeed, our User-Centred Authentication Feature Framework (Section 3) has shown that a large variety of schemes with different features can positively affect the user experience when authenticating. Thus, a better approach would be for different schemes to be used in different contexts. Users should be able to authenticate to systems with a scheme that offers sufficient security, usability, and accessibility for the specific user, account, and threat model.

But how can an authentication system know a user's preferences and requirements for security, usability, and accessibility? People may have different preferences for schemes supporting a particular memory modality, input modality, cueing, obfuscation, and other security and usability features. These preferences may also change for different types of accounts. For example, users may prefer a scheme with high password memorability for a seldom-accessed account or a high-entropy scheme for

an online bank account. Users' priorities may change yet again for different login circumstances. They may desire a shoulder-surfing resistant scheme for an account they access in public. Given all these variations in people's preferences, priorities, and circumstances, it seems infeasible for a system to know which scheme would best suit a new user.

We propose Choose Your Own Authentication (CYOA), an authentication system that allows the user to choose their desired method of authentication amongst a selection of schemes approved by system administrators. The available schemes should be rated along specific security and usability measures that non-experts can understand and use to make the most appropriate scheme selection for their needs.

The remainder of this chapter will proceed as follows. We will first discuss the differences between CYOA and proposals of similar intent or scope in Section 7.2. We will then describe CYOA's user experience (Section 7.3), followed by two possible architectural designs for CYOA, including their benefits and caveats in Section 7.4. Our prototype implementation of CYOA in our MVP experimental authentication system will be described in Section 7.5. We will then discuss the design of a future usability evaluation of CYOA in Section 7.6. Finally, we offer some concluding remarks about CYOA in Section 7.7.

7.2 Background

The modularisation of authentication schemes was first proposed by Sun Microsystems as Pluggable Authentication Modules (PAMs) [137] for Unix-based systems. PAMs allow applications to call standardised authentication functions without concern of what authentication scheme or password hashing method is actually in use. Operating systems supporting PAMs also allow developers to build and use their own authentication scheme. Unlike CYOA, PAM only allows a single authentication scheme to be available at a time. Thus, PAM end-users are restricted to using whatever authentication scheme was implemented by the system administrators.

Section 2.6.1 briefly describes *single sign-on (SSO)* proposals. SSO protocols generally involve three parties: the user (or client), the *relying party* (*RP*, who controls the service or resource the user wishes to access), and the *identifying party* (*IdP*, who

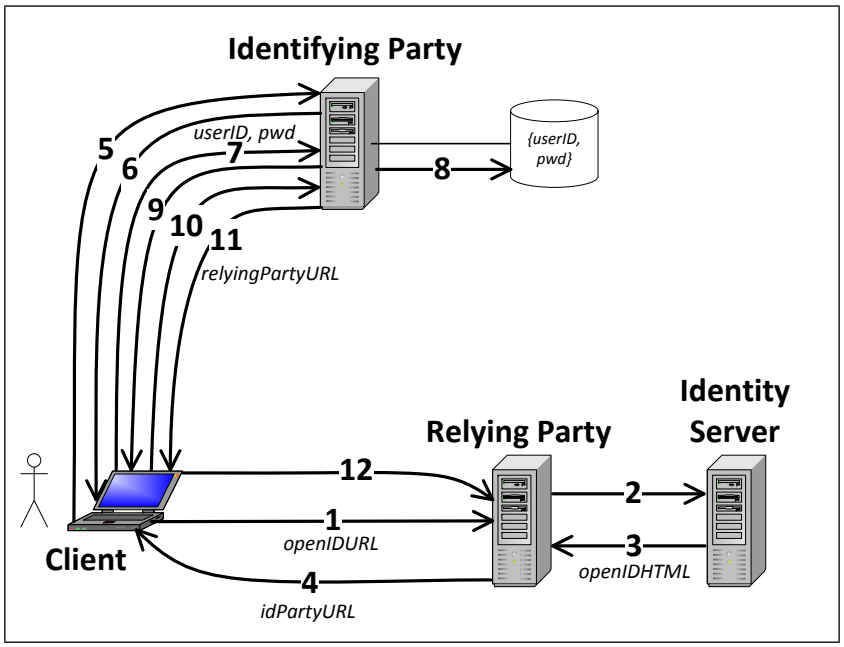


Figure 7.1: OpenID [148, 149, 167] protocol.

ultimately authenticates the user). OpenID [148,149] is the best-known SSO solution. As discussed in Section 2.6.1, a user's identity is defined by their OpenID, which is a URI (controlled by the user) that points to an XML document containing the location of the user's IdP. An OpenID authentication process can be summarised as follows (where the steps noted in parentheses correspond to the numbered arrows in Figure 7.1):

- The user requests access to a service by providing their OpenID URI to the RP controlling the service (step 1).
- The RP parses the XML document at the user's OpenID URI, extracts the location of the user's IdP, and redirects the user to their IdP (steps 2 to 5).
- The user authenticates to their IdP with their username and password, proving ownership of the OpenID (steps 6 to 8).
- Assuming the user authentication is successful, the user sends the identifying party confirmation of user trust in the relying party (steps 9 and 10).

- The IdP redirects the user to the RP, and the IdP confirms to the RP that the user successfully authenticated for the provided OpenID (steps 11 and 12).

SSO solutions like OpenID directly address a different aspect of the Password Problem (Section 1.1) than CYOA. SSOs primary purpose is to reduce the number of passwords users need to remember, thereby lowering the memory burden on users, which potentially enables them to focus their efforts on remembering a single (or a few) more random, and hence secure, passwords. CYOA is different in that it offers users the choice of several authentication methods for each system. This has several benefits we will describe in later sections. Unlike SSO solutions, CYOA users' accounts for different websites must be logged into separately. However, SSO and CYOA are complementary solutions: an SSO identifying party could use CYOA to authenticate its users.

A related proposal named OAuth [101, 102] is an IETF standard [109] for a protocol to allow an entity (relying party) to access some resource owned by a second-party (the resource owner) and controlled by a third-party (the resource controller), without requiring the resource owners' credentials. For example, a user (as the resource owner) could authorise their social network website (the relying party) to access specific images on their photo-sharing website (the resource controller). The OAuth protocol proceeds as follows:

1. The relying party (RP) requests authorisation to access a resource from its owner.

2. Provided the resource owner wishes to grant the RP authorisation to access the resource, the owner provides the RP with an *authorisation grant* containing the credential representing the owner's authorisation.

3. The RP requests an access token for the resource from the authorisation server by sending it the aforementioned authorisation grant.

4. The authorisation server authenticates the RP and verifies that the authorisation grant is valid. Provided the authentication and validation are successful, the authorisation server returns an access token for the resource to the RP.

5. The RP requests access to the resource by sending the access token to the resource server.

6. The resource server validates the access token and provides the resource to the RP, provided the validation is successful.

OAuth is currently undergoing a second version draft, since the first version of OAuth [101] had performance problems when deployed on a large scale and usability issues both for end-users and OAuth protocol implementers [100]. OAuth solves a different problem than CYOA. CYOA is used to authenticate people while OAuth requests authorisation from users to provide third-parties with access to users' digital assets.

7.3 User Experience

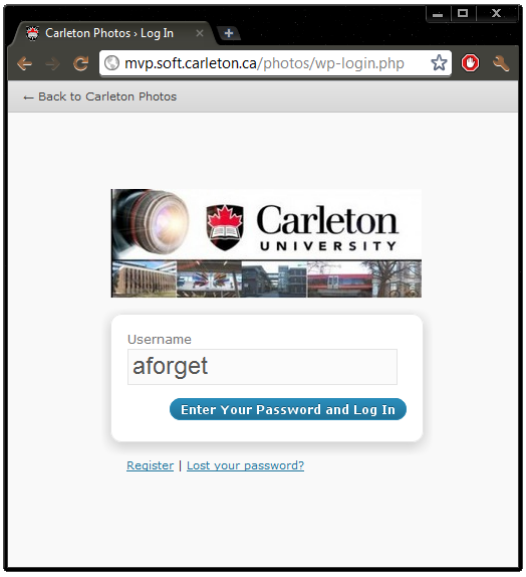
Before describing the CYOA design and architecture, we will first illustrate CYOA from the user's perspective. What follows is the simplest example of user registration or login to a website with a web-based implementation of CYOA (such as our prototype implementation discussed in Section 7.5). The user begins by navigating to the website's login page and authenticates as follows:

1. The user first enters their username into a field (or provides their user ID in some manner) and submits it to the website (Figure 7.2a).

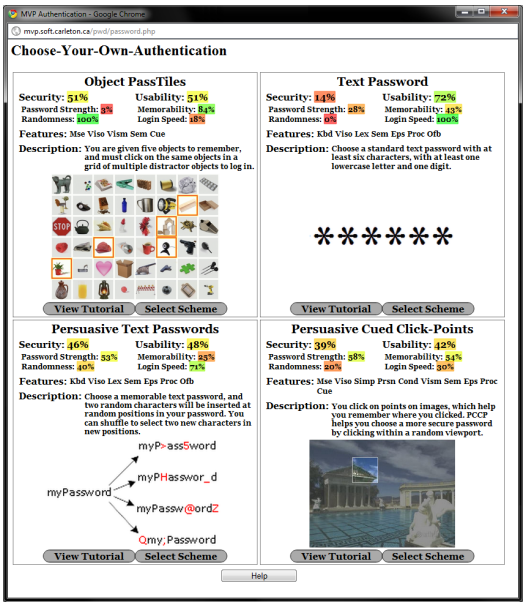
2. The user is then shown a scheme selection form presenting a variety of available authentication schemes (Figure 7.2b). The user may examine the ratings, descriptions, and tutorials of the available schemes through the scheme selection form. The user chooses their authentication scheme amongst the selections.

3. The selected scheme is displayed to the user, who enters their password (Figure 7.2c). If creating a new password, they would be asked to enter the same password a second time for confirmation.

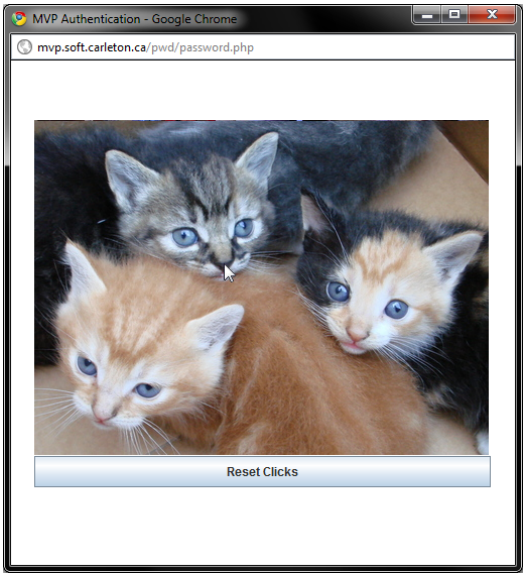
4. When logging in, if the selected scheme and entered password respectively match the scheme chosen and password created during the account's registration, then the user is granted access to the account (Figure 7.2d).



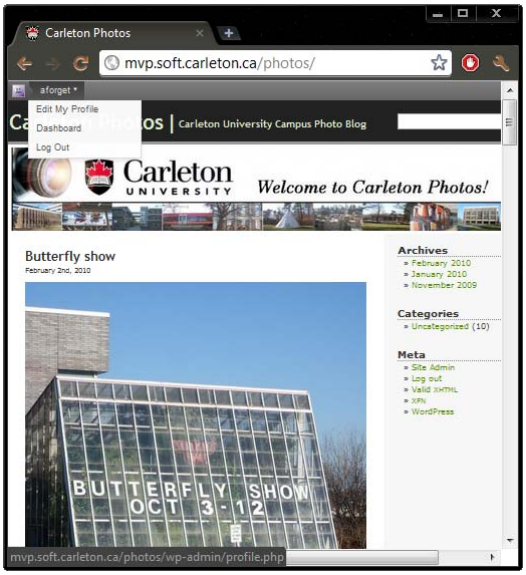
(a) The user enters their username



(b) The user selects their authentication scheme



(c) The user enters their password (with Persuasive Cued Click-Points [46] in this example)



(d) If the selected scheme and the entered password are both correct, the user is granted access

Figure 7.2: Example of the user experience when logging into a system using CYOA.

Clients only need a typical web browser to use such a web-based implementation of CYOA. Some authentication schemes may use multimedia web technologies such as Adobe Flash, Microsoft Silverlight, or Java, which may require an installation.

However, we believe most schemes could be implemented with modern JavaScript, which only require an up-to-date browser (and most modern browsers automatically update themselves).

7.4 Design

We will now describe CYOA's architectural design. We assume that all communications between parties are secure (using SSL or other encryption technology). In all cases, there are three main entities involved in the CYOA authentication protocol:

1. ***Client***: The machine with which the user remotely authenticates to the application. The client's user wishes to log in to the application.

2. ***Application server***: The server hosting the application resources and services which require authentication before access is granted. For CYOA, the application server is responsible for storing and verifying users' passwords. Most other authentication-related tasks are delegated to the CYOA module.

3. ***CYOA module***: The component that stores and manages the data and code necessary for the CYOA architecture to function. This component can either be on its own server or part of the application server. The CYOA module contains the supported authentication schemes, related data, and a database table that stores information keyed to the username and scheme. Users' passwords are never stored in the CYOA module in any form. We discuss the CYOA module in greater detail in Section 7.4.1.

7.4.1 CYOA Module

This section will describe and illustrate our concept of the CYOA module's internal structure. However, not all CYOA modules must be designed in this specific manner, nor do they need to be object-oriented (as are our UML class diagrams in this section). In fact, a different architecture may be preferable, depending on the target implementation environment. Our goal here is to first provide the reader with a general high-level concept of how the core CYOA module's internal architecture may

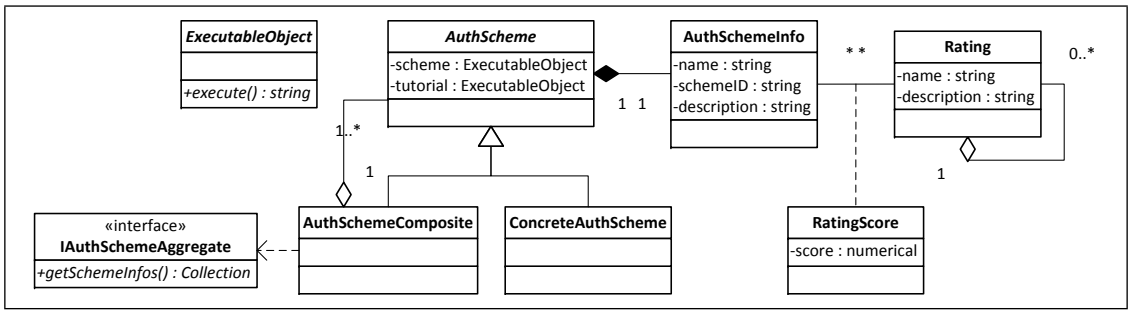


Figure 7.3: UML 2.4.1 class diagram of the CYOA module's general architecture.

be visualised with UML, and subsequently provide more specific examples of possible CYOA implementations, based on the aforementioned internal architecture.

Figure 7.3 shows a UML 2.4.1 class diagram illustrating the general architecture of the classes in the static structure of the CYOA module. `AuthScheme` is an abstract class representing the concept of an authentication scheme. An `AuthScheme` object can be thought of as a wrapper for the scheme itself, which enables CYOA to incorporate and manipulate schemes in a consistent manner. `AuthScheme` has the following attributes (and respective `get` methods):

- *`scheme`* is an `ExecutableObject` representing the object the user interacts with to generate an encoded password. Ideally, the implementation of `ExecutableObjects` should be self-contained and as language-independent as possible. That is, `ExecutableObjects` should be encapsulated such that the CYOA module and the authentication scheme execution environment (e.g. a web site in a web browser or an application) on the client should not need to know which scheme is running (i.e. which scheme the user chose). An `ExecutableObject` must only be able to `execute()`, which runs the object. This method is called by the application when the user's chosen scheme is loaded and it is time for the user to login. This method returns a string, which for a CYOA scheme is the encoded password, the result of a user entering their password with the chosen scheme.

- *`tutorial`* is an `ExecutableObject` representing the object containing the tutorial the user interacts with to learn how to use the scheme. This

`ExecutableObject`'s `execute()` method will run the tutorial. Since no password entry takes place with the tutorial, the `tutorial`'s `execute()` return value may be ignored by CYOA.

Each `AuthScheme` object must also contain an `AuthSchemeInfo` object, which itself contains all descriptive data relating to the authentication scheme. This descriptive data is stored in a separate class since the CYOA module must send this set of attributes to the user separately (*step 2* in Section 7.3) from the scheme, tutorial, or other attributes later in the CYOA registration or login process. `AuthSchemeInfo` stores this information in the following attributes:

- *`name`* is a string storing the name of the authentication scheme.
- *`schemeID`* is a string that uniquely identifies the authentication scheme.
- *`description`* is a string describing the authentication scheme.

The `AuthSchemeInfo` class also stores a `Collection` of `RatingScores`. A `RatingScore` represents the score given to a scheme for a particular `Rating`. A `Rating` represents a metric by which CYOA administrators measure schemes' effectiveness, usually in terms of its usability or security. `Ratings` may also contain other `Ratings` as sub-ratings, in a tree structure. For example, a `Rating` named ``‘‘Usability’’`` may contain `Ratings` named ``‘‘Memorability’’`` and ``‘‘Login Speed’’``. Section 7.5.1 provides more examples of ratings. The `name`, `description`, and `RatingScores` stored by `AuthSchemeInfo` objects are provided to users to assist in selecting an authentication scheme best suited to their preferences and environment.

Authentication schemes that are to be added to a CYOA module must extend the `AuthScheme` abstract class. This ensures that all CYOA authentication schemes, regardless of implementation details, all have a minimum set of available information (attributes) and set of behaviours (public methods). The class `ConcreteAuthScheme` represents such an implemented authentication scheme. `ConcreteAuthSchemes`' constructors may take parameters which modify the scheme's behaviour. For example, a password scheme constructor may have a parameter to set a minimum password length. An administrator could instantiate several objects from the same

ConcreteAuthScheme class, each with different parameters. This would give users the freedom to select not only from several authentication schemes, but also many configurations of a particular scheme. This is especially useful if users prefer a given scheme, but wish to select stronger or easier-to-use configurations, depending on the context and threat model.

The AuthSchemeComposite class stores a Collection of AuthSchemes, which may contain ConcreteAuthSchemes and other AuthSchemeComposites. This relationship between AuthSchemes, ConcreteAuthSchemes, and AuthSchemeComposites follows the composite design pattern [88], whereby a set of objects of a particular type can be treated as a single object of that same type. An AuthSchemeComposite object can be thought of as an authentication scheme whose resulting encoded password is the concatenation of the user's selection of one of the AuthSchemeComposite's supported authentication schemes, and the encoded password result of said selected scheme. AuthSchemeComposites must implement the getSchemeInfos() method from the IAuthSchemeAggregate interface. This method returns a Collection of AuthSchemeInfo objects of the AuthSchemes contained in the AuthSchemeComposite. The getSchemeInfos() method is typically used to provide information about the available authentication schemes to the user, when a scheme must be chosen.

Figure 7.4 shows an example implementation of classes in a CYOA module. The AuthScheme, AuthSchemeInfo, ExecutableObject, IAuthSchemeAggregate, Rating, and RatingScore classes remain the same as before. The CYOASchemeSelector class represents an AuthSchemeComposite from Figure 7.3. The CYOA module will instantiate a CYOASchemeSelector object to hold and manage instantiated authentication schemes as AuthScheme objects.

The remaining classes in Figure 7.4 are examples of ConcreteAuthSchemes. PassPoints, Cued Click-Points, Persuasive Cued Click-Points, and PassTiles are representations of cued-recall graphical password schemes discussed in Section 2.3. Persuasive Text Passwords, Cued Gaze-Points, and Persuasive Cued Text Passwords represent the schemes presented in this thesis in Chapters 4 and 5. Finally, the Text Passwords class is an AuthScheme implementation of the plaintext

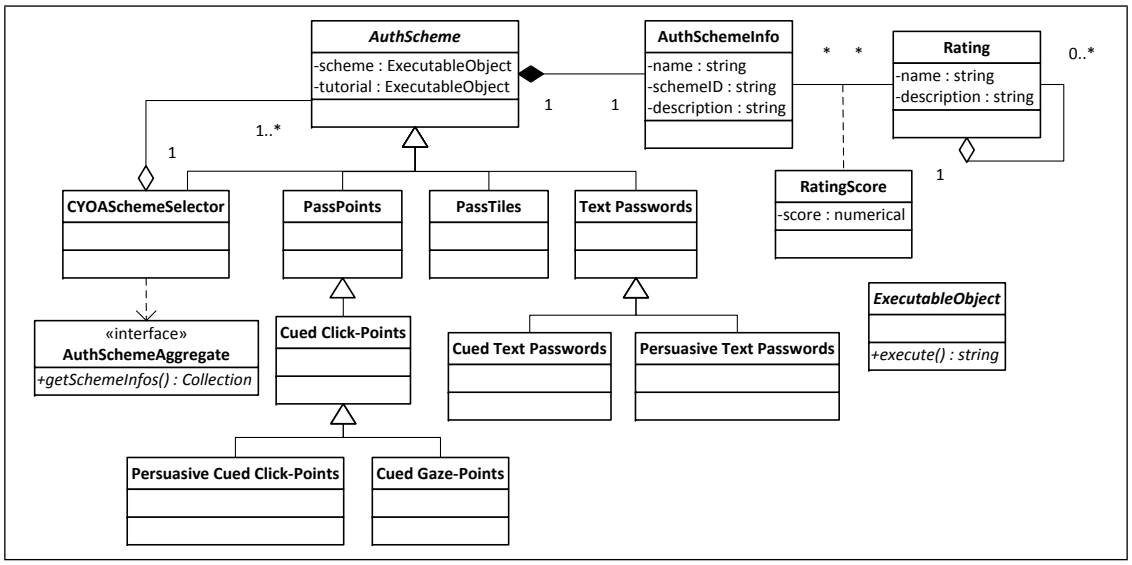


Figure 7.4: UML 2.4.1 class diagram of an example implementation of the CYOA module.

password system most current authentication systems employ. As previously mentioned, these classes' constructors may take parameters to alter their behaviour. For example, the `PassPoints` scheme may support different image sizes or numbers of click-points in a password. Since the Cued Click-Points scheme is a derivative of PassPoints (Section 2.3), the `Cued Click-Points` class could inherit and improve upon the `PassPoints` class by extending it. The same relationship occurs between `Cued Click-Points` and its successors, `Persuasive Cued Click-Points` and `Cued Gaze-Points`. This of course assumes that these authentication schemes are developed in a proper object-oriented manner, which is not strictly necessary for a scheme to be supported by CYOA. For example, an authentication scheme developer could simply copy the `PassPoints` class, rename the copy to `Cued Click-Points`, and add any new code as required by the scheme, all without any explicit inheritance. This approach forfeits the benefits of an object-oriented design, but such a discussion is outside the scope of this thesis. However schemes are implemented, so long as they inherit from the `AuthScheme` abstract class, thereby supporting its attributes and implementing its methods, then the CYOA module can supported the scheme.

We will now discuss how such a CYOA module fits into the larger architecture of an application using CYOA as its authentication service.

7.4.2 Two-Party Architecture

In its simplest form, the CYOA module can simply be part of the application server as any typical authentication scheme. For this implementation, the CYOA module may be on the same physical machine as the application system or on its own server within the application's domain. However, for the purpose of clarity, our discussion will treat the application and CYOA module as separate systems in the same domain.

We will illustrate how CYOA would work within the current Internet infrastructure, where the application is a typical web-based system of any implementation and offers any kind of services or resources for authenticated clients. We assume all communications are secured with SSL, TLS, or better. In this scenario, a typical CYOA user authentication over the Internet would begin with the user navigating to the application website's login page and proceeding as illustrated in Figure 7.5:

1. The user enters their username into the application's login form and submits it to the application server.

2. The application server requests the list of available authentication schemes from the CYOA module.

3. The CYOA module returns an XML file listing the available schemes and optional scheme-related data (e.g descriptions, ratings) to the application server.

4. The application server converts the XML list of schemes and relevant data into a scheme selection web form and returns it to the client.

5. On the presented webpage, the user selects the authentication scheme associated with their account and submits the form to the application server, thereby sending a request for the chosen scheme.

6. The application server requests the chosen scheme from the CYOA module.

7. The CYOA module returns the chosen scheme to the application server.

8. The application server adds the chosen scheme to a web form for authentication and returns it to the client.

9. The user enters their password with the selected authentication scheme, which converts it into a string and submits it to the application server. If registering a new password, the user may need to enter and submit their password a second time for confirmation. If the chosen scheme requires communication with the CYOA server during password creation or entry, such communication would happen between this step and the previous one.

10. The application server sends the password string to the CYOA module.

11. The CYOA module encodes the password string with the chosen scheme's password encoding function, and returns the encoded password string to the application server.

12. If registering a new password, the application server stores the encoded password string keyed to the username. If logging in, the application server compares the encoded password string with the one stored for the corresponding username.

13. If the user is registering a new password, the user is sent confirmation of the successful password registration. If logging in and the compared passwords match, the user is granted access. Otherwise, the user is denied access.

When logging in, the user might select the wrong scheme, a scheme other than the one chosen at registration to create their password. Since different schemes encode passwords differently, and the password entry methods can be completely different, it is highly improbable that an encoded password derived from one scheme would be the exact same as another encoded password from a different scheme (e.g. see hashing in Section 2.2). However, if a user selects the incorrect scheme at login, they should be allowed to attempt to login with the chosen scheme, even though the login attempt will fail. The user should not be alerted that their choice of scheme was incorrect, because doing so would allow imposters to trivially determine the user's correct scheme, thereby negating the password strength benefit of keeping it secret (Section 7.4.2). We discuss the trade-off in the usability and security benefits of automatically providing users with the correct scheme at login in Section 7.4.2.

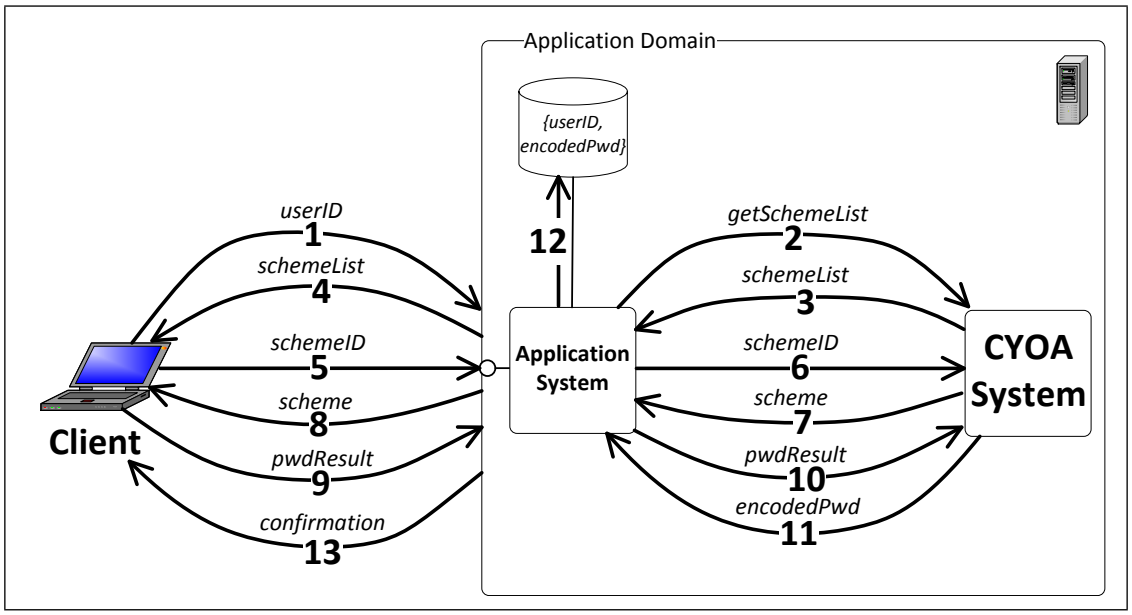


Figure 7.5: Basic two-party CYOA architecture where the application server and CYOA module are managed by the same organisation.

Some schemes may provide users with a personalised login, such as cues or some form of reproducible state which would have been established during registration. For example, a graphical password scheme (Section 2.3) may need to store the names of images that should be shown to a specific user. At login, if the user selects such a scheme incorrectly, the CYOA module would find no state data for the scheme to use. If this situation is not handled properly, it could be easy for an imposter to determine which schemes a user did not select, since they may default to some unusual behaviour (such as providing different states for the same user at each login, or providing the same default state for any unrecognised user). To solve this problem, CYOA requires that schemes provide a function that effectively generates a password for an unknown user, and stores its state. Thus, whenever the CYOA module attempts to retrieve the state data for a given user-scheme pair, if no such data exists, the scheme's password generation function will create and store the required state, and the CYOA module will finally pass the new state data to the scheme for login. Should the user (or an imposter) later select the incorrect scheme again, the previously-generated state will be given to the scheme, providing for a consistent experience which an imposter should not be able to differentiate from the behaviour of the user's correct scheme.

Technical Benefits

There are a number of technical benefits to CYOA:

Easy adoption for existing systems. CYOA authentication schemes return encoded passwords as strings. Since modern password systems already store passwords as strings, integrating a CYOA module into an existing password system requires only minor modifications. Specifically, the CYOA module would replace the existing authentication scheme (most likely the text password system) module. None of the supporting technology (e.g. database or other infrastructure) need to be modified, since CYOA uses the same technology in the same way as existing text password systems. Some user interface modifications may be needed to accommodate scheme selection, but no additional software need be installed on the client. Although Figure 7.5 depicts the CYOA module as an external module separate from the application, CYOA could just as easily be implemented into the application itself. Two simple examples of such an integrated implementation include as a WordPress plug-in or as the built-in authentication system used by any web application, including an OpenID identity provider.

Supports most knowledge-based authentication (KBA) schemes. Researchers designing novel KBA schemes often strive to require minimal changes to the host application. Thus, many KBA schemes are likely to store some encoded derivative string of the user's password to use when verifying future login attempts. All such schemes are supported by CYOA, since the application server stores all encoded password results as a string and always performs a bitwise comparison of stored and entered encoded passwords for verification. However, this precludes CYOA from supporting authentication mechanisms that require a different method of password verification. One example of such schemes include challenge-response (or obfuscation) schemes, which require that verification vary depending on the challenge. Another example is biometrics, which compare an input biometric to a stored template using heuristics. A simple bitwise comparison is typically impossible for such schemes. However, the Caveats section below discusses how this limitation can be overcome.

Modular architecture. The CYOA module itself is designed modularly, so system administrators can "plug-in" novel schemes without any disruption to users.

Authentication schemes can be implemented in any language, so long as the executable code can be run by clients. In the modern web infrastructure, this means authentication schemes can be implemented in any format that can be run in standard web browsers. New authentication modules can easily be created and added to the system, and administrators can choose which are suitable for their particular environment. Administrators can also easily remove schemes that have a recently-discovered vulnerability or no longer fit their organisation's security policy.

Expert-certified authentication security. We envision that a network of third-party authentication experts and certification authorities could be organised. The function of this network would be analogous to the academic peer-review or SSL certification processes. Researchers and developers may independently implement novel authentication schemes and submit them to third-party authentication experts who independently analyse, review, certify, and serve these schemes for application administrators to download and plug-in to their CYOA module. Application administrators could subscribe to one or more such third-party CYOA scheme certification authorities and thus be relieved from the burden of building or auditing the multitude of possible schemes. However, there is no obligation for application administrators to subscribe to any such scheme-certifying third-party. Administrators are free to implement, add, or remove any authentication scheme from their CYOA module, regardless of whether or not they also subscribe to any CYOA scheme certifiers. Conversely, in Section 7.4.3, we extend the potential role of a trusted third-party. Users wishing to register with or sign in to the application could be redirected to the trusted third-party, who serves the authentication scheme directly to the client, and returns the authentication result to the application for storage or verification.

Resistance against password guessing attacks. Any system using CYOA is more resistant to password guessing attacks than any single authentication scheme, because the attacker does not know which scheme a given user has chosen. In effect, the more authentication schemes that are in use, the larger the whole system's theoretical password space (TPS), and thus the larger the effective password space (Section 2.2). However, this growth in password spaces is additive, and thus only has a small effect on the size of the TPS. For example, assume that all password schemes

have a TPS of 20 $bits = log_2(2^{20})$. A CYOA module supporting 8 such schemes would have a TPS of $log_2(8 \times 2^{20}) = log_2(2^3 \times 2^{20}) = log_2(2^{23}) = 23\ bits$. Admittedly, a large number of schemes only makes a small difference. Nonetheless, it is still an improvement. The additional effort required to build dictionaries of candidate passwords for many different authentication schemes may deter potential attackers.

Resistance against phishing. CYOA increases the cost and complexity of launching phishing attacks. A credible phishing site would need to replicate the target application's CYOA module and the authentication schemes it supports (which may not be publicly available). This would be particularly difficult for schemes that provide user-dependent cues, like Persuasive Cued Click-Points (Section 2.3) [46]. Without the correct cues, users would most likely be *unable* to enter their actual password, assuming they were still fooled in spite of the incorrect cue. The simplest way to overcome these complications may be to also perform a man-in-the-middle (MitM) attack, whereby all the user's input on the phishing site are passed to the legitimate site. Another option in building a phishing attack would be to compromise the legitimate application's CYOA module to obtain the schemes and user-specific pre-authentication information, and use the data for their own phishing site's CYOA module. However, in the two-party architecture, the CYOA module is no more vulnerable than the application server, since they are both administrated in the same organisation's domain, and the application server is directly accessible to the outside, while the CYOA module is not. Should an attacker breach the application's domain defences, there is little point in attacking the CYOA module when the application server (which stores all the assets of value) is already compromised.

Usability Benefits

In addition to technical benefits, there are also important usability benefits to CYOA:

Accommodates user preference. CYOA allows users to select whichever authentication scheme they wish. Users with better visual memory can choose graphical passwords instead of text, or vice versa. Users may also select schemes that provide greater password strength for their accounts of higher personal value or risk (such as bank accounts). Conversely, users may opt for more memorable (but possibly less

secure) schemes for low-value accounts they seldom access. Schemes requiring particular hardware (such as eye tracking for gaze-based schemes (Chapter 5) or smart card readers for two-factor authentication) can be selected when the hardware is available.

Educates about authentication concerns. The CYOA scheme selection interface should provide a description and various ratings for each scheme to help users make their choice. The ratings may be available at multiple levels of granularity. Examples of such ratings include high-level measures of overall security and usability, a list of features supported by the authentication scheme (Section 3), or more specific measures, such as those discussed by Bonneau et al. [24, 25]. Plain-language explanations of the ratings should be available for users to understand their meaning.

Supports accessibility. Current text password systems may pose barriers to people with special needs. For example, anyone who has difficulties with textual tasks, including comprehension and articulation, may have trouble using text passwords. Fink [73] found that academically- and professionally-accomplished dyslexic people had significantly more difficulty spelling non-words than non-dyslexics. Fink also reported that the dyslexic participants' reading ability was very context-dependent. These results suggest that even very capable dyslexic people may have difficulty with text passwords, which should be more cryptic than non-words used in Fink's study.

Another group of people who may have difficulty typing text passwords are users with physical motor-control impairments. These users may instead use speech recognition software (such as Dragon NaturallySpeaking [144]) to verbally articulate their passwords. However, Rebman et al. [166] found that users made significantly more errors when typing with speech recognition software compared to a keyboard. The author also found that when focusing on accuracy, typing with a keyboard was significantly faster than using speech recognition software, despite the fact that people can generally speak more quickly than type. These results suggest that speech recognition users require significantly more time to enter text passwords then typists, since text passwords should contain uncommon characters to be secure, and must be 100% accurate for a successful login. Using speech recognition software for password entry also makes users more vulnerable to observation attacks, since it may be easier to overhear someone speak their password than watch it being typed.

Clearly there is a large gap between modern authentication systems and users with special needs. The World Wide Web Consortium sponsors a Web Accessibility Initiative (WAI) [221] that explores, develops, and publishes guidelines and techniques for developing accessible websites and web-related technologies. However, we could find no guidance on providing an accessible method of authentication. The Americans with Disabilities Act [205] and the Accessibility for Ontarians with Disabilities Act [132] require employers and service providers to provide reasonable accommodations for people with disabilities. These regulations are likely to be amended to explicitly include digital content and services, given the growing dependence and popularity of digital resources. While digital accessibility legislation and awareness remains generally absent, the only way for users to authenticate to most computer systems and websites remains with a text password. These systems typically offer no accommodation for challenged users who may have above-average difficulties in creating, remembering, or entering text passwords.

Providing an authentication method that is accessible to a newly-registering user, whose abilities are unknown to the system, is a difficult problem [69]. Renaud [169] demonstrates that there is no single authentication solution that can accommodate all users. This highlights the need for CYOA, since supporting multiple authentication methods is currently the only solution to the accessible authentication problem.

CYOA modules can support authentication schemes more suitable for users with special needs. CYOA offers the possibility to easily accommodate users with accessibility requirements, which have thus far been largely overlooked in authentication research. Newly-developed accessible authentication schemes can easily be added to a CYOA module. Systems that adopt CYOA can easily comply with evolving accessibility requirements and legislation without any disruption to existing users who do not need to use a new accessible authentication scheme. CYOA would benefit all users that may have challenges with text passwords. Users that have difficulties with text could select a graphical password scheme. Users with visual impairments could choose some form of audio-based authentication scheme. Users with other impairments can similarly select authentication schemes better suited to their abilities.

Caveats

There are some minor variations that CYOA administrators may wish to consider when integrating CYOA in their applications. One such variation is whether or not to require users to select their authentication scheme before logging in. The CYOA module could simply remember each user's scheme and present them with it immediately, instead of requiring the user to select the correct scheme. In this case, steps 2 to 5 in Figure 7.5 would be skipped at login. However, this would reveal the correct scheme to an imposter (pretending to login as the user by entering their username), thereby simplifying a password guessing attack to a single scheme. Both variants are vulnerable to observation attacks, whereby an adversary watches or records the user logging in. Whether or not the usability advantage of reducing users' memory load is worth the decrease in security provided by CYOA is a decision administrators should make appropriate to their system's threat model. The repercussions of either choice should be explored in future work (Section 7.6).

As previously mentioned, CYOA may not be able to support challenge-response (or obfuscation) schemes or many biometrics, because they would require a special verification function. CYOA does not support these authentication methods by default since one of our design goals is to minimise the modifications required to the application server's existing authentication process. However, CYOA could be adapted to accommodate such schemes by making one of two additions, each with their respective disadvantages:

- Schemes could be required to include a modular comparison method that the CYOA module can provide to the application server on request. However, this would require the application server to execute code from an external source (being from wherever the password verification code originated), which may be undesirable in some threat models. The verification function could be run in a sandbox or virtual machine, but this may be too cumbersome to implement and time-consuming to execute.

- CYOA schemes could post their verification algorithms in a public location, whereby application administrators could implement the algorithms themselves.

However, this puts a significant burden on the application administrators, particularly if CYOA schemes are added, updated, or removed frequently.

Some existing text password systems enforce password restriction policies that limit the length or set of characters allowed in passwords (Section 2.2.1). Should these systems wish to adopt CYOA, there may be conflicts between the encoded passwords generated by CYOA schemes and the limits imposed by the existing restriction policy. Administrators have two options to resolve this conflict. Bypassing or removing the password restriction policy is likely to be the easiest and simplest solution. In most text password system implementations, it should not be difficult to remove or bypass the password restrictions, since the text password system is already being replaced by CYOA. However, some legacy password storage systems may not be able to support arbitrary bit-strings. In this case, the system could convert the scheme-encoded passwords into correctly-formatted bit-strings before storing or validating the passwords. This could be done with a one-way hashing algorithm that outputs the hash in the desired format [172].

7.4.3 Three-party architecture

The CYOA module could also be managed by a third-party authority responsible for ensuring the security and integrity of the offered authentication schemes. Applications can then subscribe to such an authority and users wishing to be authenticated are referred to this authority. After the user selects and enters their password with their chosen scheme, the authority returns the encoded password to the subscribing application for storage or verification. Thus, the user performs the password-entry process with a centralised, reputable, and trusted authority, but the hashed password itself is still stored and verified by the application.

Applications wishing to refer their users to a third-party CYOA service must first subscribe to it. The third-party CYOA authority will require from the application a URL of where the client should be directed (and the encoded password sent) after entering a password. The CYOA authority will pair this URL with a unique application ID and store the pair. Finally, the authority will send the application their application ID and the URL to which users should be directed to login to the application.

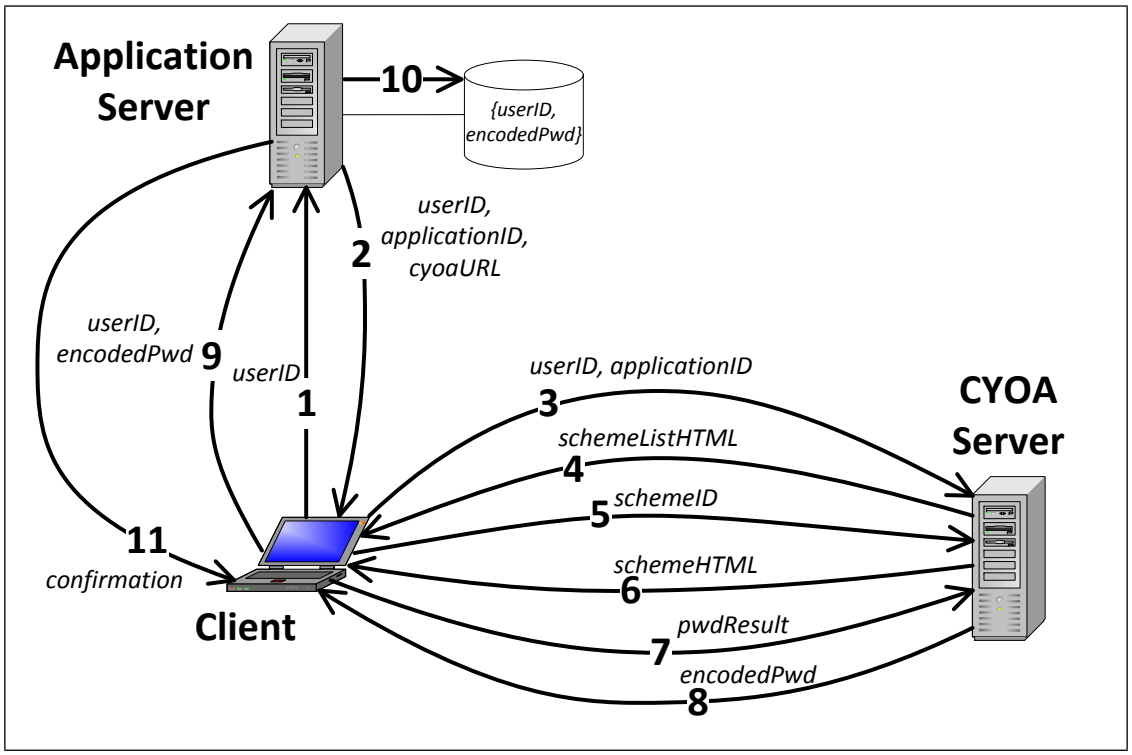


Figure 7.6: CYOA architecture involving a trusted third-party which manages the CYOA module on their own server

Either party may also request other ancillary information not specifically relevant to the CYOA protocol. As with two-party CYOA, we assume all communications between all parties is secured (presumably with some encryption technology).

We will illustrate how three-party CYOA would work within the current Internet infrastructure. As with two-party CYOA, the application is assumed to be any typical web-based system of any implementation and offers any kind of services or resources to authenticated clients. Figure 7.6 illustrates the sequence of events that occur throughout the CYOA protocol, beginning when a user navigates to the application's login webpage:

1. The user enters their account's username into the application's login form and submits it to the application server.

2. The application instructs the client to open a new window and navigate to the URL of the third-party CYOA server, sending along the application's ID and the user's username.

3. The client opens a new window and sends a request to the CYOA server (located at the URL previously provided by the application), passing along the username and the ID of the application to which the user is authenticating.

4. The CYOA server returns to the client a web form containing a list of available authentication schemes, with any optional scheme-related data (such as brief descriptions, security and usability ratings, and so on).

5. On the presented web form, the user selects their authentication scheme associated with their account and submits the form, requesting the chosen scheme from the CYOA server.

6. The CYOA server returns a web form containing the chosen scheme to the client.

7. The user enters their password with the selected authentication scheme, which converts it into a string and submits it to the CYOA server. If registering a new password, the user may need to enter and submit their password a second time for confirmation. If the chosen scheme requires communication with the CYOA server during password creation or entry, such communication would happen between this step and the previous one.

8. The CYOA server encodes the password string with the chosen scheme's password encoding function, returns the encoded password string to the client, and informs the client the password entry is complete.

9. The client requests to be granted access to the account by providing the user's username and encoded password string to the application server. The new window on the client is closed, since the password entry is complete.

10. If logging in, the application server compares the encoded password string with the one stored for the corresponding username. If registering a new password, the application server simply stores the encoded password string keyed to the username.

11. If logging in and the passwords compared in the previous step match, the user is granted access. If logging in and the passwords do not match, the user is denied access. If the user is registering a new password, the user is sent confirmation of the successful password registration.

Benefits

The three-party CYOA architecture contains all of the benefits of the two-party architecture, as well as the following additional advantages:

Delegates authentication security to experts. Research has shown that modern website application administrators continue to have difficulty providing secure and usable authentication [70, 75, 87]. We envision the third-party CYOA service being administrated by trusted and credible authentication security experts, who would be constantly independently reviewing available schemes, offering only those they deem adequately secure and usable, and discarding or removing flawed or malicious schemes. A subscription to a third-party CYOA service relieves the burden of maintaining a secure and usable authentication system from the application developers and administrators, who are probably not security experts themselves.

Decentralised password storage. The third-party CYOA service centralises the password entry process. However, the applications remain responsible for actually storing and verifying users' passwords, while the third-party CYOA server does *not* store the passwords. This prevents attacks on the third-party CYOA server to obtain users' passwords for all subscribing applications, as the authority does not store users' passwords in any form.

Caveats

A challenge of this architecture is determining where the password encoding should be performed. It may be inadvisable for the client to perform the encoding since this may reveal information to an adversary impersonating the user. One example is the Centred Discretization [45] algorithm used in some graphical password schemes [46] to encode click-points into a string. There is no evidence that this encoding algorithm reveals information about users' passwords, but just like a system's choice of

password hashing algorithm, there is no reason to purposefully make it accessible to adversaries. Thus, if possible, we may wish to avoid performing the encoding on the client. Instead, the application server could perform the encoding, but it would face the same challenges as though supporting custom password verification methods, which are discussed in Section 7.4.2. The only remaining alternative is for the CYOA server to perform the password encoding (as illustrated in Figure 7.6), but this also has a weakness: The subscribing applications and users must trust their passwords to the third-party CYOA server. However, users of network-based password managers (like LastPass [131]) and SSO solutions (like OpenID [148]) must also trust a third-party with their passwords. Thus, the three-party CYOA architecture is no more vulnerable than existing authentication solutions that also involve a third-party.

7.5 Prototype Implementation

We have implemented a CYOA prototype in our authentication experimentation platform, Multiple Versatile Passwords (MVP) [38]. CYOA is implemented as an MVP authentication scheme. The schemes available through CYOA are also implemented as MVP schemes, which have been tested in other experiments [46, 105, 190, 191]. This section will describe the MVP CYOA scheme selection user interface and describe UML 2.4.1 activity diagrams for registration and login with MVP CYOA.

7.5.1 Scheme Selection Interface

Our CYOA prototype (Figure 7.7) would provide participants with a variety of password schemes well-known to the usable authentication research community, as well as schemes we believe offer unique usability or security advantages, based on experience and research [19]. Example authentication schemes that we have implemented and would be available through our MVP CYOA prototype include:

- ***Text Passwords.*** Including text passwords would determine if users will select standard text passwords over other authentication schemes. We believe this distinct possibility would entail serious repercussions to be considered (Section 7.6.1) should users be reluctant to take advantage of novel schemes.

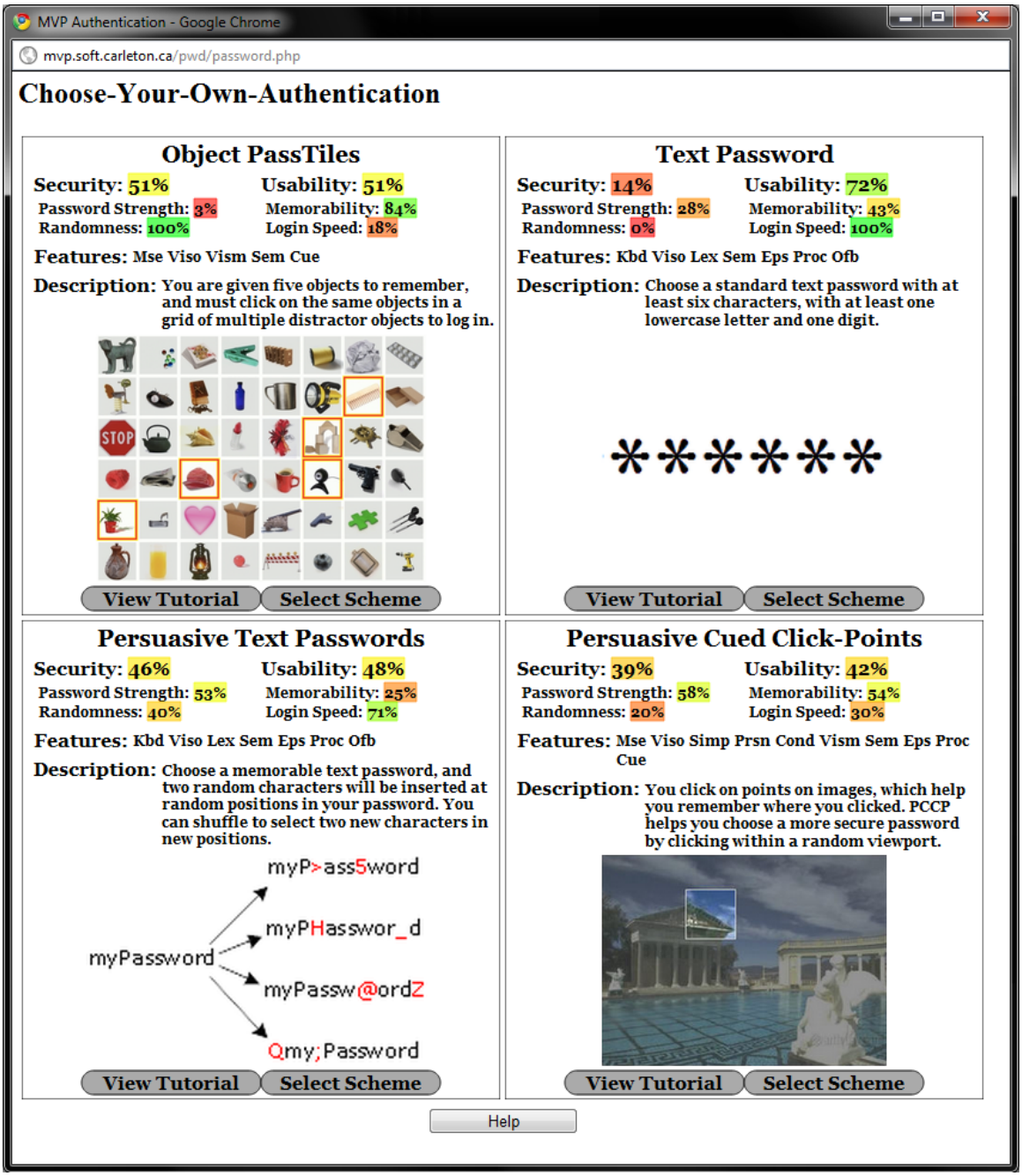


Figure 7.7: Current version of the MVP CYOA scheme selection interface.

- ***Persuasive Text Passwords (PTP, Chapter 4).*** PTP offers a more secure text-based password scheme than standard text passwords (Chapter 4).

- ***Object PassTiles (OPT) [188].*** This scheme is similar to the well-known commercial recognition-based graphical password scheme PassFaces [155]. The

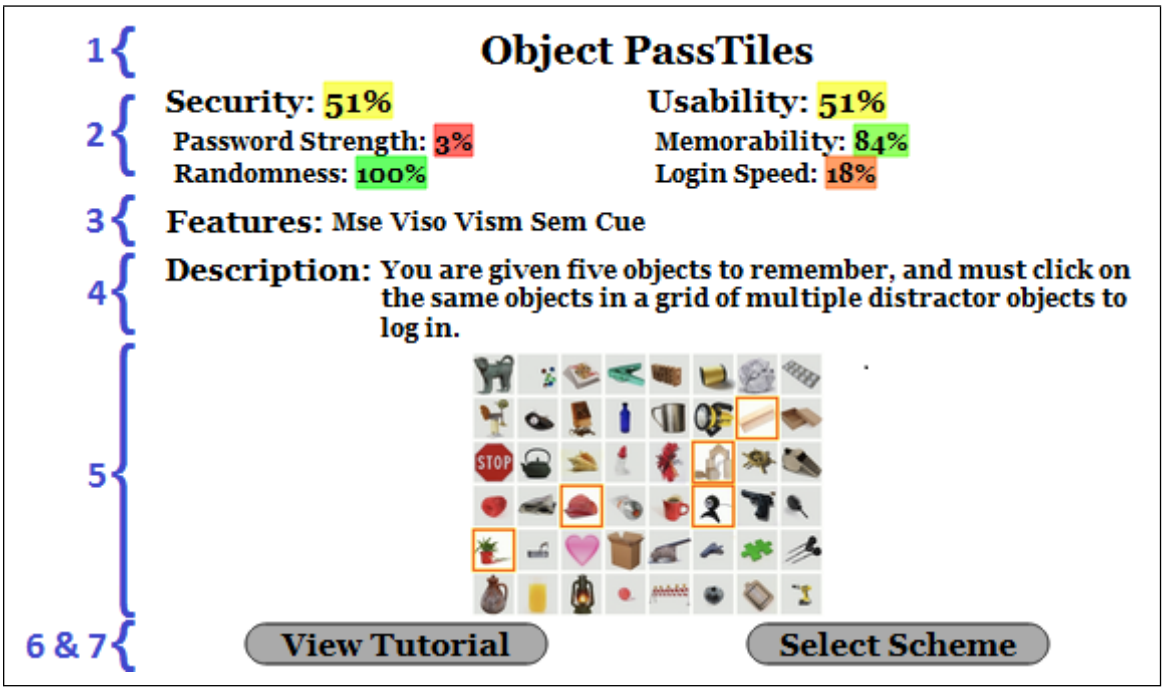


Figure 7.8: Object PassTiles in the CYOA scheme selection interface (with annotations).

main difference between the two is that PassFaces uses images of faces while OPT uses object images. We believe OPT may be more usable than PassFaces because Hlywa et al. [105] found that users have a better memory for distinct objects over faces.

- ***Persuasive Cued Click-Points (PCCP) [46].*** Studies by Chiasson et al. [46] and comparisons with other graphical password schemes [19] have suggested that PCCP may be one of the more usable and secure graphical password schemes currently published.

It would be unfair to require users to select one of several unfamiliar password schemes without any information about them. To help users choose, CYOA provides the following information about each scheme (Figure 7.8):

1. The name of the scheme.

2. Security and usability ratings (described below).

3. A list of supported features (from the User-Centred Authentication Feature Framework in Chapter 3).

4. A brief description of the scheme.

5. A screenshot or illustration of the scheme.

6. The `View Tutorial` button displays the scheme's tutorial in a new window.

7. The `Select Scheme` button selects the authentication scheme for password registration or login.

The security and usability ratings are based on published authentication research [19, 24,74,75,212]. A brief description of each rating and feature would be displayed when users hover their mouse cursor over them. All rating scores are shown as percentages between 0% (worst) and 100% (best). The specific ratings would include:

- ***Security***: Overall security would be scored as an average of the following security ratings.

  - ***Password Strength.*** We follow Biddle et al.'s [19] theoretical password space (TPS) ratings for graphical passwords, whereby schemes' TPS falling within a particular range are awarded a proportion of the maximum possible score for this rating. Based on their password strength rating, Equation 7.1 illustrates the formula to calculate this score. Schemes with a theoretical password space (TPS) of 20 bits receive a score a 0%, since this is the bare minimum TPS schemes should support [75]. Thus, the formula first subtracts 20 from the TPS. The result is then multiplied by 2.5 to award 100% of the maximum possible score to 60-bit schemes. Clearly, TPS values of under 20 bits or over 60 bits will fall outside of the 0% to 100% range. Thus, such schemes would be respectively scored 0% or 100%, since schemes with a TPS below 20 bits are considered insecure, and TPS values over 60 bits are considered to provide little additional practical security [19].

$$S_{MaxPwdStr} = (TPS - 20) \times 2.5 \tag{7.1}$$

  - ***Randomness.*** Schemes that provide assistance in choosing more secure passwords are scored approximately proportional to the increase in security added. Schemes that assign completely randomly-generated passwords are awarded a full score.

- ***Usability***: Overall usability would be scored as an average of the following usability ratings.

  - ***Memorability.*** Schemes are scored based on available research quantifying password recall at least one day after creating the password. This score is higher the fewer password entry errors they committed, and the greater the time between the password's creation and subsequently measured login attempt.

  - ***Login Speed.*** The faster users can enter their password, the higher the scheme's login speed score.

7.5.2 Tutorials

We have developed a tutorial for each of the available schemes. The tutorials are either in the text or hypertext formats described in the Authentication Scheme Learnability study (Chapter 6), since they seemed to help users learn the novel scheme the most efficiently and as effectively as the other tutorials. Users may view these tutorials before selecting an authentication scheme by pressing the `View Tutorial` button for the scheme of interest. This allows users to make their choice based on the tutorial material, should they wish to explore a scheme further before making a selection. We have also designed a tutorial for CYOA itself, which would be shown immediately before registering a password. The CYOA tutorial describes the purpose and function of CYOA, as well as highlights the meaning of the various interface items.

7.5.3 Activity Diagrams

To illustrate how CYOA works within MVP, Figures 7.9 and 7.10 illustrate the registration and login process in UML 2.4.1 activity diagrams. These diagrams have two *swimlanes* that show the respective behaviours performed by the user (*User*) and system (*MVP CYOA System*). Each of the system's behaviours are performed by specific system components, which are annotated in each state within parentheses as one of the following:

- ***CYOA***: The CYOA authentication scheme provides the functionality that allows users to select one of the supported MVP authentication schemes for login.
- ***MVP***: In Figures 7.9 and 7.10, the core MVP system is solely responsible for loading whatever scheme is requested. CYOA is implemented as one of these schemes that can be loaded by MVP. Thus, when the user begins the registration or login process from the website, it will request that MVP load the CYOA scheme. Soon thereafter, when the user chooses a scheme from the CYOA scheme selection form, CYOA will request that MVP load the user's chosen scheme.
- ***Scheme***: The user's chosen scheme merely needs to launch itself when requested by MVP, after being chosen by the user through CYOA.
- ***Website***: This is the MVP website where the user wishes to log in to their account. The website must first prompt the user for their username, which is required to determine their assigned experimental scheme (CYOA, in this case). The website also must verify that the result of the authentication process (the entered password) matches the stored authentication result for that user.

Registration

Before performing tasks that require an account, a user must first create an account. Although users must create an account for each website they wish to use, the registration process is identical for all MVP websites. Figure 7.9 represents the account

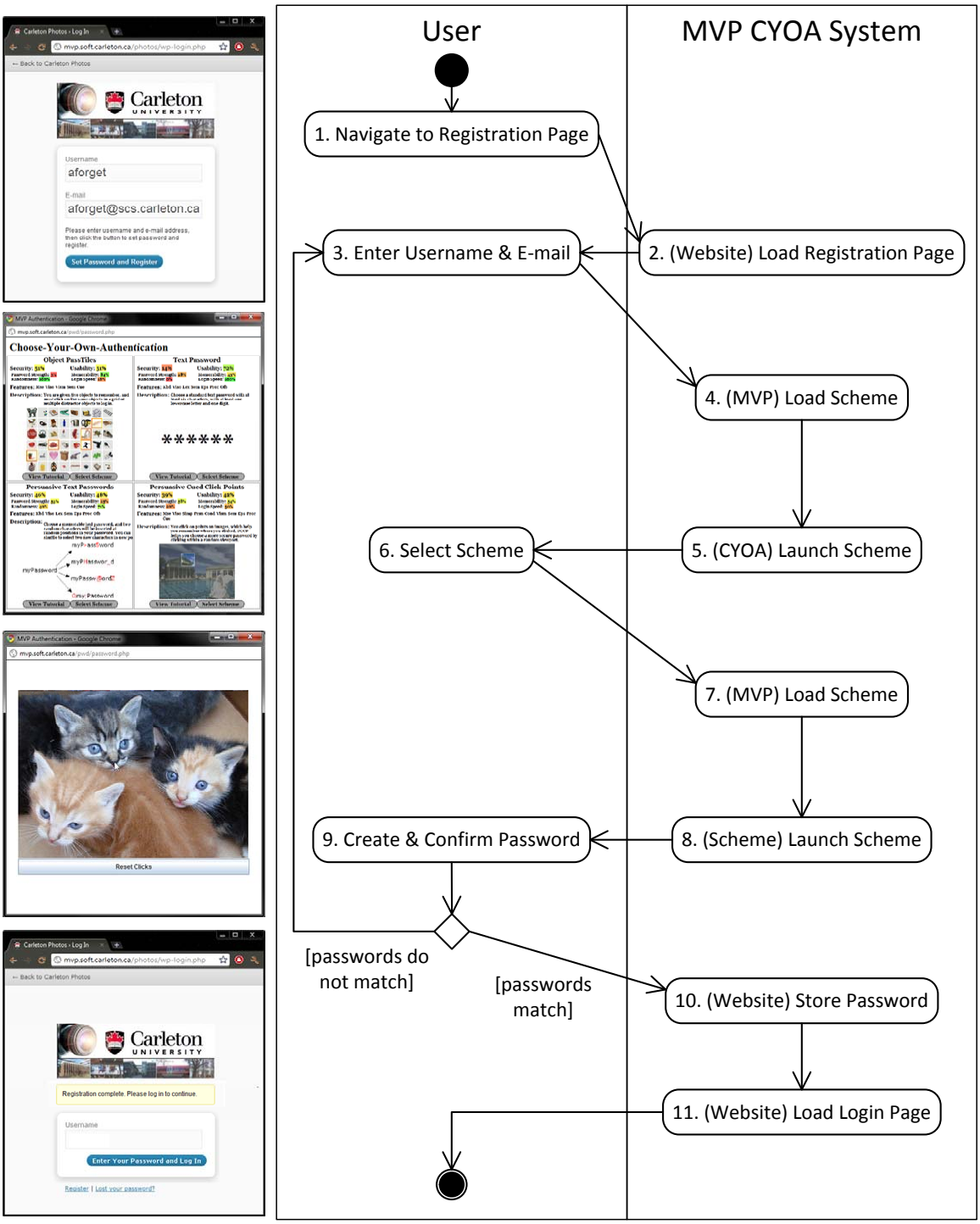


Figure 7.9: UML 2.4.1 [145] activity diagram of a user registering for an account on an MVP website using CYOA. The images on the left illustrate the user interfaces on which the user performs each corresponding action.

registration process as a UML 2.4.1 [145] activity diagram. Figure 7.9 includes the following behaviours:

1. The registration process begins with the user navigating to the MVP website's registration page with an HTTP request.

2. The website responds with an HTML form containing two text input fields labelled "Username" and "E-mail", and a button to proceed to create their password.

3. The user will enter their given username of their account and their e-mail address [1] and submit the form to the MVP scheme loader by clicking on the form's button (or pressing the `Enter` key). The form submission also includes, as hidden form input parameters, the website's unique name and the authentication mode (set to `create`, which signals that the user is creating a new password, as opposed to `enter` for logging in). A new window is opened where the server's response will be shown. This second window is where the password creation will take place.

4. Since no scheme was specified in the form submission, MVP defaults to looking up in its database the authentication scheme assigned to the specified user and website pair. In the context of this chapter, this assigned scheme is always CYOA. Thus, MVP launches CYOA.

5. CYOA displays the scheme selection form (shown in Figures 7.2b). Each panel contains the following HTML items describing an available authentication scheme supported by CYOA:

    - The name of the authentication scheme.
    - The security and usability ratings (Section 7.5.1) and the user-centred authentication features (Chapter 3) supported by the scheme. The user may mouseover any of these items for an explanation of their significance.

[1] Users' e-mail address is mainly used for the automated password reset system if the password is forgotten.

- A brief description of the scheme.
- A visual representation of the scheme, be it a screenshot, illustration, or icon.

To prevent any order-based selection bias, the schemes are presented in a random order by default, unless the administrative settings are changed to present the schemes in a fixed order. The user may click on a scheme's `View Tutorial` button to explore its tutorial. The user chooses a scheme by clicking on its `Select Scheme` button. The username, website, and authentication mode are included as three hidden `input` form variables.

6. The user may inform their choice of a scheme by examining the ratings, features, descriptions, and tutorials of the available schemes through the scheme selection form. Ultimately, the user will submit the form with their choice of scheme to the MVP scheme loader, requesting the chosen scheme.

7. MVP launches the user's chosen scheme.

8. The chosen scheme is initialised and the scheme's password entry object (which could be an HTML/Javascript form, Java applet, or any other web object) is sent to the user.

9. The user enters their new password twice with their chosen scheme; once to create it, and a second time to confirm (unless the scheme signalled to MVP that confirmation was unnecessary, because the scheme handles the confirmation internally). If the two password results do not match, then the user is informed of the password mismatch, the second window is closed (effectively returning the user to the website's registration page at *state 3* (Figure 7.9), except their username and e-mail will still be in the input text boxes, since the main page will only change if the password registration process completes successfully). If the two password results do match, then the password result is sent to the website and the MVP authentication window is closed.

10. The website stores the user's password result, paired to their username.

11. The website loads the login form with a message stating that the registration is complete, and returns the form to the user.

Login

After successfully registering for an account on an MVP website, the user may log in to their account to perform tasks requiring an account, such as commenting on blog posts, voting on polls, or whatever is appropriate for the particular website's purpose. Figure 7.10 illustrates the login process as a UML 2.4.1 activity diagram. We describe the behaviours performed at each state as follows:

1. The login process begins with the user navigating to the MVP website's login page with an HTTP request, assuming the user has not already done so.

2. The website responds with an HTML form containing a text input field labelled "Username" and a button to proceed to enter their password.

3. The user will enter the username of their account on this website and submit the form to the MVP scheme loader by clicking on the form's button (or pressing the `Enter` key). The form submission also includes the website's unique name and the authentication mode (set to `enter`, which signals that the user is performing a login, as opposed to `create` for registering a new password). A new window is opened where the server's response will be shown. This second window is where the MVP authentication will take place.

4. Since no scheme was specified in the form submission, MVP looks up in its database the authentication scheme assigned to the specified user and website pair. In the context of this chapter, this assigned scheme is always CYOA. Thus, MVP launches CYOA.

5. CYOA returns to the user a scheme selection form (shown both in Figures 7.2b and 7.9). Each panel contains the following HTML items describing an available authentication scheme supported by CYOA:

    - The name of the authentication scheme.

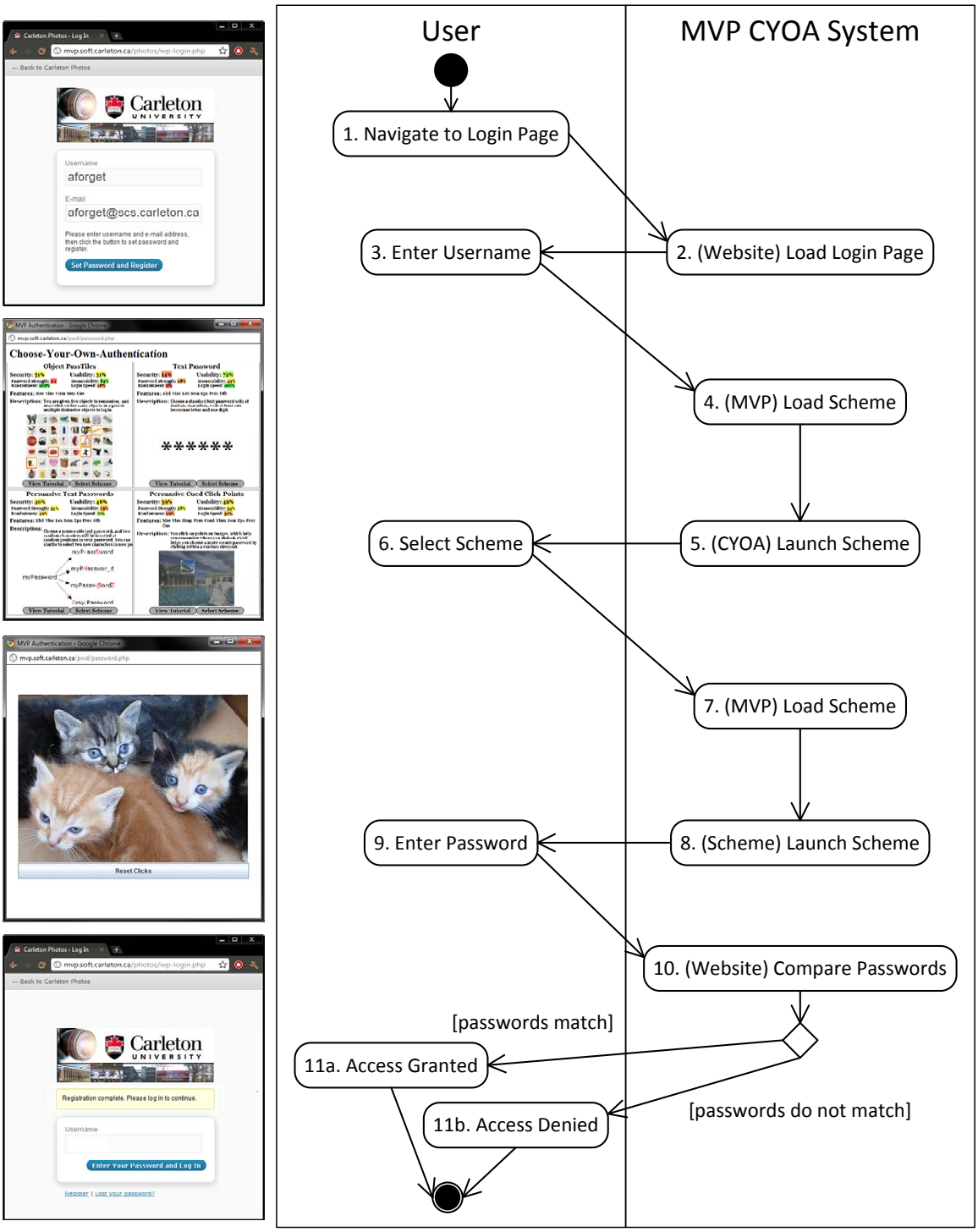


Figure 7.10: UML 2.4.1 [145] activity diagram of a user logging in to an MVP website with CYOA. The images on the left illustrate the user interfaces on which the user performs each corresponding action.

- The overall security and usability ratings (Section 7.5.1) and the user-centred authentication features (Chapter 3) supported by the scheme. The user may mouseover any of these items for an explanation of their significance.
- A brief description of the scheme.
- A visual representation of the scheme, be it a screenshot, illustration, or icon.

As during registration, the schemes are presented in a random order. The `View Tutorial` buttons are also available for users to view schemes' tutorials to help them remember which scheme they chose. The user finally chooses a scheme by clicking on its `Select Scheme` button. The username, website, and authentication mode are included as three hidden `input` form variables.

6. The user submits their scheme to the MVP scheme loader, requesting the chosen scheme.

7. MVP launches the user's chosen scheme.

8. The chosen scheme is initialised and the scheme's password entry object (which could be an HTML/Javascript form, Java applet, or any other web object) is sent to the user.

9. The user enters their password with their chosen scheme. When the password entry is complete, the password result is sent to the website and the MVP authentication window is closed.

10. The website compares the submitted password result to the user's stored account password. If the passwords match, the website will return its homepage with the additional functionality granted by the user's account (Figure 7.10, *state 11a*). If the passwords do not match, then the user is returned the website's login page (as described in *state 2* (Figure 7.10)) with an additional message informing the user of the failed authentication (Figure 7.10, *state 11b*).

11. If the authentication was successful (a), the user may perform account-specific tasks supported by the website. If the authentication was unsuccessful (b), the user may try to login again (return to *state 3* (Figure 7.10)), or attempt to reset their password by clicking on the `Lost your password?` link (shown both in Figures 7.2a and 7.10).

One set of possible behaviours not illustrated in Figures 7.9 and 7.10 is at the password entry states. Some authentication schemes may need to periodically request server-side operations or resources while the user enters their password. One example is Cued Click-Points [46], where for each click-point in the user's password, the client requests the next image from the server. It would also not be feasible to send all (at least several hundred) images to the client at every login.

Additionally, CYOA could present the user with the correct scheme at login, rather than requiring its selection. Both approaches are discussed in Section 7.4.2.

7.6 Future Usability Evaluation

CYOA gives users the choice of authentication scheme when registering and logging in to an application or website. We have discussed two possible architectures, some minor variations, explored their advantages and disadvantages, and described our working prototype. We have proposed four schemes in particular, whose usability and security has been independently established. In fact, the scheme diversity offered by CYOA could even improve the usability and security of the added schemes. However, this should be confirmed with empirical user studies.

In such studies, experimenters will closely observe users' behaviour when given a choice of multiple authentication methods. A key challenge in the study would be determining whether or not users select different schemes in different circumstances. For example, we would hope that users would select a more secure scheme (and password) when creating a password for a high-risk account (such as an online bank account) than for a lower-risk account (such as an online newspaper). However, without incorporating CYOA into users actual online bank accounts (which would pose serious ethical and logistical problems), it may be difficult to design a user study

where users are sufficiently motivated to modify their behaviour when registering passwords for fake accounts, as is typically done in user studies. Thus, we leave the final design and execution of a study of CYOA as future work, even though we have already made noteworthy progress in preparation for the study. This section will describe the work we have completed in preparation for such a user study, and discuss possible modifications to CYOA depending on the outcome of said study.

Section 7.4.2 discusses the security and usability trade-off between requiring users to select their authentication scheme at login or simply presenting them with the correct scheme. We would use the latter approach in a user study, since we feel the former would add only a little security at possibly significant cost to usability. If CYOA proves to be usable and beneficial for users, later studies could explore different CYOA configurations, including requiring users to select their scheme at login.

Our MVP prototype implementation of CYOA (Section 7.5) could be used for a user study. The MVP authentication research experimentation platform [38] includes websites where participants use an assigned scheme to login. These live websites include blogs where users login to post comments on the articles or vote on polls. We consider accounts at such websites to be low-risk, since adversaries have little to gain by attacking accounts on blog websites. Thus, a higher-risk website would also be needed to observe whether participants will modify their choice of scheme for accounts on websites with different risk profiles. Examples of higher-risk websites include an online banking or e-commerce website. However, there are challenges inherent in emulating the high-risk nature of real bank accounts with a simulated website. The high-risk website will not contain any of participants' real money, resulting in the high-risk accounts being of little actual personal value to users. Thus, participants may not perceive any risk to their simulated website accounts, and not modify their scheme selection behaviour between simulated websites. We could design a complex game with varying rewards, but we consider this out of scope in this thesis.

In one variation of such rewards to influence participants to value their simulated accounts similarly to how they value their own real-life accounts, we could tell participants that they will be provided a bonus honorarium should their accounts not

be hacked. For example, we would initially tell users that they will receive a small bonus for each low-risk website account that is not hacked, and a larger bonus if their higher-risk website account remains secure. We hope this will motivate users to care a little about their low-risk accounts' security, but care much more about their higher-risk accounts' security, as they normally would. Despite what participants will be told and their subsequent behaviour, participants should be provided with the full bonus honorarium in order to ethically treat all participants the same. We note that this approach has ethical concerns regarding deception.

7.6.1 Discussion

We speculate that users would reap the full benefits of CYOA by choosing different schemes for different accounts, particularly choosing more secure schemes for higher-risk accounts. However, we believe it prudent to anticipate alternative user study outcomes and be prepared to make adjustments.

One such possibility is that users may predominately select text passwords over any other scheme. The most obvious solution would be to simply make standard text passwords unavailable, forcing users to make another choice. However, this may not be the best approach. Users might default to selecting the most text password-like scheme. Furthermore, some users have developed complex text password generation methods which already result in secure passwords they can remember (Section 4.5). We should try to avoid frustrating said users by forcing them to learn to use new schemes.

To better understand users' behaviour, they should be asked after the user study *why* they made their particular scheme selections. Should they choose the same scheme across multiple websites, we believe this may suggest that users either did not understand the benefits of CYOA or believed that their favoured scheme (most likely text passwords) provided sufficient memorability and security for their purposes. These would most likely represent knowledge gaps regarding CYOA, password memorability and password security that we should help users bridge.

Instead of removing text passwords from CYOA, we would instead use persuasion [77] to better instruct users on the benefits of CYOA and influence them to

select different schemes. The first place to employ persuasion would be in the tutorial, which could explicitly illustrate why choosing different schemes increases both the security and memorability of their passwords. Furthermore, the CYOA interface could also attempt to persuade users to select different schemes. For example, if the mouse hovers over a scheme the user has previously selected for another account, CYOA could display a tooltip message stating, "You already chose this scheme for another account. If you choose a new scheme for this account, it might be easier to remember your passwords." Any persuasion in the CYOA interface itself must assume that the three-party CYOA architecture (Section 7.4.3) is in use, where the trusted third-party CYOA system may have some data regarding which schemes were selected by which user or computer (depending on specifically how CYOA is implemented). However, in the two-party architecture (Section 7.4.2), where the user is registering an account at a higher-risk website (such as an online bank), the system could similarly persuade users to choose more secure schemes. For example, should the user hover their mouse cursor over the standard text password scheme (or another less secure scheme), a tooltip could suggest, "To keep your money safe, you may want to choose a scheme that helps you create a more secure password, like Persuasive Text Passwords, Persuasive Cued Click-Points, or Object PassTiles."

Another possible improvement would be in how the scheme-related information is presented. For example, the rating scores could be displayed as meters rather than (or in addition to) the current numerical percentages, which may be difficult for some users to compare. Furthermore, some users may wish to gain a deeper understanding on specifically how the schemes' features were assigned and ratings were scored. Clearly, we should support users in educating themselves about security and usability issues. Thus, the ratings could be hyperlinks that display an HTML page with a detailed description of the rating or feature in question. The tutorial could also place a stronger emphasis on the ratings to encourage users to examine them while choosing a scheme.

Additional subtypes of security and usability ratings could also be added. In addition to providing more information to users, these may also prompt them to learn

about potential password threats and usability issues they had not previously considered. This additional understanding may better inform their choice of both scheme and password. Conversely, users may be overwhelmed by the amount of information they are presented. Thus, it may be beneficial to either remove information users reported was not useful or meaningful, or hide more detailed information by default. For example, the security and usability subratings (Password Strength, Randomness, Memorability, Login Speed, and any others) could initially be hidden, and upon clicking either rating, the relevant subratings could then be revealed. This could both prevent users from being overwhelmed by too much information, while still making relevant information accessible, if users desire to investigate further. Users may also find it helpful to be able to sort or filter the schemes by rating scores or features, to more quickly identify schemes with particular scores or features.

7.7 Conclusion

Plaintext passwords present well-known usability and security challenges for modern users. Given the choice, some users may prefer to authenticate to applications with different schemes. This chapter presented CYOA, an architecture to allow users the freedom to choose, from a multitude of authentication schemes, one that best suits their preferences and abilities. CYOA is designed to require very few changes to replace an existing text password system. Adding and removing schemes supported by CYOA is easy for administrators. It is also possible for a trusted third-party to offer CYOA services to which applications can delegate authentication-related tasks. This relieves the burden of maintaining an authentication system from application developers, who may not be security experts.

We also outlined a user study design to evaluate users' behaviour and decision-making process when selecting an authentication scheme amongst several available options. Results from such a study would measure users' ability and willingness to cope with choosing between novel authentication schemes. This study would help guide future usability improvements to better support users in creating and entering appropriately secure and memorable passwords in a world with many authentication schemes.

Chapter 8

Conclusion

Despite text passwords' long standing as the incumbent knowledge-based authentication (KBA) scheme, users continue to face challenges in creating secure and memorable passwords. Alternative authentication schemes have been proposed, but authentication researchers and professionals continue to struggle with developing a scheme to overthrow text passwords. We agree with Bonneau et al. [24] that is it highly unlikely a single scheme will surpass the convenience of text passwords. Thus, we propose providing users with the choice of multiple schemes. Such a world with many authentication schemes would have the following benefits:

- ***Accommodates user preference.*** People have different cognitive abilities. Users with stronger visual or lexical memory could respectively select graphical and textual passwords. Some users may appreciate schemes that assist them in creating more secure and memorable passwords, while other users may desire schemes without such assistance. Given the choice, users could select authentication schemes that better suit their preferences.

- ***Educates about authentication concerns.*** To help users differentiate between the various schemes' strengths and weaknesses, descriptive information about the available schemes (e.g. ratings, features) should be shown when the user is selecting a scheme. This would give users some exposure to the security and usability issues in authentication, which may help them form better mental models of authentication and may increase their proficiency in creating better passwords.

- ***Supports accessibility.*** Research with dyslexic users [73] and speech-to-text software [166] suggests that text passwords may be challenging for users with particular disabilities. However, creating a single universally accessible scheme

is a difficult [69] (if not unsolvable [169]) problem. Instead, users should be provided with many possible schemes, each of which cater to different users' abilities. This would make the yet unsolved problem of accessible authentication solvable, since schemes supporting users with disabilities could easily be added to the ecosystem of available schemes.

- ***Resistance against password guessing and phishing attacks.*** In a world with many authentication schemes, attackers would need to perform more work to guess or capture users' passwords. To launch an effective password guessing attack, attackers would probably need to build a different dictionary of candidate passwords for each scheme supported by the target system. It may also be more challenging for attackers to build credible phishing websites, since they would need to support the same schemes as the legitimate website. Both of these attacks could become difficult for attackers to launch, since administrators can easily modify the ecosystem of available schemes by removing older schemes and adding new ones. This would force attackers to use more complicated and costly methods, which may deter less motivated attackers.

- ***Delegates authentication security to experts.*** System administrators may not have the same expertise in authentication as researchers or professionals in our field. Thus, all stakeholders in authentication would benefit from known and trusted authentication experts developing, certifying, and widely-publishing schemes for use by any administrator. This relationship could be taken one step further by administrators delegating the scheme selection and password entry process to a trusted third-party. These approaches relieve administrators and non-security organisations of some of the authentication maintenance burden.

8.1 Research Contributions

This thesis has proposed to provide users with the ability to select an authentication scheme that suits their abilities, preferences, and usage context, out of an array of available authentication schemes. We have presented the following specific contributions to further this research agenda:

- We first proposed the User-Centred Authentication Feature Framework (Chapter 3) to support all authentication stakeholders in identifying key features in authentication schemes. Scheme designers can use our framework in specifying the features of most importance to their target audience, authentication researchers and professionals in analysing and comparing schemes' features, and users in short-listing schemes that support the features most relevant to them.

- We developed and evaluated two novel authentication schemes, Persuasive Text Passwords (PTP, Chapter 4) and Cued Gaze-Points (CGP, Chapter 5), both with different usability and security advantages than previously proposed schemes. In designing PTP, we used our framework to identify the persuasive features in the graphical password scheme Persuasive Cued Click-Points [46], and leveraged the same persuasive features to improve text-based passwords. By only modifying the input modality, CGP offers the same benefits of Cued Click-Points (CCP) [44] to users who could not have used CCP. Our user studies demonstrated that people could create and log in with passwords using the schemes' unique combination of features that users had never previously encountered.

- We built and tested four different tutorials (Chapter 6) designed for people to teach themselves how to use a novel authentication scheme. Participants using any scheme successfully learnt to create and log in with a novel scheme. However, users of the simpler text or hypertext tutorials learnt more quickly and effectively than users of more complex demo and video tutorials. Given these results and parallel findings by Stobert and Biddle [189], we recommended administrators of novel authentication schemes provide users with a relatively simple tutorial of text and static images, rather than interactive demonstrations or elaborate training videos.

- We presented Choose Your Own Authentication (CYOA, Chapter 7), an architecture supporting many authentication schemes from which users may select one that fulfills their particular usability and security requirements. CYOA helps administrators easily add and remove authentication schemes, and can potentially delegate the burden of managing the user authentication procedure

to a trusted third-party of authentication experts. Researchers and other authentication scheme designers can easily build and test CYOA schemes, and offer them to the authentication community for further research and adoption in practitioners' CYOA systems. In our MVP [38] authentication experimentation platform, we implemented CYOA, a set of compatible authentication schemes, and accompanying tutorials. We also explored the design of a user study to evaluate how CYOA users deal with and take advantage of the power of choice in authentication schemes.

8.2 Research Directions

CYOA user studies. Before widely-deploying CYOA, an empirical user study of CYOA should be performed to confirm that the usability and security of the supported schemes is not negatively impacted. Experimenters should also inquire about users' scheme selection behaviour to ascertain any impact on the usability and security of CYOA and the available schemes. We discuss the ramifications of this possibility and subsequent improvements to CYOA in Section 7.6.

CYOA support for non-KBA schemes. In Sections 7.4.2 and 7.4.2, we briefly discussed a limitation of CYOA's current design whereby schemes requiring a particular password verification function (other than the typical bitwise direct comparison) cannot be supported without special modifications to the architecture. It may be beneficial to further examine how CYOA could be extended to better incorporate challenge-response and non-KBA schemes. This would benefit users and administrators wishing to leverage the entire space of authentication schemes, including biometrics and token-based schemes.

Accessible authentication. CYOA provides the means to easily add authentication schemes designed for users with disabilities. However, there are very few schemes that have been proposed to support authentication accessibility needs. The social awareness and demand for accessibility is on the rise, as is the need to bridge this gap in the usable authentication literature.

Novel schemes through feature manipulation. In Chapter 3, we identified features that authentication schemes may support. We then presented Persuasive

Text Passwords (Chapter 4) and Cued Gaze-Points (Chapter 5) as examples of how such features identified in one promising scheme can be manipulated to become a novel scheme which retains many of the original scheme's benefits. Further re-combinations of schemes' features should be performed to provide novel representations of the beneficial aspects of promising schemes. Such derived schemes that prove to be usable and secure will of course benefit all stakeholders of CYOA when added into the ecosystem of authentication schemes.

8.3 Conclusion

We believe this thesis makes significant contributions to the field of usable authentication, particularly knowledge-based authentication (KBA). The considerable benefits of KBA (Section 2.1) demonstrate why it has remained the authentication method of choice for decades and is likely to remain popular [24, 103]. Should practitioners rely more heavily on other forms of authentication (e.g. biometrics, token-based), we believe our User-Centred Authentication Feature Framework and Choose Your Own Authentication (CYOA) proposals could be extended to include forms of authentication beyond KBA. Until then, we hope the contributions of this thesis are refined through further research. We look forward to the day when our proposals are widely-deployed to provide all users with the practical benefits of a world with many authentication schemes.

Bibliography

[1] A. Adams and M. Sasse. Users are not the enemy. *Communications of the ACM*, 42(12), 1999.

[2] AgileBits Inc. 1password, accessed July 2012. `https://agilebits.com/onepassword`.

[3] Amazon.com Inc. Amazon mechanical turk, accessed September 2011. `https://www.mturk.com`.

[4] J. Anderson and G. Bower. Recognition and retrieval processes in free recall. *Psychological Review*, 79(2), March 1972.

[5] R. Anderson. Why cryptosystems fail. In *1st Conference on Computer and Communications Security*. ACM, December 1993.

[6] T. Anderson, M. Blank, B. Daniels, and D. Lebling. *Zork I*. Infocom, 1980.

[7] D. Ankrum. Viewing distance at computer workstations. *Workplace Ergonomics*, 2(5), September/October 1996.

[8] Apple Inc. iPhone website, accessed July 2012. `http://www.apple.com/iphone/`.

[9] A. Baddeley, M. Eysenck, and M. Anderson. *Memory*. Psychology Press, 1st edition, 2009.

[10] A. Baddeley and G. Hitch. *Working Memory*, volume 8, pages 47–89. Academic Press, 1974.

[11] X. Bai, W. Gu, S. Chellappan, X. Wang, D. Xuan, and B. Ma. PAS: Predicate-based authentication services against powerful passive adversaries. In *Annual Computer Security Applications Conference (ACSAC)*. IEEE, 2008.

[12] Bartavelle. Patches for John the Ripper, accessed February 2008.

[13] B. Barton and M. Barton. User-friendly password methods for computer-mediated information systems. *Computers & Security*, 3(3), 1984.

[14] S. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *Research in Security and Privacy*. IEEE, May 1992.

[15] A. Bianchi, I. Oakley, and D. Kwon. The secure haptic keypad: A tactile password system. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2010.

[16] A. Bianchi, I. Oakley, J. Lee, and D. Kwon. The haptic wheel: Design & evaluation of a tactile password system. In *Conference on Human Factors in Computing Systems (CHI) Extended Abstracts*. ACM, 2010. Work-in-progress poster.

[17] K. Bicakci, N. Atalay, M. Yuceel, and P.C. van Oorschot. Exploration and field study of a browser-based password manager using icon-based passwords. In *International Conference on Financial Cryptography and Data Security (FC) Workshop on Real-Life Cryptographic Protocols and Standardization*. Springer, 2011. LNCS 7126.

[18] K. Bicakci, N. Atalay, M. Yuceel, and P.C. van Oorschot. Exploration and field study of a browser-based password manager using icon-based passwords. Technical Report TR-11-07, School of Computer Science, Carleton University, January 2011.

[19] R. Biddle, S. Chiasson, and P.C. van Oorschot. Graphical passwords: Learning from the first twelve years. *ACM Computing Surveys*, 44(4), 2012. in press.

[20] M. Bishop. A proactive password checker. Technical Report PCS-TR90-152, National Aeronautics and Space Administration (NASA), 1990. `http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19920018383_1992018383.pdf`.

[21] M. Bishop and D. Klein. Improving system security via proactive password checking. *Computers & Security*, 14(3), 1995.

[22] H. Bojinov, D. Sanchez, P. Reber, D. Boneh, and P. Lincoln. Neuroscience meets cryptography: Designing crypto primitives secure against rubber hose attacks. In *USENIX Security Symposium*, 2012.

[23] M. Bond. Comments on GrIDsure authentication. `http://www.cl.cam.ac.uk/~mkb23/research/GridsureComments.pdf`, March 2008.

[24] J. Bonneau, C. Herley, P.C. van Oorschot, and F. Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *IEEE Symposium on Security and Privacy*, 2012.

[25] J. Bonneau, C. Herley, P.C. van Oorschot, and F. Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. Technical Report 817, University of Cambridge Computer Laboratory, 2012. `www.cl.cam.ac.uk/techreports/UCAM-CL-TR-817.html`.

[26] J. Brainard, A. Juels, R. Rivest, M. Szydlo, and M. Yung. Fourth-factor authentication: Somebody you know. In *13th Conference on Computer and Communications Security (CCS)*. ACM, 2006.

[27] S. Brewster and L. M. Brown. Non-visual information display using tactons. In *Conference on Human Factors in Computing Systems (CHI) Extended Abstracts*. ACM, 2004.

[28] S. Brostoff, P. Inglesant, and M. Sasse. Evaluating the usability and security of a graphical one-time pin system. In *BCS Interaction Specialist Group Conference*. British Computer Society, 2010.

[29] J. Brustoloni and R. Villamarin-Salomón. Improving security decisions with polymorphic and audited dialogs. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2007.

[30] W. Burr, D. Dodson, and W. Polk. Electronic authentication guideline. Technical report, National Institute of Standards and Technology (NIST), April 2006. Special Publication 800-63 Verstion 1.0.2.

[31] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer. OpenPGP message format. Technical Report RFC 4880, IETF, November 2007. `http://tools.ietf.org/html/rfc4880`.

[32] J. Carroll, P. Smith-Kerker, J. Ford, and S. Mazur-Rimetz. The minimal manual. *Human-Computer Interaction*, 3(2), 1987.

[33] D. Carstens, L. Malone, and P. McCauley-Bell. Applying chunking theory in organizational password guidelines. *Journal of Information, Infortmation Technology, and Organizations*, 1, 2006.

[34] C. Castelluccia, M. Duermuth, and D. Perito. Adaptive password strength meters from Markov models. In *Network and Distributed System Security Symposium (NDSS)*. ISOC, February 2012.

[35] D. Charoen, M. Raman, and L. Olfman. Improving end user behaviour in password utilization: An action research initiative. *Systemic Practice and Action Research*, 21(1), 2008.

[36] S. Chiasson. *Usable Authentication and Click-Based Graphical Passwords*. PhD thesis, School of Computer Science, Carleton University, 2008.

[37] S. Chiasson, R. Biddle, and P.C. van Oorschot. A second look at the usability of click-based graphical passwords. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2007.

[38] S. Chiasson, C. Deschamps, E. Stobert, M. Hlywa, B. F. Machado, A. Forget, N. Wright, G. Chan, and R. Biddle. The MVP web-based authentication framework. In *International Conference on Financial Cryptography and Data Security (FC)*. Springer, 2012.

[39] S. Chiasson, A. Forget, and R. Biddle. Accessibility and graphical passwords. In *Symposium on Accessible Privacy and Security (SOAPS)*. ACM, 2008.

[40] S. Chiasson, A. Forget, R. Biddle, and P.C. van Oorschot. Influencing users towards better passwords: Persuasive Cued Click-Points. In *People and Computers XXII*. British Computer Society, 2008.

[41] S. Chiasson, A. Forget, R. Biddle, and P.C. van Oorschot. User interface design affects security: Patterns in click-based graphical passwords. *International Journal of Information Security*, 8(6), 2009.

[42] S. Chiasson, A. Forget, E. Stobert, R. Biddle, and P.C. van Oorschot. Multiple password interference in text and click-based graphical passwords. In *16th Conference on Computer and Communications Security (CCS)*. ACM, November 2009.

[43] S. Chiasson, P.C. van Oorschot, and R. Biddle. A usability study and critique of two password managers. In *USENIX Security Symposium*, 2006.

[44] S. Chiasson, P.C. van Oorschot, and R. Biddle. Graphical password authentication using Cued Click Points. In *European Symposium On Research In Computer Security (ESORICS), LNCS 4734*, 2007.

[45] S. Chiasson, J. Srinivasan, R. Biddle, and P.C. van Oorschot. Centered discretization with application to graphical passwords. In *Usability, Psychology, and Security (UPSEC)*. USENIX, 2008.

[46] S. Chiasson, E. Stobert, A. Forget, R. Biddle, and P.C. van Oorschot. Persuasive Cued Click-Points: design, implementation, and evaluation of a knowledge-based authentication mechanism. *IEEE Transactions on Dependable and Secure Computing (TDSC)*, March/April 2012.

[47] N. Christin, S. Egelman, T. Vidas, and J. Grossklags. It's all about the benjamins: An empirical study on incentivizing users to ignore security advice. In *International Conference on Financial Cryptography and Data Security (FC)*. Springer, 2011.

[48] N. Clarke and S. Furnell. Authentication of users on mobile telephones a survey of attitudes and practices. *Computers & Security*, 24(7), 2005.

[49] B. Coskun and C. Herley. Can "something you know" be saved? In *Information Security Conference*, volume 5222/2008. Springer LNCS, 2008.

[50] N. Cowan. The magical number 4 in short-term memory: A reconsideration of mental storage ca-pacity. *Behavioral and Brain Sciences*, 24(1), 2001.

[51] L. Cranor and S. Garfinkel. *Security and Usability: Designing Systems that People Can Use*. O'Reilly, 2005.

[52] D. Davis, F. Monrose, and M. Reiter. On user choice in graphical password schemes. In *USENIX Security Symposium*, 2004.

[53] A. De Angeli, L. Coventry, G. Johnson, and K. Renaud. Is a picture really worth a thousand words? Exploring the feasibility of graphical authentication systems. *International Journal of Human-Computer Studies*, 63(1-2), 2005.

[54] A. De Luca, M. Denzel, and H. Hussmann. Look into my eyes! Can you guess my password? In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2009.

[55] A. De Luca, E. von Zezschwitz, and H. Hußmann. VibraPass - secure authentication based on shared lies. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2009.

[56] S. Designer. John the Ripper password cracker, accessed December 2008. `http://www.openwall.com/john/`.

[57] R. Dhamija and L. Dusseault. The seven flaws of identity management: Usability and security challenges. *Security & Privacy*, March/April 2008.

[58] R. Dhamija, J. Tygar, and M. Hearst. Why phishing works. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2006.

[59] A. Dirik, N. Menon, and J. Birget. Modeling user choice in the PassPoints graphical password scheme. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, July 2007.

[60] A. Duchowski. *Eye Tracking Methodology: Theory and Practice*. Springer, 2$^{\text{nd}}$ edition, 2007.

[61] P. Dunphy, A. Fitch, and P. Olivier. Gaze-contingent passwords at the ATM. In *4th Conference on Communication by Gaze Interaction (COGAIN)*, 2008.

[62] eBay Inc. eBay home page, accessed June 2012. `http://www.ebay.com/`.

[63] Eclipse Foundation. Higgins Personal Data Service, accessed January 2012. `http://www.eclipse.org/higgins/`.

[64] H. Ellis and R. Hunt. *Fundamentals of Human Memory and Cognition*. Wm. C. Brown, 4$^{\text{th}}$ edition, 1989.

[65] J. Ellis and L. Kvavilashvili. Prospective memory in 2000: Past, present, and future directions. *Applied Cognitive Psychology*, 14(7), 2000.

[66] A. Evans, W. Kantrowitz, and E. Weiss. A user authentication scheme not requiring secrecy in the computer. *Communications of the ACM*, 17(8), August 1974.

[67] K. Everitt, T. Bragin, J. Fogarty, and T. Kohno. A comprehensive study of frequency, interference, and training of multiple graphical passwords. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2009.

[68] EyeTrackShop. Webcam Eye Tracking, accessed May 2012. `http://eyetrackshop.com/eye-tracking`.

[69] P. Fairweather, V. Hanson, S. Detweiler, and R. Schwerdtfeger. From assistive technology to a web accessibility service. In *International Conference on Assistive Technologies (ASSETS)*. ACM, 2002.

[70] L. Falk, A. Prakash, and K. Borders. Analyzing websites for user-visible security design flaws. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2008.

[71] Federal Information Processing Standards (FIPS). Password usage. Publication 112, May 1985. `http://www.itl.nist.gov/fipspubs/fip112.htm`.

[72] Federal Information Processing Standards (FIPS). Withdrawn FIPS by numerical order index, accessed December 2008. `http://www.itl.nist.gov/fipspubs/withdraw.htm`.

[73] R. Fink. Literacy development in successful men and women with dyslexia. *Annals of Dyslexia*, 48(1), 1998.

[74] D. Florêncio and C. Herley. A large-scale study of WWW password habits. In *International World Wide Web Conference (WWW)*. ACM, May 2007.

[75] D. Florêncio and C. Herley. Where do security policies come from? In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, July 2010.

[76] D. Florêncio, C. Herley, and B. Coskun. Do strong web passwords accomplish anything? In *Workshop on Hot Topics in Security (HotSec)*. USENIX, 2007.

[77] B. Fogg. *Persuasive Technology: Using Computers to Change What We Think and Do*. Morgan Kaufmann Publishers, 2003.

[78] A. Forget and R. Biddle. Memorability of persuasive passwords. In *Conference on Human Factors in Computing Systems (CHI) Student Research Competition*. ACM, 2008.

[79] A. Forget, S. Chiasson, and R. Biddle. Lessons from brain age on persuasion for computer security. In *Conference on Human Factors in Computing Systems (CHI) Work-In-Progress*. ACM, 2009.

[80] A. Forget, S. Chiasson, and R. Biddle. Input precision for gaze-based graphical passwords. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2010.

[81] A. Forget, S. Chiasson, and R. Biddle. Shoulder-surfing resistance with eye-gaze entry in click-based graphical passwords. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2010.

[82] A. Forget, S. Chiasson, R. Biddle, and P.C. van Oorschot. Persuasion as education for computer security. In *World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education (E-Learn)*. AACE, 2007.

[83] A. Forget, S. Chiasson, R. Biddle, and P.C. van Oorschot. Can old dogs learn new password tricks? supporting learning of novel authentication schemes. In *World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education (E-Learn)*. AACE, 2012. in submission.

[84] A. Forget, S. Chiasson, P.C. van Oorschot, and R. Biddle. Improving text passwords through persuasion. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2008.

[85] A. Forget, S. Chiasson, P.C. van Oorschot, and R. Biddle. Persuasion for stronger passwords: Motivation and pilot study. In *3rd International Conference on Persuasive Technology*, 2008.

[86] I. C. Foundation. Information card specifications, accessed January 2012. `http://informationcard.net/specifications/`.

[87] S. Furnell. An assessment of website password practices. *Computers & Security*, 26(7-8), 2007.

[88] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.

[89] S. Gathercole. Cognitive approaches to the development of short-term memory. *Trends in Cognitive Science*, 3(11), 1999.

[90] S. Gaw and E. Felten. Password management strategies for online accounts. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, July 2006.

[91] P. Ginns. Meta-analysis of the modality effect. *Learning and Instruction*, 15(4), August 2005.

[92] F. Gobet and G. Clarkson. Chunks in expert memory: Evidence for the magical number four...or is it two? *Memory*, 12(6), 2004.

[93] E. Goldstein. *Cognitive Psychology: Connecting Mind, Research, and Everyday Experience*. Wadsworth Publishing, 3rd edition, 2011.

[94] Google Inc. Gmail: Email from google, accessed June 2012. `http://www.gmail.com`.

[95] Google Inc. YouTube - broadcast yourself, accessed July 2012. `http://www.youtube.com`.

[96] T. Grossman and G. Fitzmaurice. Toolclips: an investigation of contextual video assistance for functionality understanding. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2010.

[97] T. Grossman, G. Fitzmaurice, and R. Attar. A survey of software learnability: metrics, methodologies and guidelines. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2009.

[98] W. Haga and M. Zviran. Question-and-answer passwords: An empirical evaluation. *Information Systems*, 16(3), 1991.

[99] J. Halderman, B. Waters, and E. Felten. A convenient method for securely managing passwords. In *14th International World Wide Web Conference (WWW)*, 2005.

[100] E. Hammer. Introducing OAuth 2.0, May 2010. `http://hueniverse.com/2010/05/introducing-oauth-2-0/`.

[101] E. Hammer-Lahav. The OAuth 1.0 protocol. Technical Report RFC 5849, IETF, April 2010. `http://tools.ietf.org/html/rfc5849`.

[102] E. Hammer-Lahav, D. Recordon, and D. Hardt. The OAuth 2.0 authorization protocol draft. Technical Report Draft 25, IETF, March 2012. `http://tools.ietf.org/html/draft-ietf-oauth-v2-25`.

[103] C. Herley and P.C. van Oorschot. A research agenda acknowledging the persistence of passwords. *IEEE Security & Privacy*, January-February 2012.

[104] C. Herley, P.C. van Oorschot, and A. Patrick. Passwords: If we're so smart, why are we still using them? In *International Conference on Financial Cryptography and Data Security (FC)*. Springer, February 2009.

[105] M. Hlywa, A. Patrick, and R. Biddle. Facing the facts about image type in recognition-based graphical passwords. In *Annual Computer Security Applications Conference (ACSAC)*. IEEE, 2011.

[106] H.S. Al-Sinani and C. Mitchell. Implementing passcard - a cardspace-based password manager. Technical Report RHUL-MA-2010-15, Royal Holloway, University of London, December 2010.

[107] H.S. Al-Sinani and C. Mitchell. Using cardspace as a password manager. In *IFIP Policies and Research in Identity Management (IDMAN)*. Springer, 2010.

[108] P. Inglesant and M. Sasse. The true cost of unusable password policies: Password use in the wild. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2010.

[109] Internet Engineering Task Force (IETF). Request for comments. `www.ietf.org/rfc.html`, accessed July 2012.

[110] D. Jablon. Strong password-only encrypted key exchange. *SIGCOMM Computer Communication Review*, 26(5), October 1996.

[111] R. Jacob and K. Karn. Eye tracking in human-computer interaction and usability research: Ready to deliver the promises. In J. Hyona, R. Radach, and H. Deubel, editors, *The Mind's Eye: Cognitive and Applied Aspects of Eye Movement Research*, chapter 4 commentary, pages 573–605. Elsevier Science, 2003.

[112] A. Jain, A. Ross, and S. Pankanti. Biometrics: A tool for information security. *Transactions on Information Forensics and Security*, 1(2), June 2006.

[113] I. Jermyn, A. Mayer, F. Monrose, M. Reiter, and A. Rubin. The design and analysis of graphical passwords. In *USENIX Security Symposium*, 1999.

[114] S. Jeyaraman and U. Topkara. Have the cake and eat it too - infusing usability into text-password based authentication systems. In *Annual Computer Security Applications Conference (ACSAC)*. IEEE, 2005.

[115] M. Just. Designing secure yet usable credential recovery systems with challenge questions. In *CHI 2003 Workshop on Human-Computer Interaction and Security Systems*, 2003.

[116] M. Just. Designing and evaluating challenge-question systems. *Security & Privacy*, 2(5), 2010.

[117] M. Just and D. Aspinall. Personal choice and challenge questions: A security and usability assessment. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2009.

[118] M. Just and D. Aspinall. Challenging challenge questions: An experimental analysis of authentication technologies and user behaviour. *Policy & Internet*, 2(1), 2010.

[119] M. Keith, B. Shao, and P. Steinbart. The usability of passphrases for authentication: An empirical field study. *International Journal of Human-Computer Studies*, 65(1), 2007.

[120] P. Kelley, S. Komanduri, R. Shay, M. Mazurek, T. Vidas, L. Bauer, N. Christin, L. Cranor, and J. Lopez. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In *Symposium on Security and Privacy*.

[121] W. Khalifa, A. Salem, M. Roushdy, and K. Revett. A survey of EEG based user authentication schemes. In *Conference on Informatics and Systems (INFOS)*. IEEE, 2012.

[122] W. Kintsch. Models for free recall and recognition. In D. Norman, editor, *Models of human memory*, chapter Models for free recall and recognition. Academic Press, 1970.

[123] A. Kitter, E. Chi, and B. Suh. Crowdsourcing user studies with Mechanical Turk. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, April 2008.

[124] D. Klein. Foiling the cracker: A survey of, and improvements to, password security. In *USENIX Security Workshop*, 1990.

[125] S. Komanduri and D. Hutchings. Order and entropy in picture passwords. In *Graphics Interface Conference (GI)*. ACM, May 2008.

[126] S. Komanduri, R. Shay, P. Kelley, M. Mazurek, L. Bauer, N. Christin, L. Cranor, and S. Egelman. Of passwords and people: Measuring the effect of password-composition policies. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2011.

[127] M. Kumar, T. Garfinkel, D. Boneh, and T. Winograd. Reducing shoulder-surfing by using gaze-based password entry. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2007.

[128] M. Kumar, J. Klingner, R. Puranik, T. Winograd, and A. Paepcke. Improving the accuracy of gaze input for interaction. In *Eye Tracking Research & Applications Symposium (ETRA)*. ACM, 2008.

[129] C. Kuo, S. Romanosky, and L. Cranor. Human selection of mnemonic phrase-based passwords. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2006.

[130] L. Lamport. Password authentication with insecure communication. *Communications of the ACM*, 24(11), November 1981.

[131] LastPass.com. LastPass, accessed January 2012. https://lastpass.com/.

[132] Legislative Assembly of Ontario, editor. *Accessibility for Ontarians with Disabilities Act*, chapter 11. Statues of Ontario. Queen's Printer for Ontario, 2005. http://www.e-laws.gov.on.ca/html/statutes/english/elaws_statutes_05a11_e.htm.

[133] D. Lieberman. *Human Learning and Memory*. Cambridge University Press, 2012.

[134] C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[135] Microsoft Corporation. Windows CardSpace, accessed January 2012. http://www.microsoft.com/windows/products/winfamily/cardspace/default.mspx.

[136] Microsoft Corporation (Identity and Access Team)). Beyond Windows CardSpace, accessed January 2012. http://blogs.msdn.com/b/card/archive/2011/02/15/beyond-windows-cardspace.aspx.

[137] S. Microsystems. Unified login with pluggable authentication modules (PAM), October 1995. http://www.kernel.org/pub/linux/libs/pam/pre/doc/rfc86.0.txt.gz.

[138] G. Miller. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological Review*, 63(2), 1956.

[139] W. Moncur and G. Leplâtre. Pictures at the ATM: Exploring the usability of multiple graphical passwords. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, April 2007.

[140] R. Morris and K. Thompson. Password security: A case history. *Communications of the ACM*, 22, 1979.

[141] S. Murdoch and R. Anderson. Verified by Visa and MasterCard SecureCode: Or, how not to design authentication. In *International Conference on Financial Cryptography and Data Security (FC)*. Springer, 2010.

[142] D. Nelson, V. Reed, and J. Walling. Pictorial superiority effect. *Journal of Experimental Psychology: Human Learning and Memory*, 2(5), 1976.

[143] C. Neuman, T. Yu, S. Hartman, and K. Raeburn. The Kerberos network authentication service (v5). Technical Report RFC 4120, IETF, July 2005. http://tools.ietf.org/html/rfc4120.

[144] Nuance Communications Inc. Dragon NaturallySpeaking, accessed May 2012. http://www.nuance.com/dragon/index.htm.

[145] Object Management Group Inc. Documents associated with UML version 2.4.1, August 2011. http://www.omg.org/spec/UML/2.4.1/.

[146] L. O'Gorman. Comparing passwords, tokens, and biometrics for user authentication. *Proceedings of the IEEE*, December 2003.

[147] OneID Inc. OneID website, accessed April 2012. http://www.oneid.com/.

[148] OpenID Foundation. OpenID authentication 2.0, accessed January 2012. http://openid.net/specs/openid-authentication-2_0.html.

[149] OpenID Foundation. OpenID website, accessed January 2012. http://openid.net.

[150] Organization for the Advancement of Structured Information Standards (OASIS). Extensible resource identifier (XRI) resolution version 2.0, February 2008. http://docs.oasis-open.org/xri/2.0/specs/xri-resolution-V2.0.html.

[151] A. Oulasvirta and P. Saariluoma. Long-term working memory and interrupting messages in human-computer interaction. *Behaviour & Information Technology*, 23(1), 2004.

[152] Outpost9.com. Word lists, accessed January 2012. http://www.outpost9.com/files/WordLists.html.

[153] A. Paivio. *Mind and its evolution: a dual coding theoretical approach*. Lawrence Erlbaum Associates, 2007.

[154] A. Paivio, T. Rogers, and P. Smythe. Why are pictures easier to recall than words? *Psychonomic Science*, 11(4), 1968.

[155] Passfaces Corporation. The science behind Passfaces. White paper, http://www.passfaces.com/enterprise/resources/white_papers.htm.

[156] Passfaces Corporation. Passfaces online demo, December 2011. http://www.passfaces.com/demo/.

[157] A. Patrick. Monitoring corporate password sharing using social network analysis. In *International Sunbelt Social Network Conference*, 2008.

[158] PayPal. PayPal home page, accessed July 2012. http://www.paypal.com.

[159] P.C. van Oorschot and J. Thorpe. On predicting and exploiting hot-spots in click-based graphical passwords. Technical Report TR-08-21, School of Computer Science, Carleton University, November 2008.

[160] P.C. van Oorschot and J. Thorpe. On predictive models and user-drawn graphical passwords. *Transactions on Information and System Security*, 10(4), 2008.

[161] M. Peters. Revised Vandenberg & Kuse mental rotations tests: forms MRT-A to MRT-D. Technical report, Department of Psychology, University of Guelph, 1995.

[162] L. Peterson and M. Peterson. Short-term retention of individual verbal items. *Experimental Psychology*, 58(3), 1959.

[163] R. Pond, J. Podd, J. Bunnell, and R. Henderson. Word association computer passwords: The effect of formulation techniques on recall and guessing rates. *Computers & Security*, 19(7), 2000.

[164] R. Proctor, M.-C. Lien, and K.-P. Vu. Improving computer security for authentication of users: Influence of proactive password restrictions. *Behavior Research Methods, Instruments, & Computers*, 32(2), 2002.

[165] A. Rabkin. Personal knowledge questions for fallback authentication: Security questions in the era of facebook. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, July 2008.

[166] C. Rebman, M. Aiken, and C. Cegielski. Speech recognition in the human-computer interface. *Information and Management*, 40(6), 2003.

[167] D. Recordon and D. Reed. OpenID 2.0: A platform for user-centric identity management. In *Digital Identity Management Workshop*. ACM, 2006.

[168] D. Reichl. KeePass password safe website, accessed July 2012. `http://keepass.info/`.

[169] K. Renaud. Quantification of authentication mechanisms - a usability perspective. *Journal of Web Engineering*, 3(2), 2004.

[170] K. Renaud, A. McBryan, and P. Siebert. Password cueing with Cue(ink)blots. IADIS, 2008.

[171] Y. Rogers and M. Scaife. How can interactive multimedia facilitate learning? In J. Lee, editor, *Intelligence and Multimodality in Multimedia Interfaces: Research and Applications*. AAAI Press, 1998.

[172] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J. Mitchell. Stronger password authentication using browser extensions. In *USENIX Security Symposium*, 2005.

[173] V. Roth, K. Richter, and R. Freidinger. A PIN-entry method resiliant against shoulder surfing. In *Conference on Computer and Communications Security (CCS)*. ACM, 2004.

[174] C. Rovee-Collier, H. Hayne, and M. Colombo. *The Development of Implicit and Explicit Memory*. John Benjamins Pub. Co., 2001.

[175] H. Sasamoto, N. Christin, and E. Hayashi. Undercover: Authentication usable in front of prying eyes. In *Conference on Human Factors in Computing Systems (CHI)*, Florence, Italy, April 2008. ACM.

[176] M. Sasse, S. Brostoff, and D. Weirich. Transforming the 'weakest link' - a human/computer interaction approach to usable and effective security. *BT Technology*, 19(3), July 2001.

[177] S. Schechter, A. Brush, and S. Egelman. It's no secret. Measuring the security and reliability of authentication via 'secret' questions. In *IEEE Symposium on Security and Privacy*, May 2009.

[178] S. Schechter, S. Egelman, and R. Reeder. It's not what you know, but who you know: A social approach to last-resort authentication. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, April 2009.

[179] S. Schechter, C. Herley, and M. Mitzenmacher. Popularity is everything: A new approach to protecting passwords from statistical-guessing attacks. In *Workshop on Hot Topics in Security (HotSec)*. USENIX, 2010.

[180] B. Schneier. Password Safe, February 2012. `http://www.schneier.com/passsafe.html`.

[181] C. Shannon. Prediction and entropy of printed english. *Bell System Technical Journal*, 30(1), 1951.

[182] R. Shay, P. Kelley, S. Komanduri, M. Mazurek, B. Ur, T. Vidas, L. Bauer, N. Christin, and L. Cranor. Correct horse battery staple: Exploring the usability of system-assigned passphrases. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2011.

[183] R. Shay, S. Komanduri, P. Kelley, P. Leon, M. Mazurek, L. Bauer, N. Christin, and L. Cranor. Encountering stronger password requirements: User attitudes and behaviours. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2010.

[184] S. Singh, A. Cabraal, C. Demosthenous, G. Astbrink, and M. Furlong. Password sharing: Implications for security design based on social practice. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2007.

[185] N. Sofer. WebBrowserPassView - recover lost passwords stored in your web browser, accessed January 2012. `http://www.nirsoft.net/utils/web_browser_password.html`.

[186] L. St. Clair, L. Johansen, W. Enck, M. Pirretti, P. Traynor, P. McDaniel, and T. Jaeger. Password exhaustion: Predicting the end of password usefulness. In *2nd International Conference on Information Systems Security*. Springer, December 2006.

[187] L. Standing, J. Conezio, and R. Haber. Perception and memory for pictures: Single-trial learning of 2500 visual stimuli. *Psychonomic Science*, 19(2), 1970.

[188] E. Stobert. Memorability of assigned random graphical passwords. Master's thesis, Carleton University, 2011.

[189] E. Stobert and R. Biddle. Memorability and usability of graphical password forms: A graphical password bake-off. In *Annual Computer Security Applications Conference (ACSAC)*. IEEE, 2012. in submission.

[190] E. Stobert, S. Chiasson, and R. Biddle. User-choice patterns in PassTiles graphical passwords. In *Annual Computer Security Applications Conference (ACSAC)*. IEEE, 2011.

[191] E. Stobert, A. Forget, S. Chiasson, P.C. van Oorschot, and R. Biddle. Exploring usability effects of increased security in click-based graphical passwords. In *Annual Computer Security Applications Conference (ACSAC)*. IEEE, 2010.

[192] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna. Your botnet is my botnet: Analysis of a botnet takeover. In *16th Conference on Computer and Communications Security (CCS)*. ACM, 2009.

[193] A. Stubblefield and D. Simon. Inkblot authentication, msr-tr-2004-85. Technical report, Microsoft Research, Microsoft Corporation, 2004.

[194] S.-T. Sun, Y. Boshmaf, K. Hawkey, and K. Beznosov. A billion keys, but few locks: The crisis of web single sign-on. In *New Security Paradigms Workshop (NSPW)*. ACM, 2010.

[195] S.-T. Sun, K. Hawkey, and K. Beznosov. Systematically breaking and fixing OpenID security: Formal analysis, semi-automated empirical evaluation, and practical countermeasures. *Computers & Security*, 31, 2012.

[196] F. Tari, A. Ozok, and S. Holden. A comparison of perceived and real shoulder-surfing risks between alphanumeric and graphical passwords. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, July 2006.

[197] N. Tempelman-Kluit. Multimedia learning theories and online instruction. *College & Research Libraries*, 67, July 2006.

[198] J. Thorpe, P.C. van Oorschot, and A. Somayaji. Pass-thoughts: Authenticating with our minds. In *New Security Paradigms Workshop (NSPW)*. ACM, 2005.

[199] U. Topkara, M. Topkara, and M. Atallah. Passwords for everyone: Secure mnemonic-based accessible authentication. In *USENIX Annual Technical Conference (ATC)*, 2007.

[200] H. Touyama and M. Hirose. The use of photo retrieval for EEG-based personal identification. In *Asia-Pacific Conference on Computer-Human Interaction (APCHI)*. Springer, 2008.

[201] E. Tulving and W. Donaldson. *Organization of Memory*. Academic Press, 1972.

[202] E. Tulving and Z. Pearlstone. Availability versus accessibility of information in memory for words. *Journal of Verbal Learning and Verbal Behavior*, 5, 1966.

[203] E. Tulving, D. Schacter, and H. Stark. Priming effects in word fragment completion are independent of recognition memory. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 8(4), 1982.

[204] E. Tulving and M. Watkins. Continuity between recall and recognition. *American Journal of Psychology*, 86(4), 1973.

[205] United States Congress, editor. *Americans with Disabilities Act*, chapter 12101. Number 42. Office of the Law Revision Counsel of the House of Representatives, 1990. `http://www.ada.gov/`.

[206] B. Ur, P. Kelley, S. Komanduri, J. Lee, M. Maass, M. Mazurek, T. Passaro, R. Shay, T. Vidas, L. Bauer, N. Christin, and L. Cranor. How does your password measure up? the effect of strength meters on password creation. In *USENIX Security Symposium*, 2012.

[207] R. Vertegaal. A Fitts' Law comparison of eye tracking and manual input in the selection of visual targets. In *10th International Conference on Multimodal Interfaces (ICMI)*. ACM, 2008.

[208] K.-P. Vu, R. Proctor, A. Bhargav-Spantzel, B.-L. Tai, J. Cook, and E. Schultz. Improving password security and memorability to protect personal and organizational information. *International Journal of Human-Computer Studies*, 65(8), 2007.

[209] R. Weber. The statistical security of GrIDsure. Technical report, University of Cambridge, 2006. `http://www.gridsure.com/uploads/Stats%20report%20-%20Richard%20Weber.pdf`.

[210] D. Weinshall, K. Bowers, M. van Dijk, and A. Juels. Exploring implicit memory for painless password recovery. In *Conference on Human Factors in Computing Systems (CHI)*. ACM, 2011.

[211] D. Weinshall and S. Kirkpatrick. Passwords you'll never forget, but can't recall. In *Conference on Human Factors in Computing Systems (CHI) Extended Abstracts*. ACM, 2004.

[212] M. Weir, S. Aggarwal, M. Collins, and H. Stern. Testing metrics for password creation policies by attacking large sets of revealed passwords. In *17th Conference on Computer and Communications Security (CCS)*. ACM, 2010.

[213] D. Weirich and M. Sasse. Pretty good persuasion: A first step towards effective password security in the real world. In *New Security Paradigms Workshop (NSPW)*. ACM, 2001.

[214] A. Whitten and J. Tygar. Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In *USENIX Security Symposium*, 1999.

[215] S. Wiedenbeck, J. Waters, J. Birget, A. Brodskiy, and N. Memon. Authentication using graphical passwords: Basic results. In *International Conference on Human-Computer Interaction*. Lawrence Erlbaum Associates.

[216] S. Wiedenbeck, J. Waters, J. Birget, A. Brodskiy, and N. Memon. PassPoints: Design and longitudinal evaluation of a graphical password system. *International Journal of Human-Computer Studies*, 63(1-2), 2005.

[217] S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon. Authentication using graphical passwords: Effects of tolerance and image choice. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2005.

[218] S. Wiedenbeck, J. Waters, L. Sobrado, and J. Birget. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In *International Working Conference on Advanced Visual Interfaces (AVI)*, 2006.

[219] Wikipedia. Leet, accessed December 2008. `http://en.wikipedia.org/wiki/Leet`.

[220] M. Wilkes. *Time-Sharing Computer Systems*. American Elseviser, 1968.

[221] World Wide Web Consortium (W3C). Web accessibility initiative (WAI), accessed May 2012. `http://www.w3.org/WAI/`.

[222] N. Wright, A. Patrick, and R. Biddle. Do you see your password? applying recognition to textual passwords. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2012.

[223] T. Wu. The SRP authentication and key exchange system. Technical Report RFC 2945, IETF, September 2000. `http://tools.ietf.org/html/rfc2945`.

[224] J. Yan, A. Blackwell, R. Anderson, and A. Grant. Password memorability and security: Empirical results. *Security & Privacy*, 2(5), 2004.

[225] Q. Yan, J. Han, Y. Li, and R. Deng. On limitations of designing leakage-resilient password systems: Attacks, principles and usability. In *Network & Distributed System Security Symposium (NDSS)*. USENIX, 2012.

[226] S. Yardi, N. Feamster, and A. Bruckman. Photo-based authentication using social networks. In *Proceedings of the 1st Workshop on Online Social Networks*. ACM, 2008.

[227] N. Zakaria, D. Griffiths, S. Brostoff, and J. Yan. Shoulder surfing defence for recall-based graphical passwords. In *Symposium on Usable Privacy and Security (SOUPS)*. ACM, 2011.

[228] M. Zviran and W. Haga. A comparison of password techniques for multilevel authentication mechanisms. *The Computer Journal*, 36(3), 1993.

Appendix A

Persuasive Text Passwords User Study Materials

This appendix contains the materials used for Persuasive Text Password study. During the course of the user study, each participant was asked to complete the following documents:

1. A 2-page consent form was first given to participants to read and sign.
2. A 2-page participant information (demographics) questionnaire was completed by participants before beginning the experiment.
3. A 5-page post-test questionnaire was answered by participants before the end of the study.
4. A 2-page debriefing form was given to participants at the end of the study.

Consent Form – Usability Test

Research Personnel

Alain Forget Principle Investigator Carleton University (613) 520-2600 Ext. 6628 aforget@scs.carleton.ca	Dr. Robert Biddle Faculty Sponsor Carleton University (613) 520-2600 Ext. 6317 robert_biddle@carleton.ca

Purpose

The purpose of this usability test is to evaluate the usability, security, and effectiveness of enhancements to text-based passwords. We are trying to determine whether these enhancements are viable for standard passwords.

Task Requirements

We will ask you to complete a series of tasks that involve choosing and entering text-based passwords. We will then ask that you fill out a short questionnaire, and at the conclusion of the testing session you will be asked to provide us with any suggestions you might have to improve the software.

Duration and Locale

Each session should take approximately 1 hour. You will be either paid a $10 honorarium or given course credit for your time. The testing will primarily take place at the HOTLab located in room 210 SSRB.

Potential Risk/Discomfort

There will be no psychological or physical risk.

Anonymity/Confidentiality

All data that is collected will be held completely confidential. The data will only be made available to those people involved with this testing. Data will be coded for identification purposes. Sessions may be audio and video taped. If the session is to be recorded, you will be asked to sign a separate consent form indicating your agreement.

Right to Withdraw

You have the right to withdraw at any time, without any explanation as to the reason for withdrawing from the testing. You will receive the $10 honorarium or course credit even if you choose to withdraw from the study.

If you have concerns about the ethics of this research, please contact Dr. Avi Parush. For other questions about the research, please contact Dr. Anne Bowker:

Dr. Anne Bowker Chair, Department of Psychology Carleton University (613) 520 2600 ext 2648 psychchair@carleton.ca	Dr. Avi Parush Chair, Carleton University Ethics Committee for Psychological Research Carleton University (613) 520 2600 ext 6026 avi_parush@carleton.ca

Signatures
I have read and understand the above terms of testing and I understand the conditions of my participation. My signature indicates that I agree to participate in this experiment.

Participant's Name: _____

Participant's Signature: _____

Researcher's Name: _____

Researcher's Signature: _____

Participant Information – Usability Test

This information will be held completely confidential.
(Please, do not put your name on this form!)

Age: _____ years

Sex: (check one) ☐ Male ☐ Female

At what level are you studying?

☐ Undergraduate ☐ Masters ☐ Ph.D ☐ Other

What year of study are you in? _____

In what academic program are you enrolled?

On a scale of 1 (novice) to 10 (expert), how would you rate yourself with respect to your computer skills?

Novice									Expert
1	2	3	4	5	6	7	8	9	10

How often do you browse the web?

☐ Daily ☐ Several times a week ☐ Once a week ☐ Less than once a week

Approximately how many web sites do you visit that require a username and password? (Please answer with a number)

Do you sometimes re-use the same password on different web sites?

☐ Yes ☐ No

How do you decide if a web site is secure?

What criteria do you use for choosing a password? (Select more than one if appropriate)

☐ It is easy for you to remember
☐ It is suggested by the system
☐ It is difficult for others to guess
☐ It is the same as another password you currently have
☐ Other (please specify):

How concerned are you about the security of your passwords?

Not at all concerned									Very concerned
1	2	3	4	5	6	7	8	9	10

If you had to create a new password for your bank account using a normal password system, how would you go about choosing a new password?

Post-Test Questionnaire

If you had to create a new password for your bank account using a normal password system, how would you go about choosing a new password?

1. PTP passwords are easier for other people to guess than normal passwords.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

2. I had a hard time logging in with PTP

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

3. I could easily create a password with PTP

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

4. It would be easy for other people to guess a PTP password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

5. I would avoid using the PTP password scheme if I could use normal passwords instead

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

6. The PTP symbols did not help remind me of my password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

7. My accounts would be secure if protected by the PTP password scheme

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

8. If I was in a hurry, I would rather enter a normal password than a PTP password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

9. Creating PTP passwords was difficult

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

10. PTP helps me create more secure passwords.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

11. I would not be able to remember my PTP password if I did not use for a few weeks

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

12. It is easier to create PTP passwords than normal passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

13. The PTP symbols helped me figure out where I had made a mistake in my password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

14. Normal passwords are more secure than the PTP passwords scheme

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

15. PTP passwords are easier to remember than normal passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

16. Logging on with a PTP passwords was easier than with normal passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

17. PTP passwords would be hard for other people to guess.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

18. It is harder to remember PTP passwords than it is to remember normal passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

19. Logging in regularly with a PTP password would be too time-consuming

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

20. PTP makes passwords harder to create than normal passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

21. The passwords I create with PTP are more secure than normal passwords.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

22. PTP does not help me create more secure passwords.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

23. If I didn't log in to my account for a few weeks, I would still remember my PTP password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

24. It is quicker to login with PTP passwords than normal passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

25. I would not trust the PTP password scheme to protect my accounts

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

26. Given a choice between the PTP password scheme and a normal password system, I would use the PTP password scheme.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

27. The PTP symbols helped me remember the next character to type when re-entering my password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

28. It was easy to login with PTP

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

29. The PTP password scheme helps me create passwords that are more secure than normal passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

30. It is harder for other people to guess PTP passwords than it is to guess normal passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

31. It was harder to login with a PTP password than with a normal password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

32. If I made a mistake while confirming or logging in with my password, the PTP symbols did not help me notice or find the mistake

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

33. With practice, I could quickly login with a PTP password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

34. My PTP passwords are not any more secure than ones I create with normal password systems.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

What strategy did you use to create PTP passwords?

Did using PTP teach you anything about how to create more secure passwords? If so, what did you learn?

Do you think you would be able to apply any of the PTP strategies to normal password systems? If so, which strategies, and why or why not?

Would you apply any PTP strategies when using normal password systems? Why or why not?

Do you have any suggestions on improving the PTP password scheme?

Debriefing Form - Usability Test

The research we are conducting is part of a larger study examining the usability, practicality, and security of persuasive cued text-based password enhancements. These usability studies aim to assess the effectiveness of persuasion as a means to assist, motivate, guide, and teach users to create more secure passwords, as well as memorability cues to help users remember their passwords. You will find a list of the persuasive mechanisms we are testing on the back of this debriefing form.

The usability test was designed to identify problems with using these enhancements.

The results of the usability test will be used to evaluate the practicality of using persuasive cueing with text-based passwords and to make recommendations on how they can be improved. Your thoughts, comments, and opinions will be taken into consideration in making design recommendations. Thank you for participating in this usability test. Your time and effort are greatly appreciated!

If you have any further questions regarding this research, please contact:

Alain Forget Principle Investigator Carleton University (613) 520-2600 Ext. 6628 aforget@scs.carleton.ca	Dr. Robert Biddle Faculty Sponsor Carleton University (613) 520-2600 Ext. 6317 robert_biddle@carleton.ca

If you have concerns about the ethics of this research, please contact:

Dr. Anne Bowker Chair, Department of Psychology Carleton University (613) 520 2600 ext 2648 psychchair@carleton.ca	Dr. Avi Parush Chair, Carleton University Ethics Committee for Psychological Research Carleton University (613) 520 2600 ext 6026 avi_parush@carleton.ca

Persuasive Enhancements and Cues

The various persuasive security enhancements and cues that we are testing are as follows:

1. Users are allowed to choose whichever characters they want when creating a password (as usual), except for some number of characters, which are system-determined, randomly chosen and randomly positioned (e.g. _ 4 _ _ X _ # _). Users may press a Shuffle button which will prompt the system to randomly choose new characters and positions.

2. Users are allowed to choose whichever characters they want when creating a password (as per typical password systems). After they have entered their password, the system will insert characters, which are again system-determined, randomly chosen and randomly positioned (e.g. "fluffy" becomes "fl3uf@fDy"). A Shuffle button is available, which upon pressing it will prompt the system to randomly choose new characters and positions.

   2.1. A variant of [2] would be to replace characters instead of insert (e.g. "sandwich" becomes "sa8dXic%").

3. As an extension to either [1] or [2], upon confirming or logging in, as the users attempt to re-type their password, we could display some cueing symbol above characters which were system-imposed, to cue the user. For example, if the password I created was "fl3uf@fDy", upon typing "fl", I would then see the cueing symbol above the next character I was to type, notifying me that the next character is was system-assigned. So in the above password example of "fl3uf@fDy", this I would see the cueing symbol when I have typed "fl", "fl3uf", and "fl3uf@f". For character combinations not contained in the user's password, the system may or may not display the cueing symbol, so as to not give any of the users' password information to potential attackers.

4. Another extension of [1] (and possibly [2]) could entail mapping each of the 95 US English keyboard-typeable characters to some unique UNICODE symbol, and showing each of these symbols directly below each character as it is typed (when creating, confirming and logging in). These symbols would act as a memory cue to help users recognise whether or not they're typing their password correctly.

   4.1. A variant of [4] could use thumbnail images instead of UNICODE symbols.

   4.2. Another variant of [4] and [4.1] would be to show the symbols or images one at a time, whereupon at password creation, for each symbol/image, users would be asked to enter either a character of their choice or a system-imposed random character (or shuffling until they find a symbol/image & character combination they like). When confirming and/or logging in, they would be prompted by the symbol/image of their first character, and would proceed to enter their password, as the symbol/image cues change as they type, to assist them in remembering the following character in their password.

Appendix B

Cued Gaze-Points User Study Materials

This appendix contains the materials used for Cued Gaze-Points study. During the course of the user study, each participant was asked to complete the following documents:

1. A 2-page consent form was first given to participants to read and sign.
2. A 3-page participant information (demographics) questionnaire was completed by participants before beginning the experiment.
3. A 4-page post-test questionnaire was answered by participants before the end of the study.
4. A 1-page debriefing form was given to participants at the end of the study.

Consent Form – Graphical Passwords Usability Test

Research Personnel

Alain Forget	Dr. Robert Biddle
Principle Investigator	Faculty Sponsor
Carleton University	Carleton University
(613) 520-2600 Ext. 6628	(613) 520-2600 Ext. 6317
aforget@scs.carleton.ca	robert_biddle@carleton.ca

Purpose

The purpose of this usability test is to evaluate the usability, security, and effectiveness of graphical passwords. We are trying to determine whether graphical passwords are a viable alternative to regular passwords. We are also investigating the use of an eye tracker as an input device instead of a mouse. An eye tracker will be detecting eye movements during the session.

Task Requirements

We will ask you to complete a series of tasks that involve choosing and entering graphical passwords. We will then ask that you fill out a short questionnaire, and at the conclusion of the testing session you will be asked to provide us with any suggestions you might have to improve the software.

Duration and Locale

Each session should take approximately 1 hour. You will be paid a $10 honorarium OR receive course credit for your time. The testing will primarily take place at the HOTLab located in room 210 SSRB.

Potential Risk/Discomfort

You may experience temporary eye discomfort throughout the study. This is an unlikely occurrence, and the risk is no worse than working at a computer for several hours. If you experience discomfort, you may take breaks as needed or withdraw from the experiment.

Anonymity/Confidentiality

All data that is collected will be held completely confidential. The data will only be made available to those people involved with this testing. Data will be coded for identification purposes.

Right to Withdraw

You have the right to withdraw at any time, without any explanation as to the reason for withdrawing from the testing. You will receive the $10 honorarium or course credit even if you choose to withdraw from the study.

If you have concerns about the ethics of this research, please contact Dr. Avi Parush. For other questions about the research, please contact Dr. Janet Mantler:

Dr. Janet Mantler Chair, Department of Psychology Carleton University (613) 520 2600 ext 4173 Janet_Mantler@carleton.ca	Dr. Avi Parush Chair, Carleton University Ethics Committee for Psychological Research Carleton University (613) 520 2600 ext 4371 Avi_Parush@carleton.ca

Signatures
I have read and understand the above terms of testing and I understand the conditions of my participation. My signature indicates that I agree to participate in this experiment.

Participant's Name: _____

Participant's Signature: _____

Researcher's Name: _____

Researcher's Signature: _____

Date: _____

Participant Information – Usability Test

This information will be held completely confidential.
(Please, do not put your name on this form!)

Age: _____ years

Sex: (check one) ☐ male ☐ female

At what level are you studying?

☐ Undergraduate ☐ Masters ☐ Ph.D. ☐ Other

What year of study are you in? _____

In what academic program are you enrolled?

Before this user study, had you ever used a graphical password (using a picture to enter a password instead of typing in letters and numbers)? If so, where?

Do you wear corrective lenses (glasses or contacts) or have you had corrective eye surgery (laser or otherwise)? (Select more than one if appropriate)

☐ Glasses ☐ Contact Lenses

☐ Laser eye surgery ☐ Other (please specify):

On a scale of 1 (novice) to 10 (expert), how would you rate yourself with respect to your computer skills?

Novice									Expert
1	2	3	4	5	6	7	8	9	10

How often do you browse the web?

☐ Daily ☐ Several times a week ☐ Once a week ☐ Less than once a week

Approximately how many web sites do you visit that require a username and password? (Please answer with a number)

Do you sometimes re-use the same password on different web sites?

☐ Yes ☐ No

What criteria do you use for choosing a password? (Select more than one if appropriate)

☐ It is easy for you to remember
☐ It is suggested by the system
☐ It is difficult for others to guess
☐ It is the same as another password you currently have
☐ Other (please specify):

How concerned are you about the security of your passwords?

Not at all concerned									Very concerned
1	2	3	4	5	6	7	8	9	10

How do you decide if a web site is secure?

If you had to create a new password for your bank account using a normal password system, how would you go about choosing a new password?

Graphical Passwords Post-Task Questionnaire

1. I could easily create a graphical password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

2. It was easy to decide what points to choose as part of my password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

3. It is easier to create text passwords than graphical passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

4. With practice, I could quickly enter graphical passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

5. I would trust a graphical password to protect my financial information.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

6. My eyes are tired from choosing points on images

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

7. Graphical passwords are quicker to use than text-based passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

8. Logging on using a graphical password was easy

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

9. It was easy to gaze at the right points for my password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

10. I would use a graphical password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

11. If I didn't log in to my account for a few weeks, I would still remember my graphical password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

12. It would be easier to remember 5 different text passwords than 5 different graphical passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

13. I like using my eyes to select points on the screen

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

14. Given the choice between a text password and a graphical password, I would choose a graphical password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

15. My accounts would be secure if protected by a graphical password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

16. Graphical passwords would be easy for attackers to guess

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

17. Logging on using a graphical password was easier than with a text password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

18. Text passwords are more secure than graphical passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

19. I prefer text passwords to graphical passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

20. I am uncomfortable with computers that track where I am looking on the screen

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

21. It was easy to select a good graphical password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

22. Graphical passwords are easy to remember

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

23. I think using eye gaze as input is safer than typing or using the mouse

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

24. It was difficult to gaze at exactly the right points when entering my graphical password even though I knew my password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

25. Everyone would select different graphical passwords for the same images

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

26. Graphical passwords are too time-consuming

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

27. If I was in a hurry, I would rather enter a text-based password than a graphical password

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

28. I would be happy if computer systems used graphical passwords instead of text passwords

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

What strategy did you use for selecting your graphical passwords?

How would your strategy change if you had several graphical passwords to remember?

How would your strategy change if you used the mouse to select points, instead of your eyes?

Which images were most helpful in selecting easy-to-remember passwords? Why?

If you could pick your own images, what would you choose? Why?

Do you have any suggestions on improving this eye-gaze graphical password scheme?

Debriefing Form – Graphical Password Usability Test

The research we are conducting is part of a larger study examining the usability, practicality, and security of graphical passwords. These usability studies aim to assess the effectiveness of graphical passwords as alternatives to text-based passwords for systems requiring user authentication. In this particular study, we were also investigating whether an eye tracker could be used as an input device for password entry.

The usability test was designed to identify problems with using these programs.

The results of the usability test will be used to evaluate the practicality of graphical passwords and to make recommendations on how they can be improved. Your thoughts, comments, and opinions will be taken into consideration in making design recommendations. Thank you for participating in this usability test. Your time and effort are greatly appreciated!

There is a slight chance that you experienced temporary mild eye strain from using the eye tracker and focusing on the computer screen. This risk is no worse than ordinary extended computer use. Your eyes should feel better within a few minutes of completing the study. If you would like further information about computer eye strain and steps that can be taken to alleviate these symptoms, please visit: http://hotsoft.carleton.ca/eyestrain

If you have any further questions regarding this research, please contact:

Alain Forget Principle Investigator Carleton University (613) 520-2600 Ext. 6628 aforget@scs.carleton.ca	Dr. Robert Biddle Faculty Sponsor Carleton University (613) 520-2600 Ext. 6317 robert_biddle@carleton.ca

If you have concerns about the ethics of this research, please contact:

Dr. Janet Mantler Chair, Department of Psychology Carleton University (613) 520 2600 ext 4173 Janet_Mantler@carleton.ca	Dr. Avi Parush Chair, Carleton University Ethics Committee for Psychological Research Carleton University (613) 520 2600 ext 4371 Avi_Parush@carleton.ca

Appendix C

Learnability User Study Materials

This appendix contains the materials used for the Authentication Scheme Learnability study. During the course of the user study, each participant was asked to complete the following documents:

1. A 2-page consent form was first given to participants to read and sign.
2. A 3-page pre-test questionnaire was completed by participants before beginning the experiment.
3. A 1-page demographics questionnaire was also filled out by participants before the end of the first (*day 0*) session.
4. A 1-page interim debriefing form was provided to participants at the end of the first (*day 0*) session.
5. A 7-page post-test questionnaire was answered by participants before the end of the last (*day 7*) session.
6. A 1-page debriefing form was given to participants at the end of the last (*day 7*) session.

Consent Form – Website Usability Test

The purpose of an informed consent is to ensure that you understand the purpose of the study and the nature of your involvement. This informed consent must provide sufficient information such that you have the opportunity to determine whether you wish to participate in the study.

Research Personnel

Alain Forget Principal Investigator Carleton University (613) 520-2600 Ext. 4577 aforget@scs.carleton.ca	Dr. Robert Biddle Faculty Sponsor Carleton University (613) 520-2600 Ext. 6317 robert_biddle@carleton.ca

Purpose
The purpose of this usability test is to evaluate the usability of several websites. In particular, we are investigating how easy or hard it is to log in and perform various tasks on several websites.

Task Requirements
We will ask you to log in and complete tasks on various websites. We will then ask you to fill out several questionnaires. At the conclusion of the study, you will be asked to provide us with any suggestions you might have to improve the websites.

Duration and Locale
The first session should take approximately 1 hour and the subsequent online sessions should take approximately 30 minutes in total. The second lab session will take about 30 minutes. The total duration of the study (including the online sessions) will be between one week and 6 months. You will be paid a $20 honourarium OR receive course credit for your time. You will receive your honourarium after completion of the online sessions. The lab session will primarily take place at the HotSoft lab located in room 2110 in the HCI building at Carleton University. If you choose to receive the $20 honourarium, you will be asked to return to the HotSoft lab to pick up your honourarium.

Potential Risk/Discomfort
There will be no psychological or physical risk.

Audio Recording
The lab sessions will be audio recorded to assist experimenters in gathering data about the study. Audio recordings will be destroyed once they have been transcribed.

Anonymity/Confidentiality
All data that is collected will be held completely confidential. The data will only be made available to those people involved with this testing. Data will be coded for identification purposes.

Right to Withdraw
You have the right to withdraw at any time, without any explanation as to the reason for withdrawing from the testing. You also have the right to refuse to answer any questions you do not feel comfortable answering, for any reason and without explanation. You will receive the $20 honorarium or course credit even if you choose to withdraw from the study. If you choose to stop responding to our emails, we will attempt to contact you but if we cannot contact you, we will consider you as withdrawn from the study, and will send you information on how to collect your compensation.

If you have concerns about the ethics of this research, please contact Dr. Monique Sénéchal. For other questions about the research, please contact Dr. Janet Mantler:

Dr. Janet Mantler Chair, Department of Psychology Carleton University (613) 520 2600 ext 4173 psychchair@carleton.ca	Dr. Monique Sénéchal Chair, Carleton University Ethics Committee for Psychological Research Carleton University (613) 520 2600 ext 1155 Monique_Senechal@carleton.ca

Signatures
I have read and understand the above terms of testing and I understand the conditions of my participation. My signature indicates that I agree to participate in this experiment.

Participant's Name: _____

Participant's Signature: _____

Researcher's Name: _____

Researcher's Signature: _____

Date: _____

Pre-test Questionnaire

On a scale of 1 (novice) to 10 (expert), how would you rate yourself with respect to your computer skills?

Novice									Expert
1	2	3	4	5	6	7	8	9	10

How often do you browse the web?

☐ Daily ☐ Several times a week ☐ Once a week ☐ Less than once a week

How often do you check your e-mail?

☐ Daily ☐ Several times a week ☐ Once a week ☐ Less than once a week

Approximately how many web sites do you visit that require a username and password?
(Please answer with a number)

Do you sometimes re-use the same password on different web sites?

☐ Yes ☐ No

How do you decide if a web site is secure?

What criteria do you use for choosing a password? (Select more than one if appropriate)

☐ It is easy for you to remember

☐ It is suggested by the system

☐ It is difficult for others to guess

☐ It is the same as another password you currently have

☐ Other (please specify):

How concerned are you about the security of your passwords?

Not at all concerned									Very concerned
1	2	3	4	5	6	7	8	9	10

What kind of strategies do you use to remember your passwords? (Choose all that apply)

- ☐ Write passwords down
- ☐ Use password manager
- ☐ Save passwords in browser
- ☐ Other: (please specify)

If you had to create a new password for your bank account using a normal password system, how would you go about choosing a new password?

What is a blog (weblog)?

How often do you comment on blog posts?

☐ Never ☐ Daily ☐ Weekly ☐ Monthly ☐ Yearly

How concerned are you about your privacy and security when commenting on a blog post?

Not at all concerned									Very concerned
1	2	3	4	5	6	7	8	9	10

What is an online forum or message board?

How often have you read or participated in an online forum or message board?

☐ Never ☐ Daily ☐ Weekly ☐ Monthly ☐ Yearly

How concerned are you about your privacy and security when posting on a message board?

Not at all concerned									Very concerned
1	2	3	4	5	6	7	8	9	10

What items do you buy online?

How often do you shop online?

☐ Never ☐ Daily ☐ Weekly ☐ Monthly ☐ Yearly

What is the price range of items you normally shop for? (Please check all that apply)

☐ Less than $10 ☐ $10 - $100 ☐ $100 - $500 ☐ $500 and over

How concerned are you about your privacy and security when shopping online?

Not at all concerned									Very concerned
1	2	3	4	5	6	7	8	9	10

Demographics Questionnaire

This information will be held completely confidential. **(Please, do not put your name on this form!)**

Age: _____ years

Sex: (check one) ☐ male ☐ female

At what level are you studying?

☐ Undergraduate ☐ Masters ☐ Ph.D. ☐ Other

What year of study are you in? _____

In what academic program are you enrolled?

Have you ever been in a password study before? If so, please describe the study.

Have you ever been in a website usability study before? If so, please describe the study.

Have you ever used a graphical password (using a picture to enter a password instead of typing in letters and numbers)? If so, please describe it as best you can, and tell us where you used it.

Post-test Questionnaire

Note: Participants were asked a subset of these questions, as applicable to their assigned condition. They will be reminded that the websites mentioned in the questions refer to the websites from the study.

In the following questions, the term "password" refers to the type of password you used in this study.

1. The message board was easy to use.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

2. It was difficult to move around the message board.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

3. The layout of the message board was intuitive.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

4. On the message board, the wording was hard to understand.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

5. It was easy to complete my tasks on the message board.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

6. I was able to complete my tasks quickly on the message board.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

7. On the message board, the fonts were difficult to read.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

8. It was difficult to move around the blog website.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

9. It was easy to complete my tasks on the blog.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

10. On the blog website, the fonts were difficult to read.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

11. The layout of the blog website was intuitive.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

12. I was able to complete my tasks quickly on the blog website.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

13. On the blog website, the wording was hard to understand.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

14. The blog website was easy to use.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

15. I was able to complete my tasks quickly on the shopping website.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

16. It was easy to complete my tasks on the shopping website.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

17. On the shopping website, the wording was hard to understand.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

18. On the shopping website, the fonts were difficult to read.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

19. The shopping website was easy to use.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

20. The layout of the shopping website was intuitive.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

21. It was difficult to move around the shopping website.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

22. I could easily create a password.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

23. It was easy to create my password.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

24. I would trust this password system to protect my financial information.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

25. I think that other people would choose different passwords than me.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

26. Logging in using these passwords was easy.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

27. It was easy to accurately enter my password.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

28. I would use a password of this type.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

29. If I didn't log in to my account for a few weeks, I would still remember my passwords.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

30. My accounts would be secure if protected by a password of this type.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

31. I chose a password that was particularly memorable to me.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

32. This type of password would be easy for attackers to guess.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

33. My passwords are unlikely to have any meaning to other people.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

34. It was difficult to enter my password even though I thought I remembered it.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

35. I think other people would choose the same passwords as me.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

36. This type of password is too time-consuming.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

37. With practice, I could quickly enter my passwords.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

38. I believe I could guess other people's passwords.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

39. Text passwords are more secure than this type of password.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

40. These passwords are quicker to use than text passwords.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

41. Given the choice between a text password and a password of this type, I would choose a password of this type.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

42. If I were in a hurry, I would rather enter a text password than this type of password.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

43. I prefer text passwords to this type of passwords.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

44. I would be happy if computer systems used this type of passwords instead of text passwords.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

45. Logging on using a password of this type was easier than with a text password.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

46. It would be easier to remember 5 different text passwords than 5 different passwords of this type.

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

47. How did you remember your passwords from this study?

48. Did you write any of your passwords down? (Y/N)

49. Did you use a password manager to remember any of your study passwords? (Y/N)

50. Did you save your study passwords in the browser? (Y/N)

51. What would make the passwords used in this study easier to remember?

Interim Debriefing – Website Usability Test

The purpose of this experiment is to study users accessing online systems to complete tasks. We are exploring various methods of authentication and performing typical website tasks. We are evaluating how easily users can log in and perform these tasks using the online systems. Our aim is to improve computer systems to make them more secure and easy-to-use.

More details about this study will be made available once you have finished the entire study.

If you have any further questions regarding this research, please contact:

Alain Forget Principal Investigator Carleton University (613) 520-2600 Ext. 4577 aforget@scs.carleton.ca	Dr. Robert Biddle Faculty Sponsor Carleton University (613) 520-2600 Ext. 6317 robert_biddle@carleton.ca

If you have concerns about the ethics of this research, please contact:

Dr. Janet Mantler Chair, Department of Psychology Carleton University (613) 520 2600 ext 2664 psychchair@carleton.ca	Dr. Monique Sénéchal Chair, Carleton University Ethics Committee for Psychological Research Carleton University (613) 520 2600 ext 1155 Monique_Senechal@carleton.ca

If you do not receive an e-mail from us in the *next 24 hours*, please e-mail us at:

CarletonHCIStudy@gmail.com

If you have any other questions, or if any of your email information changes, please contact us.

Please keep this form so that you remember your username and the time of your next appointment.

Username: _____

Session 2 Appointment : _____

Debriefing Form – Website Usability Test

The research we are conducting is part of a larger study examining the usability, practicality, and security of authentication schemes. These usability studies aim to assess the effectiveness of these authentication schemes as alternatives to text-based passwords for systems requiring user authentication.

The usability test was designed to identify problems with using these programs. In this study, we are studying how the password system used, the number of passwords created, the frequency of login, and the number of logins the user has to enter affects the usability and security of the system. We hypothesize increasing the frequency of logins will make it easier for users to remember their passwords, but that having more passwords will cause users to create less secure passwords.

The results of the usability test will be used to evaluate the practicality of the password systems and to make recommendations on how they can be improved. Your thoughts, comments, and opinions will be taken into consideration in making design recommendations. Thank you for participating in this usability study. Your time and effort are greatly appreciated!

If you have any further questions regarding this research, please contact:

Alain Forget Principal Investigator Carleton University (613) 520-2600 Ext. 4577 aforget@scs.carleton.ca	Dr. Robert Biddle Faculty Sponsor Carleton University (613) 520-2600 Ext. 6317 robert_biddle@carleton.ca

If you have concerns about the ethics of this research, please contact:

Dr. Janet Mantler Chair, Department of Psychology Carleton University (613) 520 2600 ext 2664 psychchair@carleton.ca	Dr. Monique Sénéchal Chair, Carleton University Ethics Committee for Psychological Research Carleton University (613) 520 2600 ext 1155 Monique_Senechal@carleton.ca