USABLE AUTHENTICATION AND CLICK-BASED GRAPHICAL PASSWORDS

by

Sonia Chiasson

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY

School of Computer Science

at

CARLETON UNIVERSITY

Ottawa, Ontario
December 2008

Your file Votre référence
ISBN: 978-0-494-47475-4
Our file Notre référence
ISBN: 978-0-494-47475-4

USABLE AUTHENTICATION AND CLICK-BASED GRAPHICAL PASSWORDS

by

Sonia Chiasson

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY

School of Computer Science

at

CARLETON UNIVERSITY

Ottawa, Ontario
December 2008

Table of Contents

List of Tables

List of Figures

Abstract

Security experts often refer to humans as the "weakest link" (Sasse, Brostoff, and Weirich, 2001) in the security chain, asserting that the problem lies not with the security systems themselves, but with users who are unable or unwilling to comply with security protocols. The shift towards usable security and including human factors in system design is an important one that has a direct impact on system security.

In this thesis, we focus on knowledge-based authentication. We examine the password problem, where passwords are either weak-and-memorable or secure-but-difficult-to-remember, despite the need for secure and memorable passwords. We concentrate on graphical passwords due to the human ability to accurately recognize and recall images. We began by cataloguing existing graphical passwords, focusing equally on usability and security characteristics, and identified PassPoints, a click-based graphical password scheme, as the scheme that appeared most promising and that we believed warranted closer evaluation. Our overall research question, therefore, asks: *"Can click-based graphical passwords simultaneously support both memorability and security, while maintaining usability?"*.

We conducted lab and field studies of PassPoints, and identified areas for usability and security improvements. We designed Cued Click-Points and Persuasive Cued Click-Points, schemes with several novel design features: one-to-one cueing to help with the memorability, implicit feedback meaningful only to legitimate users, and a safe-path-of-least-resistance influencing users to select stronger memorable passwords. Empirical studies of both schemes provide evidence of increased usability, memorability, and security. Additionally, we propose a new discretization method for such systems that improves usability by making the system more predictable from the user's perspective and improves security by allowing for smaller tolerance regions without sacrificing usability. From this empirical work, we identified the underlying design characteristics of our systems that led to success and generalized our findings as design strategies that may be applicable to other knowledge-based authentication schemes.

Acknowledgements

First and foremost, I am grateful to my supervisors Robert Biddle and Paul van Oorschot. It is due to their excellent guidance and support that this research has been possible. Their complimentary expertise in human-computer interaction and computer security proved to be the perfect combination. They have been, and continue to be, wonderful and patient mentors.

Thanks to my colleagues in the HotSoft, CCSL, and HOTLab research groups who have helped with experiments, listened to presentations, and offered valuable feedback and insight throughout the process. Special thanks to Alain Forget, with whom I have worked closely on many projects and publications over the last two years. His contributions and friendship have been invaluable in getting this dissertation completed.

Thanks to the members of my committee, Konstantin Beznosov, Timothy Lethbridge, Andrew Patrick, and Anil Somayaji for their guidance, for their expertise, and for offering different perspectives, all of which have helped shape this dissertation. I am also grateful to them for agreeing to hold my proposal and thesis defences at especially busy times of year.

To Jay, I offer many thanks and my appreciation for the tireless academic discussions, for the insight, for the proof-reading, as well as for the emotional support and understanding throughout the years.

Thanks to the several hundred participants who took part in our user studies. Their cooperation and feedback were key to this research.

My family and friends have been so incredibly understanding throughout this journey. There have been many missed special occasions, stressed phone calls, and rushed holidays over the course of this degree. Their unwavering support and confidence means a lot to me. Mom, even though there is no final grade on this thesis, you still deserve more than a few marks for all your help throughout the years.

Chapter 1

Introduction

> "Humans are incapable of securely storing high-quality cryptographic keys, and they have unacceptable speed and accuracy when performing cryptographic operations. (They are also large, expensive to maintain, difficult to manage, and they pollute the environment. It is astonishing that these devices continue to be manufactured and deployed. But they are sufficiently pervasive that we must design our protocols around their limitations.)" — C. Kaufman, R. Perlman, and M. Speciner, 2002 [62]

User authentication involves issues of both usability and security. Too often, one or the other is ignored even though both are important and necessary. This problem is evident in knowledge-based authentication systems. For example, passwords are often either memorable-but-insecure or secure-but-difficult-to-remember when they should be memorable *and* secure. Graphical passwords are potentially more memorable and secure than traditional text passwords because they harness the human ability to easily recognize and recall images. In this thesis, we advance research in the area of knowledge-based authentication through usability and security evaluations of graphical password schemes, the creation of novel schemes that offer improved memorability and security, and the identification of some underlying design strategies to inform the design of other knowledge-based authentication schemes.

1.1 Context

The field of *usable security* is a relatively new area of study combining two areas of computer science: human-computer interaction (HCI) and computer security. HCI is "a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them" [56]. Computer security is a discipline concerned with the "ability of

a system to protect information and system resources with respect to confidentiality and integrity", and is associated with several concepts: confidentiality, integrity, authentication, access control, non-repudiation, availability, and privacy [99]. Usable security, therefore, focuses on the human aspects of computer security, including both how human behaviour affects the security of a system and how the interaction design of a security system impacts its users. Many years before usable security gained widespread recognition, Saltzer and Schroeder explained:

> "It is essential that the human interface be designed for ease of use, so that users routinely and automatically apply the protection mechanisms correctly. Also, to the extent that the user's mental image of his protection goals matches the mechanisms he must use, mistakes will be minimized. If he must translate his image of his protection needs into a radically different specification language, he will make errors." — Saltzer and Schroeder, 1975 [102]

More recently, Cranor and Garfinkel [25] succinctly describe the goal of usable security as designing "secure systems that people can use."

1.2 Motivation

Computer security has traditionally focused on low-level, technical design and implementation details. Security experts often refer to humans as the "weakest link" [103] in the security chain, asserting that the problem lies not with the security systems themselves, but with users who are unable or unwilling to comply with security protocols. This approach of separating system design from user behaviour is doomed to fail because it ignores the larger context in which security systems are inevitably used.

The shift towards usable security and including human factors as part of system design is an important one that has a direct impact on the security of the system. When users misunderstand how to use security mechanisms, circumvent them because they are too obtrusive, or do not even realize the need for such systems, then the

systems are far more likely to result in overall security failures regardless of the systems' technical soundness.

People encounter security mechanisms daily, such as physical keys to unlock doors or security alarms intended to alert them of intruders. With respect to computer security mechanisms, people are most often required to authenticate themselves using knowledge-based schemes such as passwords. Even though these are commonly used, and perhaps because they are so prevalent, passwords are plagued with security and usability problems. Technical solutions such as imposing minimum password requirements, and encryption and communication algorithms, for protecting passwords in transit and storage, have not resolved the human factors problems with passwords: usability, memorability, memory interference from having multiple passwords, and predictability in user choice. The *"password problem"* has been defined [136] as the current situation where many passwords used in practice are either weak-and-memorable or secure-but-difficult-to-remember, despite the need for secure and memorable passwords.

Security and usability are often viewed by security experts as opposite extremes, and one must necessarily be sacrificed for the other. We investigate whether it is possible to increase both security and usability at the same time. In this thesis, we focus on one particular aspect of security, namely user authentication. While alternative authentication mechanisms such as biometrics [59] are widely known, these have their own security, privacy, and usability problems [22] that limit their use to specific applications. Due to their widespread usage and relatively low cost, knowledge-based schemes such as passwords are unlikely to disappear; and they may well become even more popular as more day-to-day tasks are computerized. For these reasons, we focus on improving knowledge-based authentication schemes.

Proposals for improving text passwords such as passphrases [63] or mnemonic passwords [69] have yet to deliver the desired security or usability gains. In preliminary work to this thesis, we investigated password managers [20] and these were also shown to have serious problems, at least in their present state. We next turned to graphical passwords as potentially successful knowledge-based schemes. Graphical passwords have been proposed in recent years due to their potential for improved

memorability [77,115] because of the superior human ability to recognize and remember images [65, 72, 86, 108]. However, as we discuss in Chapter 2, most graphical password schemes have not been systematically evaluated for both usability and security. We began our work with graphical passwords by conducting usability and security evaluations of PassPoints [135–137], the scheme that we felt offered the most promise among existing proposals. PassPoints exemplifies the category of *"click-based graphical passwords"*; in such schemes, passwords consist of a specific sequence of clicks on different areas of an image.

1.3 Thesis Statement

A major goal of this research is to discover how to create knowledge-based authentication schemes that are memorable, usable, and secure. We also investigate the interplay between usability and security, an issue that is not well understood in current systems.

We focused our research on click-based graphical passwords because of their potential for increased memorability and security. The main research question is:

> *Can click-based graphical passwords simultaneously support both memorability and security, while maintaining usability?*

The work began with a general investigation, with new ideas being formed and tested as we progressed with the research. Four main research objectives of this thesis are described below.

Objective 1: Catalogue existing graphical password schemes, focusing equally on usability and security characteristics, and identify the existing graphical password scheme that appears most promising and that warrants closer evaluation.

Objective 2: With respect to security and usability, empirically evaluate the most promising scheme identified through our cataloguing. (This turned out to be the PassPoints scheme.)

Objective 3: Create and empirically test new designs that address any usability and security problems identified in the scheme identified in Objective 2. (Given

that PassPoints was the identified scheme, the resulting goal ended up being to increase security and memorability of click-based graphical passwords while maintaining usability.)

Objective 4: Identify the key underlying design characteristics responsible for success of the newly proposed system(s), and generalize these to develop design strategies that can be applied to other types of knowledge-based authentication schemes.

1.4 Overview of the Thesis

The remainder of the thesis is organized as follows. The first half of Chapter 2 provides relevant background on usable security, authentication and security threats to authentication, and conducting user studies. The second half of Chapter 2 addresses *Objective 1* of Section 1.3. It surveys existing graphical password schemes, summarizes published analyses of these schemes, and provides a comparison according to selected usability and security characteristics. The chapter concludes with our rationale for further evaluation of PassPoints.

To address *Objective 2*, Chapter 3 presents our empirical studies of PassPoints. It describes our lab and field studies of PassPoints, details our analysis, and explains the usability and security problems that we discovered. Further analysis of PassPoints is provided in Chapters 5 and 7, where user choice of passwords is compared with user choice in our new schemes.

Objective 3 required design work and the creation of novel schemes, as well as analysis to determine whether our designs were effective. Chapters 4 to 8 contribute to meeting Objective 3. We present two novel graphical password schemes and a novel method for implementing click-based graphical passwords. Chapter 4 introduces Cued Click-Points (CCP), a new graphical password scheme, describes the lab study and analysis we conducted on CCP, and provides the results. It identifies the improvements over PassPoints and the areas where further work is necessary. Chapter 5 describes Persuasive Cued Click-Points (PCCP), our refined graphical password scheme. The lab study of PCCP and the results of our analysis are described. This

chapter also begins our comparison of PassPoints, CCP, and PCCP with respect to user choice in password selection, and shows that PCCP results in significantly fewer predictable passwords based on the clustering of click-points.

Chapter 6 presents "centered discretization", a method involved in translating user-entered click-points into machine-repeatable password elements, that improves the implementation of graphical passwords such as PassPoints, CCP, and PCCP. Through post-hoc analysis of the dataset from our PassPoints field study, we quantify the usability and security improvements over robust discretization [9], the corresponding method proposed by the original authors of PassPoints. This also offers the first look at how robust discretization affects usability since it was not actually implemented [12] in the prototype system used in the original PassPoints studies by Wiedenbeck et al. [135–137].

Chapter 7 offers more in-depth analysis comparing user choice within PassPoints, CCP, and PCCP. For this analysis, we conduct post-hoc analysis of the four datasets presented in Chapters 3, 4, and 5. We demonstrate the security improvements over PassPoints that arise from our design choices in CCP and PCCP.

Chapter 8 takes a broader view of security and discusses how CCP and PCCP would withstand various types of attacks. We provide comparisons to text passwords and PassPoints, where appropriate, to place our schemes in context.

Finally, Chapter 9 discusses overall design strategies that can be extracted and generalized from this research, in order to meet *Objective 4*. It also describes further research directions that fall beyond the scope of this thesis, and offers concluding remarks.

1.5 Main Contributions of this Research

This research contributes original ideas and knowledge to the field of usable security. We design and test two novel graphical password schemes and a novel algorithm for implementation of click-based graphical passwords. We conducted usability and security analysis of both a pre-existing scheme and newly proposed graphical password systems. As part of our work, we examined how design choices affect user behaviour, as well as the interaction between usability and security.

The main contributions of this research are enumerated below.

1. We reviewed existing graphical password schemes by cataloguing them according to several usability and security characteristics. We discovered that there was little consistency in the types of evaluations conducted on graphical passwords, with most evaluations focusing on either usability or security but not both. We identified the most promising scheme in terms of memorability and potential security, and decided that it was worth further evaluation.

2. We conducted two empirical user studies [15] of PassPoints, one controlled experiment conducted in the lab and one large field study where the system was deployed for real usage over several months. In our initial analysis, we show that image choice impacts the usability of PassPoints, that users are extremely accurate in entering their click-points, and that login times and success rates are generally good. In later analysis of the PassPoints datasets, we show that passwords with certain characteristics have a much higher likelihood of being chosen by users, making them vulnerable to guessing attacks.

3. We proposed Cued Click-Points [21] and Persuasive Cued Click-Points [16]. These were prototyped and evaluated with empirical user studies conducted in the lab. We show that these new schemes have usability and memorability advantages over PassPoints. They also significantly increase security with respect to known attacks by reducing the predictability of user selected passwords [17] and increasing the effort required by attackers to launch successful attacks.

4. We proposed centered discretization [19], a new method for improved implementation of click-based graphical passwords. We evaluated our method using post-hoc analysis of the empirical data collected in the PassPoints field study. Compared to the scheme proposed by the original PassPoints authors [9], centered discretization allows for smaller tolerance areas, which increases the theoretical password space, and better usability because the system behaves in a manner consistent with user expectations.

5. We extracted and generalized the main design characteristics of our new schemes

that led to significant usability and security improvements. We introduce the design strategies of *implicit feedback*, *one-to-one cueing*, *safe-path-of-least-resistance*, and *matching user expectations* with respect to knowledge-based authentication. Throughout the thesis, we demonstrate how the application of these strategies can increase the usability, memorability, and security of click-based graphical passwords.

1.6 Related Publications

Significant portions of the research presented in this thesis have been peer-reviewed and published in academic venues. I am primary author on the following papers based on work from this thesis. Much of the text in the thesis for these published portions is taken from the publications. As indicated below, parts of this work have been undertaken in collaboration with other student researchers, most notably with Alain Forget.

The peer-reviewed full-paper publications are:

S. Chiasson, P. van Oorschot, and R. Biddle. A usability study and critique of two password managers. In the proceedings of the 15th USENIX Security Symposium, August 2006.

S. Chiasson, R. Biddle, and P. van Oorschot. A second look at the usability of click-based graphical passwords. In the proceedings of the 3rd Symposium on Usable Privacy and Security (SOUPS), July 2007.

S. Chiasson, P. van Oorschot, and R. Biddle. Graphical password authentication using Cued Click Points. In the proceedings of the European Symposium On Research In Computer Security (ESORICS), LNCS 4734, pages 359-374, September 2007.

S. Chiasson, J. Srinivasan, R. Biddle, and P. van Oorschot. Centered discretization with application to graphical passwords. In the proceedings of the USENIX Usability, Psychology, and Security Workshop (UPSEC), April 2008.

S. Chiasson, A. Forget, R. Biddle, and P. van Oorschot. Influencing users towards better passwords: Persuasive Cued Click-Points. In the proceedings of the Human

Computer Interaction conference (HCI), British Computer Society, September 2008.

Full papers currently in submission are:

S. Chiasson, A. Forget, R. Biddle, and P.C. van Oorschot. User interface design affects security: Patterns in click-based graphical passwords. Technical Report TR-08-14, School of Computer Science, Carleton University, August 2008. (journal submission)

S. Chiasson, A. Forget, E. Stobert, P.C. van Oorschot, and R. Biddle. Multiple password interference in text and click-based graphical passwords. Technical Report TR-08-20, School of Computer Science, Carleton University, September 2008. (conference submission)

Chapter 2

Background

This background chapter provides an introduction to the field of usable security, with a focus on usable authentication, and summarizes relevant methodology in conducting empirical studies. It concludes with an overview of graphical passwords and a summary of published results related to their usability and security evaluations.

2.1 Usable Security

Zurko and Simon [146] introduced the term "user-centered security" in 1996. Davis' 1996 paper [26] on "Compliance Defects in Public-Key Cryptography", as well as Whitten and Tygar's 1999 paper [134] , "Why Johnny Can't Encrypt", drew further attention to the need to couple security and usability research. In particular, these demonstrated that usability problems can lead directly to security vulnerabilities. In 2005, Cranor and Garfinkel edited the first book on "Security and Usability" [25], bringing together work from various researchers and highlighting different areas within the field. The main publication venue specifically for usable security research is the Symposium on Usable Privacy and Security; it has been held annually since 2005. Usable security remains an active and growing research area.

Designing user interaction for security applications, and user authentication systems specifically, raises some interesting challenges. The area of usable security can draw from existing Human-Computer Interaction (HCI) knowledge, but some fundamental differences must be taken into account. Properties of security systems that set them apart include:

- In addition to legitimate users of a security system, there is a second group of users who are actively trying to attack the system. Such attackers will exploit any information leaked by, or that can be extracted through, the interface. They

will also leverage any way that the system can be misused or any means to trick legitimate users into revealing confidential information. This makes it difficult to provide some forms of helpful feedback in the user interface, for example to help guide users towards correct passwords, as it may also help attackers.

- Security is typically a secondary task [134]; if security impedes users' primary goals, users will often try to circumvent the security measures [5, 26, 104].

- Users have poor mental models of security [20, 134] and they may not even realize that their actions are insecure in the first place. Furthermore, they often misunderstand or underestimate the consequences of insecure actions.

- Computer security suffers from the "barn door" property [134]: if information or a system is exposed even for a brief time, there is no guarantee that it has not been compromised in an irrecoverable way. The information may have been externally leaked to attackers, or available to malware resident on the system.

While these represent security concerns, they are all directly related to users of the system and as such, solutions must focus as much on the HCI aspects of the system as on the technical security components. Usability problems may significantly impact the real-world security of the system. User interface design decisions may unintentionally sway user behaviour, often towards less secure behaviour. This may be a direct result of the particular interface, or may be compounded by external influences such as when users reveal their passwords to others due to social expectations. Furthermore, the easiest way of using a system is often also the least secure way. For example, users may choose very short, simple text passwords because these are easier to remember and enter than longer, more complex sequences of characters.

2.2 Authentication

Using Renaud's model [96], the authentication process can be described as three phases: identification, authentication, and authorization. Users must first make some claim of their identity, provide evidence to substantiate this claim, and if successfully authenticated by the system, access rights are granted to the user.

We classify authentication mechanisms according to the following categories, primarily based on Renaud's model [96]:

Something you know (recall): A secret is shared between the user and the system. Users must *recall* and correctly enter their secret to authenticate themselves. Anyone who knows or guesses the secret will also be able to authenticate as the original user. Examples include passwords and PINs (Personal Identification Numbers).

Something you recognize (recognition): The user and the system share a secret. The system provides cues and the user must correctly *recognize* the secret. Anyone able to recognize the secret will be able to authenticate as the original user. Graphical passwords where users must recognize pre-selected images from a set of decoys fall into this category. *Cued recall* systems combine recall and recognition. Users must recognize the cue presented by the system and then use this cue to recall the secret shared with the system.

Something you are (static biometrics): Biometrics measure some unique physical characteristic of the user. These are more difficult to forge than the first two categories but introduce additional concerns. They may require specialized equipment, are difficult or impossible to change if compromised, and have potential privacy implications (e.g., they may make it difficult to create different identities for various purposes, and they enable organizations to cross-reference information about a user). Static biometrics include fingerprint, iris, and facial scans, among others.

Something you do (behavioural biometrics): Some unique behavioural characteristic of the user can also be measured. Users authenticate by repeating the required action. Examples include handwritten signatures and keystroke dynamics.

Something you have (tokens): Users must carry a token to be presented for authentication. Anyone who gains access to the token will be able to authenticate

as the original user. These are often combined with a PIN or password to offer some protection in case the token is lost or stolen. A smart card, i.e., a card with embedded microprocessor chip, is an example of a token used for authentication.

Where you are (location-based authentication) [29]: Location information can be used to determine if a user is attempting to authenticate from an approved location. This is typically used as a secondary check to identify suspicious login activities. Approved locations may be specific, such as a user's office, or more general, such as identifying the city or country of origin.

2.2.1 Text passwords and the password problem

Despite the large number of options for authentication, text passwords remain the most common choice [96] for several reasons. Text passwords are easy and inexpensive to implement, and are familiar to most users. Passwords allow users to authenticate themselves without violating their privacy, as biometrics could, since users can select passwords that do not contain personal information. And finally, passwords are portable since users simply have to recall them, as opposed to tokens which must be carried. However, text passwords also have a number of the inadequacies from both security and usability viewpoints, such as being difficult to remember and being predictable if user-choice is allowed [27, 66, 103, 141].

Passwords are only secure if they are difficult for attackers to guess, yet are only usable if users can remember them. The "password problem" is defined [136] as the current situation where many passwords are either weak-and-memorable or secure-but-difficult-to-remember, despite the need for secure and memorable passwords.

Systems sometimes provide on-screen advice on how to create more secure passwords (e.g., select something memorable that would be difficult for others to guess), give feedback about password choice (e.g., with a password strength meter), or force users to create passwords that comply with specific system-defined rules (e.g., the password must include both letters and numbers). Despite these strategies, users often select weak passwords [41] that are predictable and are easy for attackers to guess. This occurs partially because users misunderstand the advice or requirements,

underestimate the risks, and because limitations of human memory mean that they must employ coping mechanisms in order to reduce the burden of remembering so many passwords [1]. These coping mechanisms may include reusing passwords across several accounts, using predictable alphanumeric combinations, or storing passwords in an easily accessible, insecure location [1, 41, 103, 130]. Although they have appealing characteristics, only limited success has been achieved through encouraging the use of passphrases [63] (passwords are longer phrases) or mnemonic passwords [69] (passwords are abbreviated from a longer word or phrase, for example by using the first letters of the words in a phrase, or including common character substitutions such as "I<3c@s" for "I love cats"). At least in their basic form, both suffer from predictability problems because users choose common character substitutions or well-known phrases. Such approaches also do not mitigate the problem of remembering which password corresponds to which account when users have multiple accounts. Furthermore, phishing [33, 60] and other social engineering [140] attacks on passwords have increased dramatically over the past few years since text passwords are easy for users to unintentionally reveal to attackers, complicating matters further.

A proposed solution to these password problems is to use *password managers*. One class of these managers maps easy to remember (weak, low-entropy) user passwords onto stronger passwords (more resistant to guessing attacks), and may also generate site-specific passwords (protecting against some phishing attacks). Password managers exist in different formats: stand-alone applications, browser plug-ins, and browser scripts.

As preliminary work to this thesis, we investigated two password managers [20]. Our work shows that while the idea of password managers is promising, in their present form these systems have a number of usability problems that lead to decreased security. We conducted a user study of two browser plug-ins: PwdHash [98] and Password Multiplier [53]. We found that the most significant problems arose from users having inaccurate or incomplete mental models of the software. Our study revealed many interesting misunderstandings, such as users who reported that a task was easy even when they were unsuccessful at completing that task, and users who believed that their passwords were being strengthened when in fact they had failed to

engage the appropriate protection mechanism. Such "dangerous errors" are especially concerning because they may have serious security consequences. Our findings also suggest that in the absence of additional education or other means of encouragement, ordinary users would be reluctant to opt-in to using these managers: users were uncomfortable with "relinquishing control" of their passwords to a manager, did not feel that they needed the password managers, and did not believe that these password managers provided greater security.

Text passwords are a type of knowledge-based authentication, where users must prove knowledge of some secret. Graphical passwords are an alternative type of knowledge-based authentication. In graphical passwords, images or visual representations are used instead of alphanumeric characters. The premise behind graphical passwords is that humans have better memory for images than text [65, 72, 86, 108], so this may be a way of devising more memorable passwords. As this is the main focus of the thesis, it will be discussed separately in Section 2.4.

2.2.2 Password spaces

We distinguish that password systems have both theoretical and effective password spaces. The former space includes the set of all (theoretically) possible passwords. The vast majority of user choices tend to fall into a much smaller subset of the full theoretical password space, known as the effective password space. To illustrate, consider the set of all possible 8-character alphanumeric passwords. Including symbols, there are 95 keyboard characters to choose from, giving a theoretical password space of $95^8 \approx 6.6 \times 10^{15}$ possible permutations. The effective password space is much smaller since many character combinations are unlikely to be selected by users (e.g., seemingly random character strings such as "R9&i}3q/"). To offer some perspective, there are approximately 1 million (10^6) words in the English language [73]. The effective password space is an approximation, based on probability estimations that given passwords are chosen by users. Passwords with probabilities higher than some agreed upon threshold make up the effective password space.

An important security goal of authentication mechanisms is to maximize the effective password space; we would like the effective password space to include as much

of the theoretical password space as possible (ideally, all of it). Since the effective password space is determined by user behaviour, the design of a system involves usability as well. Ideally, passwords should be secure without sacrificing the usability of the system. In practice, increasing one often reduces the other, so typically a middle-ground must be found where both the security and usability of the system are acceptable.

Measures of the effective password space are imprecise approximations. One approach that may help is to identify classes of passwords that have higher probability of being chosen by users. In this case, a proximity function (a measure of similarity between items) may be useful. With text passwords, there is no single, obvious measure of what makes two passwords similar: Words or letters in the same positions? Common pet names or birthdays? Some other measure? One possible measure is the "edit distance" [79]: the minimum number of operations (substitution, removal, or insertion of a single character) required to transform one string of characters into another. The edit distance, however, does not take into account the semantic meaning of passwords and may not be a very helpful metric for measuring the similarity of passwords. For example, "F3u}fy" and "Fluffy" have an edit distance of 2, while "Snowball" and "Fluffy" have an edit distance of 8, but semantically Fluffy and Snowball are both popular cat names and probably more commonly used as passwords than "F3u}fy".

2.2.3 Attack models

Many strategies exist for attacking authentication systems. No system offers perfect security; therefore schemes must be evaluated according to their vulnerabilities. For a particular attack strategy, it is possible to compare the susceptibility of different schemes. In practice, the likelihood of such attacks cannot be accurately predicted since it is unknown what attackers may target next. We now identify several possible attack models for password systems.

Dictionary Attack [14, 142]: In a dictionary attack, a list of likely passwords is compiled based on knowledge or assumptions of typical user behaviour. Entries in the dictionary can be further prioritized to test passwords with higher probability of success first (if these probabilities can somehow be calculated or

predicted), increasing chances of quickly finding a match. Dictionary attacks can lead to efficient password guessing because users are likely to select from a relatively small and predictable password space. Recent research [95, 105, 118] suggests that dictionary attacks remain a serious on-going threat, although exact statistics are not widely available since most organizations do not reveal such breeches in security.

In an *online dictionary attack*, interaction is required with the live system; usually each password is entered in turn to see if login is successful. The success of this type of attack can be reduced by limiting the number of incorrect login attempts allowed by the system (before locking the system from all further login attempts) for a particular user account. However, in multi-account attacks [93], attackers may target many accounts on the system instead of a specific account, and for example try several guesses on each of many different accounts, increasing the chances of success on at least some accounts. Furthermore, there is a usability cost to locking accounts after a number of incorrect attempts since legitimate users who simply forgot their password may also be locked out; this can also be used to launch an effective denial-of-service (DoS) attack against users by purposefully entering incorrect passwords and locking out accounts [93].

In an *offline dictionary attack*, attackers must first gain access to some verifiable text [51] (such as the hash of user's password) and do not need to go through the live system to determine if a guess is correct. Schemes that are vulnerable to offline attacks are at a higher risk than those requiring online verification because work can be done behind the scenes and trial guesses can be processed much more quickly. Hashing and salting can be used to slow offline attacks. *Hashing* encodes passwords using a one-way cryptographic hashing algorithm; only the result of the hashing operation is retained and stored by the system. To verify if a login attempt is successful, the system (or attacker) hashes the candidate password and compares the result with the stored password hash. *Salting* [66] concatenates a string of characters to a password before hashing it for storage by the real system. This salt is user-specific and stored in a manner accessible to the system, along with the hashed password, so that it can be

concatenated with the user's input password during login. The resulting string is hashed and compared for a match against the stored hash. This effectively forces attackers in an offline attack to compute the hash for each candidate password guess on a per-user basis. Password cracking tools, such as John the Ripper [30], are readily available to automate offline dictionary attacks (these tools or their dictionaries may also be modified for use in online dictionary attacks). John the Ripper takes hashed passwords and compares them to lists of potential passwords that it hashes in the same format as the passwords being examined, in an attempt to find matches. When matches are found, the program reports the plain text passwords to the attacker.

Exhaustive (brute-force) Attack [142]: Exhaustive attacks can be executed in a similar manner to dictionary attacks, except that every possible password permutation is generated and used to attack the real passwords. In a more sophisticated attack, these permutations may also be prioritized in order of decreasing probability of being selected by users, if such probabilities are somehow predictable. Like dictionary attacks, exhaustive attacks can be launched either online or offline. The advantage to this type of attack is that with enough time and computing power, a match will be found (unless an online attack is detected and stopped before the list is exhausted), but with large password spaces it may not be feasible to search the entire space. In contrast to a dictionary attack, an exhaustive attack offers better coverage but requires more time or processing power.

Shoulder-surfing [7, 70, 100, 117]: Shoulder-surfing refers to attackers acquiring knowledge of a particular user's credentials through direct observation, or through external recording devices such as video cameras, while the legitimate user enters the information. Availability of high-resolution cameras with telephoto lenses and surveillance equipment make shoulder-surfing a real concern if attackers are targeting specific users and have access to the same geographic location as these users. This is especially problematic in public environments, but may not be as serious a threat in other more private environments.

Phishing [33]: Phishing attacks involve tricking users into entering their credentials (username, password, credit card numbers, etc.) at a fraudulent website that is masquerading as a legitimate site. Users normally reach these phishing websites through spam email enticing users to click on an embedded link that directs them to a website designed to look like a site for which they have a legitimate account. When users attempt to log in, attackers record the user's credentials and subsequently use them for fraudulent purposes.

Social Engineering [74, 140]: Social engineering includes any technique used to trick people into divulging their credentials or private information to untrustworthy parties. Phishing is an example of social engineering using email and websites, but social engineering can also be done using other means, such through as phone calls claiming to be from the user's bank, credit card company, or tech support. It is often easier to obtain a password or credentials from the legitimate user than trying to break into a system by other means.

Malware [94]: Malware (i.e., malicious software) includes any unauthorized software that is installed without a user's informed consent. Such software has a malicious purpose, and can include viruses, worms, and ActiveX or JavaScript components [94, 98]. One category of malware is intended to gather confidential information, including user credentials, from the computer on which it is installed. For example, key-loggers record keyboard input, while mouse-loggers and screen scrapers capture mouse actions and the contents of screen memory, then either send this information back to the attacker or otherwise allow attackers to retrieve it.

2.3 Empirical Research on Usable Authentication

Whereas advances in user authentication used to be primarily the domain of security researchers who focused on the mathematical and technical aspects, there as been recent acknowledgement that usability of an authentication scheme is also of prime importance. User behaviour has a significant impact on the security of a system, therefore poor usability may lead directly to poor security. In this section, we provide

an overview of relevant HCI methods for assessing the usability of a given system.

Usability refers to the ease with which users can employ a particular tool to achieve a specific goal. The usability of a computer system can include factors such as its learnability, its efficiency of use, its memorability, and user satisfaction with the product [83]. There are two general categories of methods for assessing the usability of a system [82]: usability inspection methods and user studies. With usability inspection methods (such as cognitive walkthroughs [133] and heuristic evaluations [82]), evaluators inspect and evaluate usability-related aspects of a system. These are conducted without end users and require a certain level of expertise in usability [82]. They are useful in finding obvious usability problems, but are no substitute for user studies with real users because the effects of human behaviour and context in which they perform their tasks are too complex to predict accurately. Typically, usability inspection methods are used early on to guide the design process, then user studies are conducted to confirm the design decisions and find any problems that may have been overlooked.

User studies can range from closely controlled experimental studies testing specific hypotheses, to field studies where the system is deployed for real usage, and system logs and interviews are used to assess its usability. Most user studies fall somewhere in between, conducted in a lab, with pre-determined tasks, but also leaving room to observe users in a more ad hoc manner to uncover unexpected problems as they arise [109]. User studies are the primary means of determining whether a system is suitable for the intended audience and for its intended purpose.

2.3.1 Lab studies

Lab studies provide a means to evaluate the success of design decisions in isolation, quantify improvements and performance, discover unexpected usability problems, and identify designs with higher probability of success (or failure) before investing large amounts of time and resources in field studies. Lab studies have the advantage of being held in a controlled setting. The experimenter can ensure that participants are focused on the task at hand, that the study is designed to enable statistical testing of different measures, and that clear comparisons can be made to assess the effectiveness

of certain design decisions. For example, a study may have a goal of examining the effectiveness of a new password selection aid. In this case, two versions of the system would be built, differing only in the inclusion or absence of the new selection aid. The system would be instrumented to record the user's choice of passwords and input during password entry, and to include measures such as time to create a new password and number of errors made. With security systems, it is especially important to be relatively confident of a system's design in the lab before deploying it in field studies because of the potential for security and privacy breeches of the users' real resources and information if problems occur in a field study.

Besides the pre-determined measures, lab studies aim to uncover any unforseen difficulties encountered by the users as they go through a set of predetermined tasks. These tasks should be carefully chosen to reflect realistic usage scenarios. To preserve ecological validity, the environment should be set up to mimic reality as closely as possible in terms of technical details and instructions given. Users should be closely observed as they perform these tasks, as this is how many usability problems are revealed. The observer's role is mainly to observe and record what is happening. Observers need to be careful not to provide extra instructions or cues that may influence the user's actions. In fact, a script should be used to ensure that all participants receive the same information. It is important to emphasize to the user that it is the system that is being tested and not themselves; the users should feel that they are helping with the development of the system rather than feel like their performance is being evaluated. The researcher must also try to avoid biasing user behaviour, especially when dealing with security, as users may behave more or less securely than usual to "help" the researcher. A method called "think-aloud" is often used, where users keep a running commentary as they perform the tasks. Pre/post questionnaires or interviews are also useful in gathering users' opinions, attitudes, and feedback about the system. These should be a secondary source of information, used in conjunction with observations and potentially system logs, because users' reported views often do not reflect their performance and often fail to reveal crucial usability problems.

An often cited guideline, advocating smaller, quicker usability studies, states that five users are enough to discover most usability problems [81, 129]. It has long been

used to justify small usability studies. Recent work questions this assumption and highlights the fact that five users are often not enough and that in some cases, severe usability problems are only discovered after running a larger group of participants [40, 91, 110]. The likelihood of finding usability problems is not evenly distributed and may vary with the complexity of the system being tested. Some problems only arise under specific circumstances, so using a small sample of users may not be sufficient to uncover them. The variability in the number of problems found by studying any one user also makes it unlikely that a sample of five users would discover most usability problems. Faulkner [40] justifies that twenty users "can allow the practitioner to approach increasing levels of certainty that high percentages of existing usability problems have been found in the testing". When conducting user studies on authentication mechanisms that involve user choice, there is an additional motivation for larger studies: patterns in user behaviour may lead to weakened security and these patterns may only become apparent with a larger sample. Once a system is deployed, attackers may observe and gather information from a large user population in order to best plan their attack strategy; therefore, it is important that designers also attempt to uncover such vulnerabilities in order to guard against them.

2.3.2 Field studies

Field studies are typically used after lab studies have shown appropriate results since field studies require more time, effort, and often have higher costs. In a field study, the system to be tested is deployed for a group of users who incorporate the system into their regular routine over a period of time (typically a few weeks to a few months). This allows researchers and designers to observe how the system would operate in real-life and more accurately judge its acceptability, suitability, and usability. With usable authentication research involving passwords, field studies provide data on what types of passwords users really select when they need to use them regularly, whether the passwords are memorable, and whether circumstances such as interference from having to remember multiple passwords causes problems not apparent in the lab. Real-world usage is of particular concern with security systems because security is often a secondary task [134], enabling (or hindering) access to the user's primary

goal. In such cases, user behaviour may vary considerably compared to when users are asked to complete the security tasks in the lab, where it is their primary focus.

2.3.3 Web-based studies

Although less accurate, another type of user study is gaining popularity: unsupervised web-based studies [6, 41]. The advantages are that large numbers of participants can be recruited, the participant pool is likely more diverse than in most controlled studies, participants can be prompted to complete tasks at several different times, and participant behaviour may be more natural than in a lab setting. Web-based studies are often cheaper, easier, and faster than traditional controlled studies. However, several disadvantages must also be considered: it is difficult to get informed consent from participants (as required by university or institutional ethics review boards) because often a signature or other means of authentication is required, it is nearly impossible to know if the demographics information collected is accurate, it is difficult to enforce any adherence to procedures, and it is difficult to verify that the collected data reflects real behaviour.

2.3.4 Statistical analysis

When conducting user studies, statistical analysis is used to assess whether differences in the data reflect actual differences between conditions or whether these may have occurred by chance. Four types of standard statistical tests [55] for significance were used during data analysis in this thesis, each intended to determine whether the groups being analyzed were distinct from each other with respect to the factor being tested. As described in Table 2.1, results from ANOVAs are reported when comparing the means across multiple groups, t-tests are used when comparing means between two groups, Mann-Whitney tests are used when comparing ordered categorical data (e.g., Likert scale responses, where the choices are discrete and ordered, but it cannot be assumed that users view all pairs of adjacent levels as equidistant), and Chi-square tests (χ^2) are used for non-ordered categorical or nominal data (e.g., comparing login success/fail ratios for click-based graphical passwords on several different images, each login attempt results in either "success" or "fail").

Table 2.1: Summary of the statistical tests used in this thesis.

Name	Usage	Example	Variables
ANOVA (Fisher's F test)	Compares variance of the means between more than two groups	$F(a,b) = n,\ p < .05$	$a =$ between-groups degrees of freedom, $b =$ within-groups degrees of freedom, $n =$ value of the F statistic, used to determine p, $p =$ significance level.
t-test	Compares variance of the means between two groups	$t(a) = n,\ p < .05$	$a =$ degrees of freedom, $n =$ value of the t statistic, used to determine p, $p =$ significance level.
Mann-Whitney U	Compares the probability distributions of two samples of ordered categorical data	$U = n,\ p < .05$	$n =$ value of the U statistic, used to determine p, $p =$ significance level.
Chi-square χ^2	Compares the probability distributions of two or more samples of non-ordered categorical data	$\chi^2(a, N = b) = n,$ $p < .05$	$a =$ degrees of freedom, $b =$ sample size, $n =$ value of the χ^2 statistic, used to determine p, $p =$ significance level.

The statistical results in this paper are reported according to the generally accepted style for HCI publications, which is similar, but not identical, to American Psychological Association (APA) style [3]. In all cases, a value for $p < .05$ indicates that the groups being tested are different from each other with at least 95% probability, making the result statistically significant. In the tables, a value of n.s. means that the result was "not significant" — indicating no difference between the two groups with respect to the variable being tested. The p value is typically the most important value used by the reader for interpretation of the results, however, other values are also reported. In Table 2.1, we summarize the meaning of these values.

We also use spatial statistics in our analysis, but since their use is restricted to Chapters 5 and 7, these statistics will be introduced as needed within these chapters.

2.4 Graphical Passwords

For over a century, psychology studies have recognized the human brain's superior memory for recognizing and recalling visual information as opposed to verbal or textual information [65, 72, 86, 108]. The most widely accepted theory explaining this difference is the "dual-coding theory" [85], suggesting that verbal and non-verbal memory (i.e., word-based or image-based) are processed and represented differently in the mind. Images are mentally represented in way that retains the perceptual features being observed and are assigned perceived meaning based on what is being directly observed. Text is a form of knowledge representation. Text is represented symbolically, where symbols are given arbitrary meaning that describes the object represented by the text, as opposed to perceived meaning. For example, 'X' may represent the roman numeral 10 or the multiplication symbol; the exact meaning is assigned based on some deeper concept. Furthermore, images may be encoded twice, perceptually and symbolically, if meaning is assigned to the image.

Graphical passwords are intended to capitalize on this human characteristic in hopes that by reducing the memory burden on the user, more secure (e.g., longer or more complex) passwords can be produced and users will not resort to unsafe practices in order to cope [61, 77, 115].

2.4.1 Categorization of graphical passwords

This section provides an overview of graphical password schemes available in the literature. Published details, methodologies, and reported results vary greatly, making it difficult to get an accurate comparison. We have tried to compare the schemes first on usability measures and secondly on security measures. Graphical passwords can be grouped into three general categories based on the type of cognitive activity required to remember the password [27, 96]: recall, recognition, and cued recall. We begin by summarizing the usability and security results to provide an overview of the space. Detailed descriptions of each category and representative schemes are provided in Sections 2.4.2, 2.4.3, and 2.4.4 respectively. Published surveys of graphical passwords circa 2005 are also available from Suo et al. [115] and from Monrose and Reiter [77].

Table 2.2 compares the usability of 11 graphical password systems. The times

and success rates are based on published results and unfortunately may not have been calculated in exactly the same way for each scheme. They do, however, provide a range for general comparison. The types of user studies are identified as "Lab" for single session lab studies, "Multi-session" for lab studies where participants returned at least once, and "Field" where the system was deployed for real use for several weeks or months.

Evaluation of the schemes based on security measures is available in Tables 2.3, 2.4, and 2.5, organized by memory classification to keep the tables to a manageable size. Where possible, details and numbers are reported from the original publications (note that we have not independently verified these). In categories that were not addressed in the original papers, we provide our interpretation and assessment of the scheme. The columns of Tables 2.3, 2.4, and 2.5 represent the characteristics listed below.

Scheme: The name of the graphical password scheme.

Theoretical Pswd Space: A measure of the number of passwords in the theoretical password space.

Effective Pswd Space: A summary of any characteristics of the scheme that may make it more susceptible to dictionary attacks or targeted attacks. Dictionary attacks can be successful when user choice is allowed in password creation because people tend to make similar, and predictable choices. When observing a large number of user-selected passwords, one finds that passwords are not selected from the theoretical password space with equal probability, leading to the smaller effective password space. Attackers who can predict which passwords fall within the effective password space (or portions thereof) can build dictionaries of passwords with higher probability of being selected, therefore increasing the effectiveness of their attack. We define targeted attacks as attacks targeted or customized (personalized) towards a particular user. Attackers may use knowledge of the user to determine likely passwords, if password selection allows for personally customizable/identifiable choices.

Offline Attack: Ideally, passwords are encoded using a cryptographic one-way hash for storage, to provide an additional level of security if an attacker gains access

to the stored passwords. This means that the system has no record of the clear text password and can only decide if a login attempt is successful by first hashing the login input and comparing it to the stored hash value, looking for a match. In some graphical password schemes, the system must retain knowledge of some details of the shared secret, i.e., user-specific profile data. For example, in recognition schemes, the system must know which images belong to a user's portfolio so that it can display them. This information must be stored "in the clear" (in the sense of being known to the system; storage under reversible encryption, for example, would be fine), and thus would be available to anyone who gains access to the stored information.

Shoulder-surfing: The number of logins that would need to be observed or recorded in order to have enough information to successfully log in. Some schemes reveal the entire password with every login, while others reveal only partial information so several logins need to be observed before gaining sufficient knowledge to replicate the password entry.

Phishing: A summary of whether this scheme is susceptible to phishing attacks. With some schemes, the fraudulent site requires no preliminary information about the user's account. For others, the phishing site needs to retrieve and relay information between the legitimate site and the user, therefore, a "man-in-the-middle" (MITM) attack is necessary for a successful phishing attack. Similarly to shoulder-surfing, we also note whether one login is sufficient to gather all necessary information to log in independently, or whether the attacker would need to trick the user into logging on multiple times before gaining enough information. It should be noted, however, that with a MITM attack, attackers will always be able to log in to the legitimate site at least once, while the attack is in progress.

Social Engineering: A summary of the scheme's susceptibility to social engineering attacks where an attacker may trick the user into revealing their password. While being resistant to social engineering attacks can be viewed as beneficial for security, it may also make it more difficult from a usability perspective.

For example, users may be unable to effectively write down their passwords for storage or backup purposes, and it may be difficult to legitimately reset such a password over the phone.

Malware: The types of malware that could be used to record enough information for the attacker to log in independently. We focus on keyboard loggers ("keyboard"), mouse loggers ("mouse"), and screen scrapers ("screen").

This survey of published graphical password research revealed a significant lack of consistency in the type of usability and security evaluations conducted on different schemes. Few schemes have been thoroughly evaluated from both usability and security perspectives; typically the authors have focused on (at best) one or the other. Complicating matters further, the metrics reported vary considerably for different schemes, making it very difficult to accurately compare the performance and characteristics of various graphical password schemes. This survey does not include references to any of the new schemes contained within this thesis, and tables 2.2 and 2.5 do not include results of our published PassPoints studies, presented later in this thesis. However, tables 8.2 and 8.3 in Chapter 8 provide a summary of our new graphical password schemes following the same format as the security and usability tables discussed here.

Table 2.2: Usability comparison of previous graphical passwords, ordered by type of memory. Cells with * indicate our interpretation or estimation since the relevant issue was not discussed by the original authors. The $\approx$ symbol is used when exact numbers were not available and we interpreted the information from graphs in the published papers.

Scheme	Type of Memory	Time to Create Pswd	Time to Login	Login Success Rate	Number of Images Needed	Types of User Studies
A. Draw-A-Secret (DAS) [61, 78, 127]	Recall	Not reported	Not reported	Not reported (cf. Pass-Go)	None	Paper-based
B. Passdoodle [47, 52, 128]	Recall	Not reported	Not reported; needs training to tune recognition alg.	Not reported	None	Lab, only to collect training data for algorithms
C. Pass-Go [116, 127]	Recall	Not reported	Not reported	78%	None	Field
D. Déjà Vu [32]	Recognition	45 sec	32-36 sec	90-100%	Fixed set of 10000	Multi-session
E. PassFaces and Faces [13, 27, 37, 87, 117, 122]	Recognition	180-300 sec (for 5 rounds)	Not reported	72-100%, 95%, and *$\approx$ 96%	Per user: 9 per round, 4 rounds in the studies	Field
F. Story [27]	Recognition	Not reported	Not reported	*$\approx$ 85%	Per user: 9 per panel	Field
G. Weinshall's scheme [49, 131]	Recognition	Extensive training, 2-3 sessions	90-180 sec	$>$ 95%	Per user: 80 images per panel, several panels	Multi-session
H. 3D Password [2]	Cued recall	No user study	No user study	No user study	Depends on objects / actions implemented	None
I. Inkblot Authentication [113]	Cued recall	Not reported	Not reported	*72-80%	Per user: 10 computer-generated inkblots	Multi-session
J. Blonder's scheme and Passlogix [10, 88]	Cued recall	No user study	No user study	No user study	Per user: 1 image	None
K. PassPoints [9, 35, 37, 50, 135–137]	Cued recall	64 sec + 171 sec training	9-19 sec	55-90%	Per user: 1 image	Multi-session

Table 2.3: Security comparison of recall-based graphical password schemes. Cells with * indicate our interpretation or estimation since the relevant issue was not discussed by the original authors.

Scheme	Theoretical Pswd Space	Effective Pswd Space	Offline Attack	Shoulder Surfing	Phishing	Social Engineering	Malware
A. Draw-A-Secret (DAS) [61,78,127]	Depends on grid size and pswd length. E.g., 5×5 grid, length 12, 2^{58} pwds	Symmetry and few pen strokes, *may be personally identifiable	Can be hashed	*One login	*No upfront knowledge needed, one login to repeat	*Complex to verbalize, but could be sketched, can take screen shot	*Screen or Mouse
B. Passdoodle [47,52,128]	Depends on granularity of grid, matching algorithm, drawing speed	*Patterns are likely, may be personally identifiable	*Doodle model must be available to system	*One login	*No upfront knowledge needed, one login to repeat	*Difficult with no visible grid, but could be sketched, can take screen shot	*Screen or Mouse
C. Pass-Go [116,127]	Exceeds DAS due to diagonal moves, finer grid. For 9×9 grid: 2^{77} pswds, (more if colour choice, finer grid)	Symmetry, may be personally identifiable	Can be hashed	*One login	*No upfront knowledge needed, one login to repeat	*Complex to verbalize, but could be sketched, and can take screen shot	*Screen or Mouse

Table 2.4: Security comparison of recognition-based graphical password schemes. Cells with * indicate our interpretation or estimation since the relevant issue was not discussed by the original authors. MITM denotes man-in-the-middle.

Scheme	Theoretical Pswd Space	Effective Pswd Space	Offline Attack	Shoulder Surfing	Phishing	Social Engineering	Malware
D. Déjà Vu [32]	2^{16} passwords	"Attractive images" filtered by hand to decrease likelihood of popular images	Portfolio must be available to system	A few logins	*MITM to retrieve images, multiple logins to repeat	Difficult to verbalize, can take screen shots	*Screen, multiple logins
E. Passfaces and Faces [13, 27, 37, 87,117,122]	2^{13}	Attractive female faces popular, attractive faces of user's own race	Portfolio must be available to system	*One login (observe screen or keyboard, depending on configuration)	MITM to retrieve images, one login to repeat	Difficult to verbalize, can take screen shots	*Screen, and keyboard (if keyboard entry)
F. Story [27]	2^{12}	Some patterns apparent, *may be personally identifiable	Portfolio must be available to system	One login	*MITM to retrieve images, one login to repeat	*Easy unless decoys similar to portfolio images, can take screen shot	*Screen
G. Weinshall's scheme [49,131]	Depends on parameters, e.g., 4^5 for 4 choices and 5 rounds	Portfolio assigned, so no patterns in user choice	Portfolio must be available to system	A few logins	*MITM to retrieve images, multiple logins to repeat	*Difficult because panel of images different at each round	Screen, multiple logins

Table 2.5: Security comparison of cued recall-based graphical password schemes. This table excludes research from the present thesis (and earlier publications related to same). Cells with * indicate our interpretation or estimation since the relevant issue was not discussed by the original authors. MITM denotes man-in-the-middle.

Scheme	Theoretical Pswd Space	Effective Pswd Space	Offline Attack	Shoulder Surfing	Phishing	Social Engineering	Malware
H. 3D Password [2]	*Large, must handle complex tolerance issues (e.g., time, proximity-based)	*Attacks likely possible, if actions/objects can be based on personal preferences	*Unknown, but likely some info must be available to system	*Depends on choice of actions, but likely	*Potentially MITM, but depends on implementation, one login to repeat	*Potentially complex description needed	*Screen and mouse
I. Inkblot Authentication [113]	2^{94} for 20 lowercase letters	*Attacks possible, if users ignore cue and select regular text pswd	*Password can be hashed, but images must be available to system	*One login observing typing	*MITM to retrieve images, one login to repeat	*Easy to describe text password	*Keyboard
J. Blonder's scheme and Passlogix [10, 88]	Depends on total no. of objects and clicks	*Hotspots likely, may be personally identifiable	Can be hashed, but image must be available to system	*One login	*MITM to retrieve image, one login to repeat	*Likely with object description or screen shot	*Screen or Mouse
K. PassPoints [9,35,37,50, 135–137]	Depends on no. grid squares, clicks, e.g. 2^{43} for 373 squares and 5 clicks	Hotspots and patterns, *may be personally identifiable	Can be hashed, but grid identifier and image must be available to system	*One login	*MITM to retrieve image, one login to repeat	Possible with complex description or screen shot	Screen or Mouse

2.4.2 Recall

Graphical passwords requiring pure recall are most similar to text passwords because users must remember their password and reproduce it without any cues from the system. This is a difficult memory task [23] and users sometimes devise ways of using the interface as a cue even though it is not intended as such. For example, we have evidence that users often include the name of the system as part of their text passwords [18].

A. Draw-A-Secret (DAS) [61]:

With DAS [61], users draw their password on a 2D grid using a stylus or mouse (see Figure 2.1). The password is composed of the coordinates of the grid cells that the user passes through while drawing. A drawing can consist of one continuous pen stroke or several strokes. To log in, users repeat the same path through the grid cells. The theoretical password space is determined by the coarseness of the underlying 2D grid and the complexity of the images. A coarser grid helps with usability, while a finer grid increases the size of the password space.

To date, the system has only been user tested through paper prototypes (but see also a similar system, Pass-Go, below), so it is difficult to get an accurate analysis of its usability or security. Nali and Thorpe [78] asked 16 participants to draw 6 "doodles" and 6 "logos" on 6×6 grids. These drawings were visually inspected for symmetry and number of pen strokes. They found that participants tended to draw symmetric images with few pen strokes (1-3) and tended to place their drawing approximately in the center of the grid. This preliminary study has several limitations: users were not told that their drawings were "passwords", users did not have to reproduce their drawings at any point, and data was collected on paper so users did not have to draw using the computer. Consequently, no usability data (login times, success rates, etc.) was collected for the scheme. Van Oorschot and Thorpe [127] categorized DAS passwords into password classes based on characteristics such as symmetry and number of pen strokes. Using this classification, they show that a large number of passwords from the paper-based study [78] and a subsequent study on a similar scheme [116] (Pass-Go, discussed later in this section) fall within these

predictable categories, which could help attackers identify candidate passwords with high probability of success and launch efficient dictionary attacks.

The theoretical password space for DAS depends on the number of cells in the grid and the password length (calculated as the number of coordinate pairs defining the path of the password). For example, with a 5×5 grid, with a maximum password of length of 12 strokes, the theoretical password space is $\log_2(25^{12}) = 58$ bits [61]. Since passwords are based on precise coordinates, DAS passwords may be hashed for storage (i.e., the system can use the hash of a password to verify a user-entered password). However, there is a many-to-one mapping from user-drawn passwords to system-encoded passwords (i.e., passwords in the theoretical password space); for example, all doodles drawn entirely within one grid square are equivalent to a dot.

Although not discussed in the publications about DAS, we now consider other security characteristics. DAS would be susceptible to shoulder-surfing; an attacker would need to accurately observe only one login for the entire password to be revealed. Phishing and social engineering attacks may also be of concern since users may be able to describe their password by verbalizing the path through grid squares or by showing a sketch of the password. Although this would need to be verified through user testing, we suspect that DAS password attacks may be personalized to some extent; that is, someone familiar with the user may have a higher probability of guessing the user's password. For example, some users may choose to draw the initials of their name.

As is the case for all recall-based schemes in this section, phishing attacks can easily be mounted. A phishing website simply has to copy the login page from the legitimate site, including the area for drawing the graphical password (a 5×5 grid in the case of DAS). Once users enter their username and password, this information can be utilized by attackers at the legitimate site. Furthermore, all recall-based schemes in this section, including DAS, are vulnerable to malware attacks based on screen scrapers. They may also be susceptible to mouse-loggers, if an attacker is also able to identify the position of the password entry grid on the screen through other means.

Recently, Dunphy and Yan [38] added background images to DAS to encourage users to create more complex passwords. Their study compared the new BDAS with DAS using paper prototypes. It shows that the background image reduced the amount

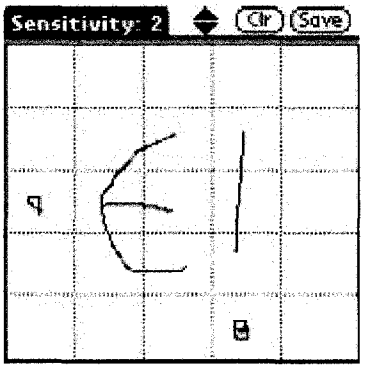


Figure 2.1: Sample Draw-A-Secret password [61]

of symmetry and led to longer passwords that were similarly memorable to the weaker DAS passwords. They did not investigate whether the background images introduced other types of predictable behaviour such as targeting similar areas of the images or image-specific patterns.

B. Passdoodle [47]:

Passdoodle is similar to DAS, allowing users to create a freehand drawing as a password, but without a visible grid. The use of additional characteristics such as pen colour, number of pen strokes, and drawing speed are suggested by the authors to add variability to the doodles. Goldberg et al. [47] report on a small paper-based prototype study of Passdoodle and found that users often remembered their final drawing, but they made mistakes in recalling the number, order, or direction of the pen strokes. In a lab study [128], 10 users created their doodle by tracing it with their finger on a touch screen. Users repeated the trace several times. This data was used as training for the recognition algorithm and it was found that similar input could be accurately interpreted as similar. No further usability or security analysis has been reported.

Later, Govindarajulu and Madhvanath [52] separately proposed a web-based password manager where a "master doodle" was used instead of a master password. In their 10-participant user study, they collected Tamil language character samples using TabletPCs and PDAs. Using only one initial doodle as the master doodle, they used handwriting recognition techniques to evaluate whether the subsequent doodles

were correct and reported 90% accuracy with one of the handwriting recognition techniques.

All three Passdoodle studies focus on the users' ability to recall and reproduce their doodles and on the matching algorithms used to accurately identify similar entries. None of the studies look at usability metrics such as login times or success rates. During password creation, however, Passdoodles would likely require training of the recognition algorithm to build an accurate model of the password.

Although no security analysis has been reported, we provide here a preliminary evaluation comments based on our understanding of the scheme. Shoulder-surfing would be possible with Passdoodle and accurately observing one login would be sufficient to learn the password. However, reproducing the drawing may be difficult and would depend on which measures (such as drawing speed) are used by the recognition algorithm. We expect that Passdoodle would be susceptible to the same types of predictability seen with DAS (symmetry and short passwords) and as such successful dictionary attacks may be possible. As with DAS, some users are likely to choose personally identifiable passwords that can be guessed by someone who knows the user. It would likely be difficult to accurately describe a Passdoodle password since there is no visible grid to act as a guide, although it may be possible to sketch and share such passwords. Passdoodle passwords (the drawings themselves) would likely need to be stored in a manner accessible to the system, as opposed to hashed, since the recognition algorithm must allow for various approximations of the original password.

C. Pass-Go [116]:

Tao's Pass-Go [116] was named for the Chinese board game of Go which consisted of strategically placing tokens on the intersections of a grid. In Pass-Go (see Figure 2.2), users draw their password on a grid, except that the intersections are used instead of grid squares. Visually, the user's movements are snapped to grid-lines and intersections so that the drawing is not impacted by small variations in the trace. Users can choose pen colours to increase the complexity of their drawing. Results of a large field study showed that login success rates were acceptable (as determined by the study's

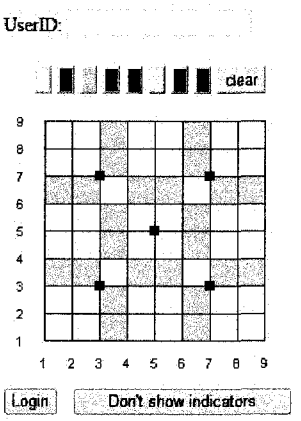


Figure 2.2: Login screen for Pass-Go [116]

authors) at 78%, but no login times were reported. Users chose more complex passwords than with DAS, although a large number of passwords were symmetrical and would be susceptible to attack [127]. The theoretical password space of Pass-Go is larger than for DAS, in part because of a finer grid (more squares), and also because Pass-Go allows for diagonal movement while DAS only permits horizontal and vertical movements. Pen colour was used as an additional parameter and the authors suggest using a finer grid to further increase the theoretical password space. Dictionary attacks may be less effective than DAS since it is reported that users selected longer passwords and used colour; both add variability to passwords. Interpreting other aspects of security, Pass-Go is similar to DAS in terms of shoulder-surfing, phishing, social engineering, and personalization.

A similar scheme was proposed by Orozco et al. [84]. It uses a haptic input device that measures pen pressure while users draw their password. They suggest that this may help protect against shoulder-surfing since an observer would have difficulty distinguishing variances in pen pressure. Results of their user study, however, show that users applied very little pen pressure and hardly lifted the pen while drawing, so the use of haptics did not increase the difficulty of guessing passwords.

2.4.3 Recognition

Several theories exist to explain the difference between recognition and recall memory, based on whether these are two unique processes or whether they are similar and differ only in their retrieval difficulty [4]. It is generally accepted, however, that recognition is an easier memory task than recall [64,121]. In recognition-based graphical password systems, users typically memorize a portfolio of images during password creation and then must recognize their images from among decoys to log in. Humans have exceptional ability to recognize images previously seen, even if those images were viewed very briefly [80, 112]. Several recognition-based graphical password schemes have been proposed in recent years. The most prominent systems available in the literature are described below.

D. Déjà Vu [32]:

In Déjà Vu (see Figure 2.3), users select and memorize a subset of images from a larger sample to create their portfolio. To log in, users must recognize images belonging to their pre-defined portfolio from a set of decoy images; in the test system, a panel of 25 images is displayed, 5 of which belong to the user's portfolio. Users must identify all of images from their portfolio and only one panel is displayed. Images of "random art" are used to make it more difficult for users to write down their password or share it with others by describing the images from their portfolio. The authors report that a fixed set of 10000 images is sufficient, but that "attractive" images should be hand-selected to increase the likelihood that images have similar probabilities of being selected by users.

A 20-participant user study showed that although slower than traditional text passwords or PINs, users could more accurately remember their Déjà Vu password one week after password creation. Users took an average of 45 seconds to create their password. They took an average of 32 seconds to log in immediately after password creation with 100% success rate, and then took an average of 36 seconds to log in a week later, achieving a 90% success rate at that time.

This type of system is not suitable as a replacement for text passwords because with a reasonably sized set of images for usability, the theoretical password space is

only comparable to a 4 or 5 digit PIN. The theoretical password space is $\binom{N}{M}$ where N = number of images in the panel, M = number of portfolio images shown. For example, $\binom{25}{5} = 53130 \approx 2^{16}$ passwords. The authors claim that Déjà Vu is resistant to dictionary attacks because few images in their user study were selected by more than one user, however, this claim has not been rigourously tested. Déjà Vu is slightly more shoulder-surfing resistant than previously described schemes since only a portion of the user's portfolio is revealed during a login attempt. Several logins would need to be observed to identify all of the images in a user's portfolio. Participants in the user study found it difficult to describe the images in their portfolio and users who had the same image gave different descriptions from each other. This provides evidence that it may be difficult for an attacker to gather enough information from a social engineering attack to log in, at least if the attacker relies on the user to verbalize the password. Similarly, it is likely to be difficult to identify images belonging to a particular user based on knowing other information about that user; it is, however, possible that users select images that include their favourite colour, for example.

Screen scraping malware could record Déjà Vu passwords, however, multiple logins would need to be observed before attackers learn all of the images in the user's portfolio. Phishing attacks are more difficult with recognition-based systems such as Déjà Vu because the system must present the correct set of images to the user before password entry. This can be accomplished with a MITM attack where the phishing site relays information between the legitimate site and the user in real-time. In this case, the phishing site would get the user to enter a username, pass this information to the legitimate site, retrieve the panel of images from the legitimate site and display these to the user on the phishing site, then relay the user's selections to the legitimate site; thus the attacker gains access to the user's account on the legitimate site. This is a more sophisticated attack than phishing attacks for recall-based schemes, requiring more effort on the part of the attacker. A similar type of MITM attack can be launched against all of the recognition-based schemes discussed in this section.

Furthermore, Déjà Vu requires that identifiers for a user's portfolio images be stored in a manner accessible to the system so the correct images can be displayed during login. This means that passwords cannot be hashed for storage (although

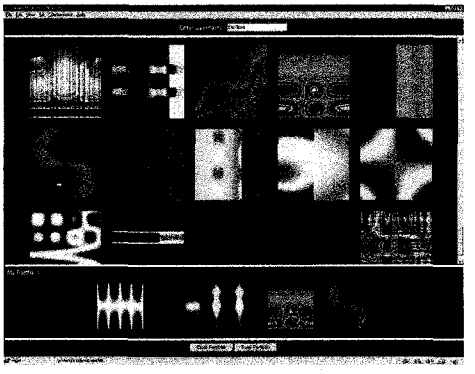


Figure 2.3: Screenshot of the Déjà Vu graphical password system [32]

storage under reversible encryption, for example, would be fine). This is true for all recognition-based systems described in this section.

E. PassFaces / Faces [27]:

In PassFaces [87] (see Figure 2.4), users pre-select a set of images of human faces. During login, they are presented with a panel of candidate faces and have to select the face belonging to their set from among decoys. This process is repeated several times with different panels, and users must perform each round correctly in order to successfully authenticate themselves. In the test systems, a panel consisted of 9 images, one of which belonged to the user's portfolio, and a user completed 4 rounds to login.

In a study with 77 users, Valentine [122] found that people could remember their PassFaces password over extended periods of time, with login success rates of between 72% to 100% by the third attempt for various intervals of time, up to 5 months. Brostoff and Sasse [13] conducted a field study with 34 users, and found mixed results. While users made fewer login errors (95% success rate for PassFaces), they tended to log in less frequently than users who had text passwords because the login process took too long (although no login times are reported). Davis et al. [27] conducted a large field study where students used one of two graphical password schemes to access class material. They implemented their own version of PassFaces, called Faces, for the study. They found that users selected predictable passwords that could be successfully guessed by attackers with little effort. Analysis of user choice revealed

that users tended to select beautiful female faces of their own race. One of their major conclusions was that many graphical password schemes, including Faces, may require "a different posture towards password selection" than text passwords, where selection by the user is the norm. None of the studies reported time to create a password, but the PassFaces corporate website [87] reports that password creation takes 3-5 minutes for a panel of 9 faces and 5 rounds.

Further research has been conducted on the security of PassFaces. Dunphy et al. [37] investigated whether PassFaces could be made less vulnerable to social engineering attacks where attackers convince users to describe the images in their portfolio. They showed that when decoy images were carefully selected so that they were similar to the users' portfolio images, someone hearing a description of the portfolio images was unlikely to correctly enter the password based on this description. However, users could still take pictures of their images and share those images. Tari et al. [117] compared shoulder-surfing risks between PassFaces, text passwords, and PINs in a lab study. They found that when PassFaces used keypad entry rather than a mouse, it was significantly less vulnerable to shoulder-surfing than even text passwords or PINs. If PassFaces uses a keyboard for password entry, then malware attacks would need both a key-logger and screen scraping software to gain enough knowledge for password entry; with regular mouse entry, only a screen scraper is necessary.

The theoretical password space for PassFaces has size M^n where M = number of images displayed in a panel, and n = number of rounds. For example, when $M = 9$, $n = 4$, there are $6561 \approx 2^{13}$ passwords. Davis et al. [27] have shown that users tend to select predictable images; therefore, successful dictionary attacks are possible. Targeted personalized attacks are also possible, for example, if attackers know a user's race or gender. For example, Davis et al. [27] were able to guess the 10% of passwords created by male participants with only 2 guesses. As discussed earlier, phishing requires a MITM attack and portfolio images must be stored in a manner accessible to the system.

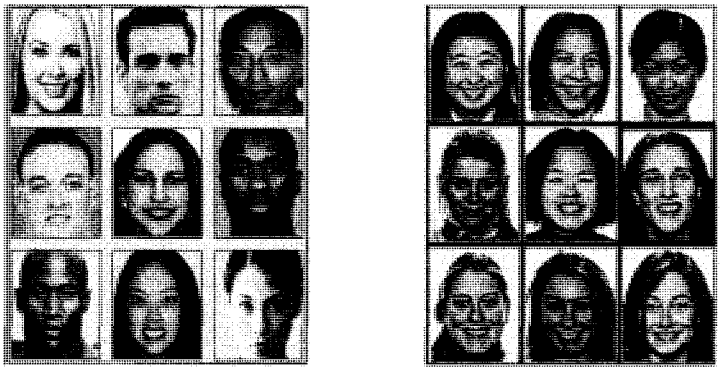


Figure 2.4: Sample panels for the PassFaces graphical password. On the left is a sample panel from the original system [27]. On the right, a panel with decoys similar to the image from the user's portfolio [37].

F. Story [27]:

Story (see Figure 2.5) was proposed by Davis et al. [27] as a comparison system for PassFaces. In Story, users first select a sequence of images for their portfolio. To log in, users are presented with one panel of images and must identify their portfolio images from among decoys. The images contained everyday objects, places, or people. Story also introduced a sequential component by requiring that users select their images in the correct order. To help with memorability, users were instructed to mentally construct a story to connect the images in their set. In the test system, a panel contained 9 images and a user's password consisted of a sequence of 4 images selected from within this panel.

Story was user tested along with Faces as part of the same field study by Davis et al. [27]. They found that user choices in Story were more varied [27] but still displayed exploitable patterns such as differences between male and female choices. Users had more difficulty remembering their Story passwords ($\approx$ 85% success rate) and most frequently made ordering errors. Surveys with participants revealed that they were unlikely to have formulated a story as a memory aid, despite the designers' intentions, which may explain the high number of ordering errors (this might possibly be overcome with different instructions or further experience with the system). Times to create a password or login were not reported.

The theoretical password space of Story depends on M, the number of images

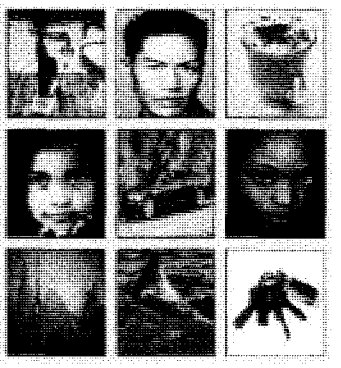


Figure 2.5: Sample panel for the Story graphical password system [27]

displayed in a panel, and n, the number of images in the password. For example, when $M = 9$, $n = 4$, there are $9 \times 8 \times 7 \times 6 = 3024 = 2^{12}$ passwords since the images in the password are in a specific sequence. Davis et al. [27] found that patterns in user choice existed in Story, indicating that it is likely possible to build an attack dictionary that accounts for these preferences. Also, since differences were seen between males and females, and it is likely that users choose images of things they like, a targeted attack may also succeed. Story is vulnerable to shoulder-surfing since the entire password is revealed with every login, especially if the mouse is used as an input device. With respect to social engineering, attackers would likely be more successful at getting users to verbalize their Story passwords than those of PassFaces or Déjà Vu since a panel will include images of various everyday objects and scenes. Similarly to other recognition-based schemes, MITM attacks are possible and portfolio images must be stored in a manner accessible to the system. Story is also vulnerable to malware attacks using screen scraping software.

G. Weinshall Cognitive Authentication Scheme Safe Against Spyware [131]:

Weinshall [131] proposed a graphical password scheme (see Figure 2.6) where login requires that users recognize images from their portfolio. The login task involves computing a path through a panel of images based on whether particular images belong to the user's portfolio. The rules are to compute a path starting from the top-left corner of the panel of images: move down if you stand on a picture from your portfolio, move right otherwise. When the right or bottom edge of the panel is

reached, identify the corresponding label for that row or column. A multiple-choice question is presented, which includes the label for the correct end-point of the path. Users perform several rounds, presented with a different panel each time. After each round, the system computes the cumulative probability that the correct answer was not entered by chance. When the probability passes a certain threshold, then the user is authenticated. This allows for some user error, but if the threshold is not passed within a certain number of rounds, the user is rejected.

The keyboard is used for input, rather than a mouse, to help reduce shoulder-surfing. Users receive system-assigned portfolios of images and receive extensive training to initially memorize their portfolio since it includes a large number of images (approximately 100), but no times were reported for this initial training phase. Login takes from 1.5 to 3 minutes on average. In a user study with 9 participants, a 95% login success rate was achieved overall, with users logging on over a period of 10 weeks.

The main advantage reported by Weinshall is resistance to observation (shoulder-surfing) attacks, however this scheme has been successfully attacked by Golle and Wagner [49]. The attack uses a SAT (boolean satisfiability problem) solver, allowing recovery of the user's secret in a few seconds, after seeing a small number of user logins.

The number of different passwords possible from a user's point of view is $\binom{N}{M}$, based on unique collections of images, where N is the number of images in a panel, and M is the number of portfolio images displayed. For example, for N=80, M=30, there are $\binom{80}{30} = 2^{73}$ passwords. However, due to the redundancy in the scheme which encodes the user's portfolio images into row and column labels, there is a many-to-one mapping of image sets onto system passwords, so the size of the theoretical password space is less than this. For example, assuming that there are exactly 5 rounds and 4 different multiple choice answers, the number of distinct system passwords is $4^5 = 2^{10}$.

Dictionary attacks and targeted attacks have no advantage over exhaustive attacks for this scheme because portfolio images are randomly assigned so all images are equally likely. It would be nearly impossible to verbalize enough information for an attacker (or a friend, if trying to share the password) to be able to log in successfully,

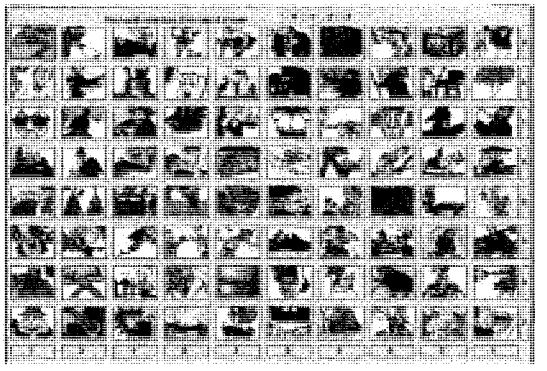


Figure 2.6: Sample panel for Weinshall's cognitive authentication scheme safe against spyware [131]

so this type of social engineering attack is not viable. As demonstrated by Golle and Wagner [49], an attack based on shoulder-surfing can be successful if a few logins are observed. Similar to the other schemes in this section, portfolio images must be stored in a manner accessible to the system, and phishing can be done using a MITM strategy. Multiple logins would need to be captured by screen scraping software for the attacker to gain sufficient knowledge for independent login.

Other recognition-based schemes

Other recognition-based systems have been proposed, but as these have similar usability and security profiles as those already mentioned in this section, we do not not cover them in detail in this thesis. De Angeli et al.'s VIP system [28, 76] displays a panel of images and users must select images from their portfolio from among decoys. Different configurations allow for multiple rounds or sequencing of images. Use Your Illusion, a system by Hayashi et al. [54], also requires that users select their portfolio images from among panels of decoys; the selected images are thereafter distorted in such a way that the legitimate user can still recognize the original images while being difficult for others to identify. The distortion is intended to help protect against social engineering and shoulder-surfing attacks. In the Convex Hull Click Scheme of Wiedenbeck et al. [138], users once again memorize a portfolio of images and must recognize their images from among decoys on the screen, iterating through several rounds. In this scheme, the images are small icons and several dozen are randomly

positioned on the screen, with each panel containing at least 3 of the user's icons. To correctly complete the task, users must identify their icons, visualize the triangle formed by these icons and click anywhere within this triangle. It is intended to help protect against shoulder-surfing, but comes at a cost of longer login times.

Renaud [97] recently completed a field study comparing different types of user involvement in selecting the portfolio images for recognition-based schemes. Users in her study could select images from a photo archive, could take their own photos, or could draw doodles that were subsequently scanned and converted to JPEG format. Results show a significant increase in login success rates when user portfolios contain self-drawn doodles rather than either type of photos. The memorability increases, however, need to be balanced with the additional risk of targeted attacks if attackers know a user's drawing style or recognize personally-identifiable features within the doodles.

2.4.4 Cued-recall

In cued-recall systems, the system provides a cue to help trigger the user's memory of the password (or portion thereof). This feature is intended to reduce the memory load on users and is an easier memory recall task than pure recall. Tulving and Pearlstone [120] explain that items in human memory may be available but not accessible for retrieval. Their results show that previously inaccessible information in a pure recall situation can be retrieved with the aid of a retrieval cue. Ideally, the cue in an authentication system will be helpful only to legitimate users and not to attackers trying to crack a given password.

Several of the cued-recall graphical password schemes surveyed require that users remember specific details within the images (or 3D environment). This is a different memory task than simply recognizing the image as a whole. Hollingworth and Henderson [57] show that people also retain accurate, detailed, visual memories of objects to which they previously attended in visual scenes; this suggests that users may be able to accurately remember specific parts of an image as their password if they initially focused on them. We now provide a survey of graphical password systems that employ cued-recall to facilitate password memory.

H. 3D Graphical Passwords [2]

Alsulaiman and El Saddik [2] proposed a 3D scheme where users navigate a 3D world and perform a sequence of actions interpreted as their password. Much like the 2D graphical passwords in this section, the 3D environment acts as a cue to prompt users to perform their actions. The authors envision that users could perform various actions such as clicking on certain areas, typing or drawing on a virtual surface, entering a biometric, interacting with certain parts of the virtual world (like turning on a light switch), and so on. Their prototype system implements only a small portion of the scheme and provides no details about the other proposed components, so it is difficult to make any usability or security evaluations. The prototype allows users to walk through a virtual art gallery and enter textual passwords at virtual computers or select pictures as part of a graphical password, but no user testing or security results are reported. This appears more of a conceptual proposal at this stage.

The theoretical password space is based on the number of actions required within the world, the number of objects available for interaction within the world, and the password space of each of these object/interaction pairs. For example, if one action is turning on a light switch, there are only two possible states, but if one action is entering an 8-character text password on a virtual computer, then the password space for that object/interaction is 95^8. Without further details of what would be included in the 3D world, it is impossible to approximate the theoretical password space.

The authors suggest that users would select their sequence of actions and interactions; we expect that there would likely be some predictability and opportunity for dictionary attacks as well as targeted attacks. We expect that shoulder-surfing is likely to be a problem since observers will at minimum see the user location within the 3D world, although the extent of the threat would depend on the types of interactions defined in the world. Social engineering attacks where attackers get users to verbalize their password may be possible, but again this depends on the types of interactions allowed (e.g., it would be easy to tell someone to turn on the light switch in the living room, but difficult to describe some types of graphical passwords used within the world). We expect that this scheme would be vulnerable to attacks using both screen scraping and mouse logging, but more implementation details would be

needed to be sure.

3D graphical passwords, as with all cued-recall systems, must provide some information to the user as a prompt before they enter their password. Phishing is, therefore, only possible if the fake system has this information. The most likely way of accomplishing this is through MITM attacks, to which all cued-recall schemes described in this section are susceptible for these same reasons.

I. Inkblot Authentication [113]

Although not strictly a graphical password system, Inkblot Authentication (see Figure 2.7) uses images as a cue for text password entry. The system presents computer generated "inkblots" and users respond by entering text characters that match those earlier selected when the password was created. During password creation, users are shown a series of inkblots and asked to type in the first and last letter of the word/phrase that best describes the inkblot. These pairs of letters become the user's password. The inkblots are displayed, in order, as cues during login and users must enter each of their 2-character responses. The authors suggest that with time, users would memorize their password and would no longer need to rely on the inkblots as cues. Twenty-five users in a lab study were presented with 10 inkblots and created a corresponding password. After one day 80% of users entered their entire password correctly, and 72% were successful after one week. With only one exception, when users made mistakes, it was on only one of their 10 character-pairs. The resulting passwords were relatively strong (20 characters long with no recognizable words, although some letters were more popular than others). The authors claim that inkblots should be abstract enough that an attacker seeing the inkblots would not have an advantage in guessing a user's password.

The theoretical password space is the same as for regular text passwords. In this case, 20-characters are entered, so $26^{20} = 2^{94}$ passwords if only lowercase characters are considered. Users are instructed to select the first and last letter of a word, so the proposed system considers only lowercase letters, but it could allow for a larger character set. If users employed the inkblots correctly (rather than ignoring them and creating a regular text password), dictionary or targeted attacks would have

Figure 2.7: Inkblots used in the Inkblot Authentication user study [113]

little leverage over brute-force. The authors note that some letters and character-pairs were more likely than others and that these follow frequency distributions seen in the English language. Also, since users are instructed to select the first and last letter of a word, the resulting passwords likely include only letters.

Displaying the inkblots on the screen probably does not reveal much for shoulder-surfing; but attackers who observe the user typing their password may gain sufficient knowledge to log in, similarly to the situation with regular text passwords. Social engineering attacks may be just as effective as for text passwords if a user has memorized their entire password (and does not require the inkblots as cues). Inkblot authentication is susceptible to key-loggers since the user's password is alphanumeric.

MITM attacks are possible with this scheme. Another type of phishing attack may also be possible, where the phishing site claims that the "inkblot server" is down for maintenance and requests that users enter their password without cueing. If users have memorized their password, this may be effective. Note that the text password could be hashed for storage, although the inkblots (or a seed for generating them) would need to be available to the system.

J. Blonder's Graphical Passwords [10]:

Blonder [10] was the first to propose click-based graphical passwords. In his scheme, a system administrator prepares an image by defining the perimeter of objects within the image ("tap regions"), typically along the outlines of the objects in the scene. Users select a sequence of these pre-defined objects as their password by clicking on

each object. For example, in Figure 2.8, a password could consist of clicking on the pocket watch, the red bead necklace, the picture on the wall, the watch on the bed, and the camera. To log in, users click on each object in the same order. The image is intended as a cue to help users remember their password. No usability or security analysis has been reported, and indeed, to our knowledge no work has been published on this scheme besides Blonder's patent [10]. According to Suo [115], Passlogix [88] had an implementation similar to Blonder's password scheme as part of their v-GO system (Figure 2.8), although it no longer appears to be available.

A relatively small number of objects could be defined within an image, therefore the number of possible password combinations is limited and could be exhaustively searched. For example, with 50 objects and 5 clicks, the theoretical password space would include $50^5 = 2^{28}$ passwords. We expect that some objects would be more popular than others and that users may tend to select personally meaningful items. In this case, dictionary and targeted attacks may both be effective. An attacker who accurately observes one login would have enough information to log in independently, so shoulder-surfing is a concern. Since distinct objects are selected as part of the password, it seems likely that the password could be verbalized and revealed in social engineering attacks. Because tap regions are distinct objects, passwords created with this scheme could be hashed for storage, i.e., the hashed password would suffice to allow the system to verify the entered password. To capture the password using malware, a mouse-logger may suffice if the attacker is able to also determine the position of the image on the screen. Alternatively, a screen scraper would be necessary to identify the image location. The screen scraper may be sufficient if the attacker can identify when the user clicked the mouse button — the user may not necessarily stop moving the cursor while clicking, especially if they are very familiar with the password.

K. PassPoints [137]:

PassPoints [135–137], as shown in Figure 2.9, is an extension of Blonder's click-based graphical passwords. During password creation, users are presented with an image and select a sequence of any 5 click-points (pixels) on this image by clicking on them

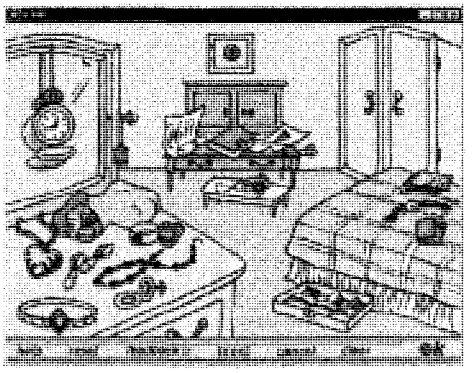


Figure 2.8: Passlogix [88] implementation of Blonder's graphical passwords. Image from [115].

with a mouse. During login, re-entry of the click-points must be accurate to within some system-specified tolerance and in the correct order. PassPoints has a larger theoretical password space than Blonder's original scheme because any pixel can be selected as a click-point. The image acts as a cue, hopefully giving users memory prompts of the location of their click-points since users are expected to select their click-points based on characteristics of this background image. We note that this is not an optimal cued-recall scenario. Users are presented with only one cue and must recall 5 pieces of information in the correct order; we discuss this issue in Chapter 4.

In user studies, Wiedenbeck et al. [135–137] found that users took 64 seconds to initially create a password, and required an additional 171 seconds of training time on average to memorize the password. Login took between 9 and 19 seconds on average and login success rates varied from 55-90%, with users returning at different intervals to log in again.

The implementation of PassPoints requires that for each click-point, an imaginary grid is overlaid onto the image; if a guessed click-point falls within the same grid square as the original point, then the guess is accepted. Birget et al. [9] propose a "robust discretization" scheme to take care of this implementation detail. The number of entries in the theoretical password space for PassPoints is s^c, based on the number s of squares in this grid and the number c of click-points in a password. For example, with the standard configuration tested in the user studies, there are 373 grid squares and 5 click-points [136], giving $373^5 = 2^{43}$ passwords.

PassPoints is vulnerable to hotspots and patterns within images [17, 35, 50, 101, 119, 126] (these issues will be discussed further in later chapters of this thesis). Hotspots are areas of the image with higher probability of being chosen by users, and patterns are simple geometric shapes formed by the 5 click-points in a user's password. These can be exploited to launch efficient dictionary attacks. Although the issue has not been specifically addressed in user studies, we expect that users select click-points that have personal meaning to them, which could potentially be exploited in targeted attacks. Shoulder-surfing can reveal a user's password in one login since the entire password is observable on the screen as the user enters it. Dunphy et al. [37] have preliminary evidence that users can sufficiently describe their password to enable someone else to enter it, so PassPoints may also be susceptible to social engineering attacks. Malware attacks using screen scrapers and mouse logging may be sufficient for learning a user's PassPoints password. These could be used in combination or separately, as discussed above for Blonder's scheme.

PassPoints passwords can be hashed for storage; additional information must, however, be stored in a manner accessible to the system, namely a grid identifier (for each click-point) so that the system can use the appropriate grid when verifying login attempts. PassPoints is described in more detail in the following section since it is the starting point for the studies undertaken for this thesis.

2.4.5 A focus on PassPoints

As mentioned above, PassPoints users create a password by clicking five ordered points anywhere on the given image. To log in, users must correctly repeat the sequence of clicks, with each click falling within an acceptable tolerance of the original click-point. To implement this aspect, a "robust discretization" scheme [9] involving three overlapping grids (invisible to the user) was proposed to determine whether each click-point of a login attempt was close enough to the corresponding original point to be accepted (i.e., is within tolerance). Robust discretization also allows for conversion of the password into a a reproducible deterministic value that can be cryptographically hashed for storage. Robust discretization was not implemented [12] in the prototype systems tested by the original PassPoints creators, so it is unknown

Figure 2.9: Example password on a PassPoints system. The small numbered boxes illustrate the acceptable tolerance area around each of the 5 click-points and would not ordinarily be visible to users [136].

how this implementation would have affected the usability results presented in the original PassPoints user studies. The issue of discretization and problems arising from using robust discretization are discussed in Chapter 6.

Wiedenbeck et al. [135–137] conducted three user studies of PassPoints, examining the effects of image choice and size of the tolerance region, and comparing PassPoints to text passwords. All three studies were conducted in-lab and consisted of having users create a password and practice until they entered it correctly ten times (a learning phase). At the end of the session, users logged in with their newly memorized password. They returned one week later to log in again; in addition, for one study they also returned at the 6-week mark. Unless specifically testing the size of the tolerance region, their prototype used a tolerance region of 20×20 pixels and all images were 451×331 pixels in size.

In the study comparing PassPoints to text passwords, they found that graphical passwords were slower to enter than text passwords and users made more mistakes in the initial learning phase [136], yet they conclude that PassPoints is sufficiently memorable because users made fewer errors with PassPoints when they logged in after one and six weeks. For the second study [137], they compared tolerance squares of

size 20 × 20, 14 × 14, and 10 × 10 on a 19-inch screen at a resolution of 1024 × 768 pixels. The stated conclusion was that while using a smaller tolerance square led to a larger password space, squares of 10 × 10 pixels were too small to be usable, and they recommended tolerance regions of 14 × 14 pixels or larger. A third study compared the usability of different images. They concluded [137] that image choice had little impact on the memorability of passwords; users performed equally well on the four images tested. The issue of "hotspots", areas on the image that users were more likely to select, was briefly considered but they concluded that further investigation was required to determine whether these were a problem.

Later analysis by the original PassPoints authors [35], separately by Golofit [50], and by members of our group [17, 101, 119, 126] confirm that hotspots are a security problem in PassPoints. These issues will be discussed in later chapters of this thesis.

PassPoints has also received attention from others, who have proposed their own modifications. To address the issue of shoulder-surfing, Suo [114] proposes a shoulder-surfing resistant version of PassPoints. During login, the image is blurred except for a small focus area. Rather than using a mouse to select their click-points, users enter Y (for yes) or N (for no) on the keyboard, or use the right and left mouse buttons, to indicate if their click-point is within the focused area. The process repeats for at most 10 rounds, until all 5 click-points are identified. Although not discussed by Suo, this method has obvious security vulnerabilities. Primarily, the user's click-points are guaranteed to be within the 10 focus areas, so observing one login narrows the search space considerably, and observing a few logins would be enough to identify the password.

A commercial version of PassPoints for the Pocket PC is available from visKey [106]. The product is used to unlock a PocketPC by tapping on the correct sequence of click-points using a stylus or finger. Users are able to define settings such as how many click-points a password contains, the size of the tolerance regions, and which image is displayed.

2.5 Terminology Used in this Thesis

In this section, we define some of the terminology used throughout the thesis with respect to graphical passwords.

Click-based graphical passwords: The category of click-based graphical passwords includes password schemes where users are presented with an image (or series of images) and enter their password by clicking on specific areas of the image. Example systems include Blonder's graphical password scheme and PassPoints, as well as the two schemes proposed in this thesis, Cued Click-Points (CCP, Chapter 4) and Persuasive Cued Click-Points (PCCP, Chapter 5).

Click-point: A click-point is as a specific pixel within an image that a user has clicked on with the mouse (or other pointing device). A password in PassPoints, CCP, and PCCP, consists of a sequence of click-points.

Tolerance square or tolerance region: PassPoints, CCP, and PCCP allow for a small margin of error around each of the click-points in the user's original password (the password set during password creation), so that approximately correct login attempts are accepted. This is achieved through the discretization of click-points. To simplify implementation, the tolerance regions are implemented as square areas encapsulating the original click-points. As long as the click-points of a login attempt fall within the tolerance squares for the original click-points, the login is successful. The size of tolerance squares is a system-defined parameter; larger squares improve usability but decrease security, and vice versa.

Hotspot: We use the term hotspot to describe areas on images that have a higher probability of containing click-points. These click-points are selected by users as part of their passwords.

Dictionary attack for click-based graphical passwords: A dictionary attack on click-based graphical passwords can be carried out by formulating a list of likely click-points, typically including hotspots, and using this list to construct candidate passwords (e.g., sets of 5 click-points) to guess user passwords. In a real

world setting, entries in an attack dictionary would consist of whole passwords (in the case of PassPoints, CCP, and PCCP, 5 click-points for the parameter choice used in our studies) and may be further prioritized based on probable click-point patterns (such as straight lines; more discussion on patterns is available in Chapter 7). In this thesis, we also examine our datasets at a click-point level (instead of at a password level) to gain a better understanding of types of click-points selected by users. The dictionary in this case would consist of a prioritized list of individual click-points (hotspots).

2.6 Rationale for the Thesis

Our survey of usable authentication, and especially graphical passwords, revealed that much of the published work to date still focused on either security or usability, with few systems being thoroughly evaluated from both perspectives. Without both usability and security evaluations, we cannot ascertain the suitability of any scheme for real world usage.

One of the main factors in the password problem is that users have difficulty remembering secure passwords. After our preliminary work with password managers revealed serious usability problems, we decided to explore other alternatives. We chose to focus on graphical passwords because of their potential for increased memorability. Recall-based schemes seemed to offer little additional memory benefit since users had to both accurately recall and reproduce their password with no cueing from the system. Recognition-based system appeared to have reasonable memorability (after initial training) but they had an inherent problem of either requiring extensive time to login because several rounds are necessary, or a small theoretical password space. Cued-recall schemes had the potential for improved memorability, reasonable login times, and large theoretical password spaces.

Of the cued-recall schemes we investigated, in our view PassPoints had the most promising features in terms of usability and had good potential based on initial security evaluations. Cued recall makes PassPoints passwords memorable without the need for lengthy training, password entry is relatively fast, and the scheme has a

large theoretical password space (configurable by parameters). Furthermore, click-based graphical passwords such as PassPoints have at least one natural proximity measure which provides an additional feature of interest in their analysis: the spatial distance between two points gives a clear metric for comparing passwords. This characteristic makes it easier to compare user choice in password selection. As such, click-based graphical passwords provide an excellent environment to explore and analyze user password choice, as well as approaches for enlarging the effective password space.

We thus began our work with further analysis of PassPoints, as discussed in the following chapter. We first conducted a more thorough usability and security analysis of the scheme to gain a clear understanding of its strengths and weaknesses. Based on our findings from lab and field studies of PassPoints, we set out to design improved schemes, which we tested for usability and security as well. In the process, we gained an understanding of the interactions and tensions between usability and security needs, as well as defining design strategies for knowledge-based authentication systems that address some of these unique challenges.

Chapter 3

Usability Evaluation of PassPoints

After our initial survey of graphical passwords, we believed that click-based graphical passwords offered the most promising alternative among those proposed so far. PassPoints was the most closely evaluated system in this category. It had a large theoretical password space, memorability appeared good, and entry times seemed reasonable. Wiedenbeck et al. [135–137] proposed PassPoints and conducted several in-lab user studies of their system. While initial results were optimistic with respect to usability, they acknowledged that further work was needed to address several remaining questions [136]. This included conducting a field study assessing the usability of PassPoints in a more realistic setting, examining whether hotspots (areas of the image that are more likely to be selected by users) cause security concerns, and looking at the effect of interference, i.e., whether having to remember multiple graphical passwords might cause memorability or usability problems.

To begin our work with usable authentication, we re-implemented and evaluated PassPoints. We conducted two user studies addressing the issues raised by the original PassPoints authors and re-examining earlier usability claims. Our first study was conducted in-lab to establish whether we could confirm the initial usability claims, look more closely at whether image choice had any impact, and gather click-point data. Secondly, we conducted a field study where 376 students used click-based graphical passwords to access their class notes during the Fall 2006 semester.

A number of our results differ materially from previous usability studies [135–137]. We found that participants were remarkably accurate in entering their passwords, indicating that tolerance regions as small as 9×9 pixels may be acceptable. It also appears that the type of image impacts memorability, with some images being too difficult to use. We further found that interference may be a problem. Participants who had two passwords (one on each of two images) had significantly lower success

rates than those who had only one. The work presented in this chapter has been published at the 2007 Symposium on Usable Privacy and Security (SOUPS) [15].

3.1 PassPoints Lab Study

We first conducted a lab study to independently evaluate the usability of PassPoints. We tested 17 different images with 43 participants, giving a range of 31 to 44 collected passwords on each image.

3.1.1 Methodology for the lab study

We used a web-based interface developed with PHP for this study. Our images were 451×331 pixels in size, the same dimensions as in the earlier PassPoints studies. The original PassPoints studies reported using a 20×20 pixel tolerance square, however it is unclear how this was implemented since it is impossible to accurately center a 20×20 square on a given pixel. We decided on a tolerance square of 19×19 pixels centered on the original click-point. In other words, confirm and login attempts where all points were less than 10 pixels in any x- or y- direction from their corresponding original click-points were considered successful.

Since we wanted to perform analysis on the passwords collected and the exact points selected, we did not use any discretization methods [9] nor hash the passwords before storing them. We simply recorded the exact coordinates of the click-points. As in the Wiedenbeck et al. studies, we used a Windows-based desktop computer with a 19-inch screen set at a resolution of 1024×768 pixels.

In our lab study, we tested 17 different images, shown in Figure 3.1. The images were selected to represent a variety in terms of level of detail, visual clutter, amount of colour, and content (landscapes, close-ups of objects, people, maps, etc.). Our set included the four images from the original PassPoints studies.

Participants created passwords on as many of these images as possible during their one-hour session. The number of images seen by each individual participant ranged from 9 to 17. In total, we collected a range of 31 to 44 passwords on each of the images. The maximum is greater than the total number of participants because some participants changed their password if they were unable to correctly re-enter it

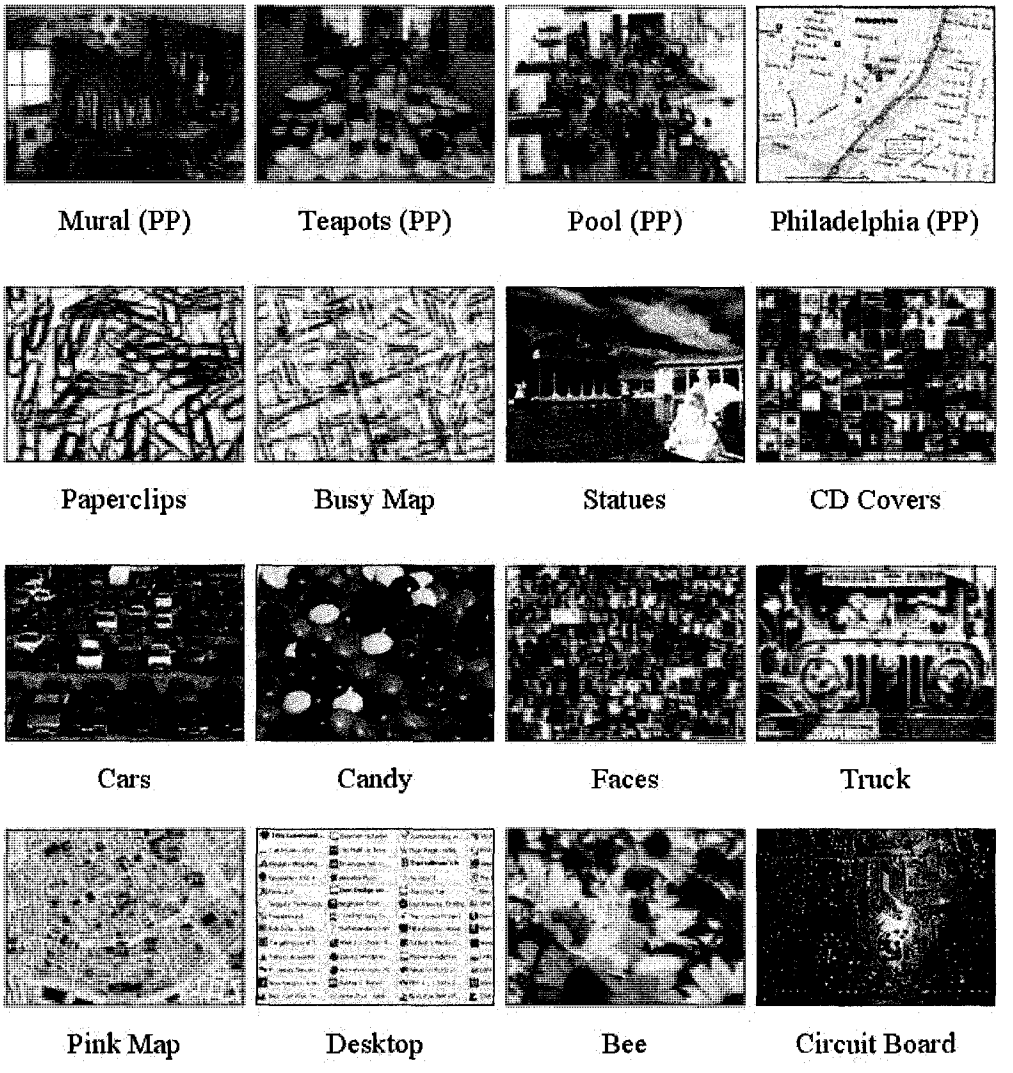


Figure 3.1: Images used in the lab study. Those denoted (PP) were used in the original PassPoints studies by Wiedenbeck et al. [135–137]. The 17^{th} image, Toys, is not displayed because it is copyrighted and we do not have permission to reprint it.

during login, and therefore, created multiple passwords on the same image (although only one was active at any given time).

Participants

Forty-three participants (25 females, 18 males) took part in this study. Data from two participants was eliminated because a malfunctioning mouse affected their performance. Our analysis considers data from the 41 remaining participants. All participants were university students from various degree programs, with an even mix of graduate and undergraduate students. Ten had technical backgrounds, but none were majoring in computer security. The average age of participants was 27 years. Thirty-seven reported using the web daily while the remaining four said they were online several times a week, so all were adequately experienced with using a computer and the web. Most participants (33) indicated that they were concerned about the security of passwords or that they took steps to reduce risks; 37 acknowledged reusing passwords. None had any experience with graphical passwords.

Task

Each participant completed a one-hour session in our usability lab. After completing the consent forms, they were introduced to the idea of graphical passwords. As part of this introduction, the experimenter showed them an image on the screen with a small superimposed square and explained that this was how accurate they needed to be with their mouse clicks when re-entering their passwords. They were advised to pretend that these passwords protected their bank information which meant that while they should pick something they could remember, they should also select passwords that would be difficult for others to guess so that no one could break into their account.

Each trial followed the steps described below. Steps 1, 2, and 5 represent the password phases on which analysis is reported later in this thesis.

1. Create Phase: Participants entered their username, selected a password by clicking five consecutive points on the given image, and clicked on the Login button. Their password consisted of these five points in the specified order.

2. Confirm Phase: The same image was presented a second time and users were asked to confirm their password. They once again entered their username and password then pressed the Login button.

3. Two-questions: After successfully confirming their password, the following screen asked two 10-point Likert-scale questions: "How easy was it to create a password on this image?" and "How difficult will it be to remember your password in one week?"

4. Mental Rotations Test (MRT): Users spent at least 30 seconds completing an MRT puzzle [92]. This was primarily intended to simulate the passage of time and work as a distraction to clear visual working memory. Psychology literature suggests that 15-30 seconds is ample time for this to occur [48].

5. Login Phase: Participants logged in using their previously created password.

If participants were unable to confirm their password or log in after 2 attempts, they were allowed to change their password (in effect returning to Step 1). If they strongly disliked the image or found it too difficult, they could skip this trial and move on to the next one.

The first two trials for each participant were considered "practice" trials, with the experimenter guiding users through the process and answering any questions they may have had to ensure that users understood the tasks. Data from these two trials were discarded during analysis. Participants then completed trials with as many images as possible in the remaining time, while working at their own pace. They were allowed to take breaks as needed between trials. After approximately half an hour, the experimenter interrupted, telling them to take a break and asking them to answer a demographics questionnaire. To avoid bias on any image due to inexperience or fatigue, the order of the images was randomly shuffled so that no two participants saw them in the same order.

At the end of the session, participants completed a post-test questionnaire. This questionnaire asked about their opinion of the system and graphical passwords then asked about their password selection strategy and the types of images they preferred.

Table 3.1: PassPoints success rate per phase (lab)

	Pool		Cars		All 17 Images	
Confirm	33/39	(85%)	31/33	(94%)	575/748	(77%)
Login	33/33	(100%)	30/32	(94%)	560/598	(94%)

3.1.2 Collected results for the lab study

Only 20 out of 41 participants had time to complete all 17 images, however since the order of the images was shuffled, we obtained at least 31 created passwords for each image. In total, data from 582 trials were analyzed. In some of the results reported here, we give primary focus to the Cars and Pool images (see Figure 3.6 and Figure 3.7) since these are the images used in the second study described in Section 3.2. Section 2.3.4 provides an explanation of the statistics used for analysis of the data in this, and subsequent, chapters.

Success Rate

Success rates were calculated as the proportion of all attempts that were successful for a given phase. A trial may have included multiple create, confirm, or login attempts if users made errors at any point or reset their password. As a result, there may be an uneven number of attempts in each of the phases. For example, a trial could consist of creating a password, failing to confirm it twice, resetting and creating a new password, then successfully confirming and logging in with the new password. This trial would have 2 create attempts, 3 confirm attempts, and 1 login attempt in total.

Taking all images into account, a total of 628 passwords were created. Of these, 35 passwords were created on the Pool image and 31 on the Cars image. Attempts at creating a password were all considered successful because the interface did not let users move on until they had clicked five points on the image, hence successfully creating a password.

The overall success rates for the Confirm and Login phases are provided in Table 3.1. Figure 3.2 shows the Confirm and Login success rates per image for each of the 17 images. There is considerable variation between images; in fact, statistically

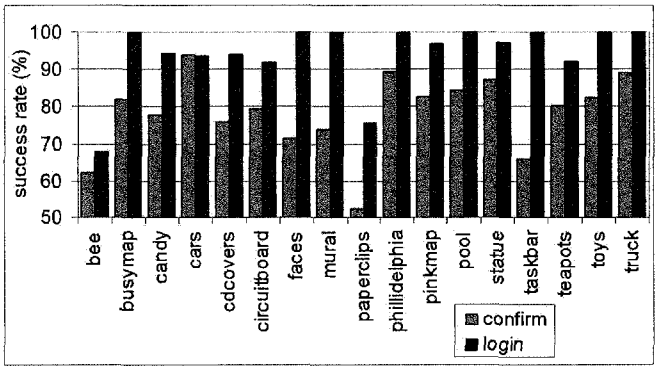


Figure 3.2: PassPoints success rate per image for each phase (lab)

significant differences between images are seen for both the Confirm ($\chi^2(16, N = 748) = 49.64, p < .001$) and Login ($\chi^2(16, N = 598) = 91.44, p < .001$) phases. For example, the Paperclips image had the worst success rate in the Confirm phase at 52% while the Cars image had a success rate of 94%. For the Login phase, the worst performer was the Bee image at 68% while several images reached success rates of 100%. This suggests that the choice of image can have substantial impact on usability, at least initially.

Two images had much lower success rates: the Bee and the Paperclips images (see Figure 3.1). These two images were also the source of most frustration and were most frequently skipped by participants in the Confirm or Login phases. The Paperclips image consisted of a random arrangement of coloured paperclips with no obvious patterns or distinguishing features. The Bee image was a close-up photo of yellow flowers with a single bee in the center of the image. Participants disliked this image, saying that it had no obvious "clickable" points other than the bee.

From these results, we are unable to predict whether Confirm and Login success rates for different images would converge after an initial learning curve. Success rates for the Confirm phase are generally lower than for the Login phase. This discrepancy may be due to the fact that the Confirm phase represents the first time users re-enter their password and as such they may have forgotten their points due to inattention, may have accidentally clicked on a different point than expected, or may remember the general area (such as "the red car") but not in precise enough detail ("the left front wheel of the red car") to accurately repeat the points. From participants' comments

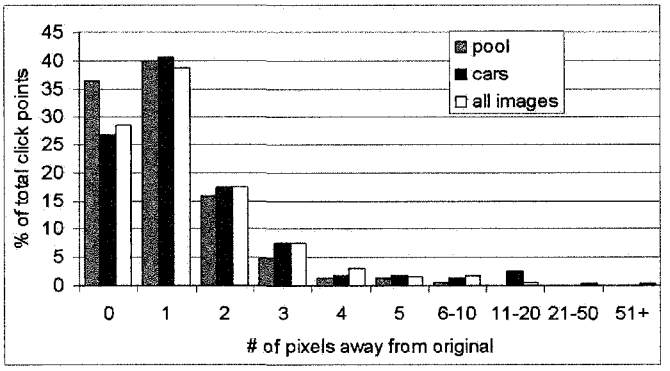


Figure 3.3: Accuracy for Login phase (lab)

and performance, the Confirm phase was part of the learning process; once they had successfully confirmed their password then they were more confident that they could repeat it during the Login phase. Several users stated that once they had confirmed their password successfully, then they knew it and even being distracted by the MRT did not affect their memory of it.

Accuracy

Participants were extremely accurate in targeting the points of their passwords. To determine accuracy, we analyzed individual click-points rather than looking at the password as a whole; each password contributed 5 data points. For each point, the maximum of $|x_{original} - x_{current}|$ and $|y_{original} - y_{current}|$ was taken as the measure of accuracy. All Confirm and Login attempts were considered in the analysis, even those that were unsuccessful.

In the Confirm phase, 96% and 94% of clicks on the Pool and Cars images respectively were within 4 pixels (1.5mm) of the original points. This means that click-points were accurate within a 9×9 pixel square. Participants were similarly accurate for the Login phase. Here, 98% of clicks were within 4 pixels for the Pool image and 94% for the Cars image. As an example, Figure 3.3 shows the distribution for the Login phase; the Confirm phase was very similar. There were slight variations, but overall participants were accurate on all images. Accuracy rates appear better than success rates because success rates are based on the entire Login/Confirm

attempt while accuracy rates consider individual click-points. One unsuccessful Login/Confirm attempt may have contributed four accurately entered click-points and only one incorrect click-point to the accuracy totals.

Times for Password Entry

As expected, it took much longer to create a password than to subsequently confirm it and log in, since participants had to initially look at the image and decide which points to select as part of their password. The total time to enter a password included typing a username (two-digits in this lab study), initial "think-time", clicking on five points, and clicking the Login button. Figure 3.4 summarizes the median total times for the Create and Confirm phases. Unfortunately, a technical glitch prevented us from gathering reliable total times for the Login phase although other timing data for this phase is reported below. We report primarily median times rather than means to avoid inflated numbers due to cases where participants stopped to comment during a trial. It also allows for comparison with our field study. The median total time for creating a password was 33 seconds (the mean time was 40 seconds), while the subsequent Confirm had a median time of 14 seconds (the mean time was 17 seconds). As shown in Figure 3.4, participants were quickest at creating passwords on the Truck image at 27 seconds while the Taskbar and Bee images took the longest at 42 seconds. During the Confirm phase however, times ranged only from 13 to 16 seconds.

Previous studies have found that graphical passwords take longer to enter than text passwords [115, 136], although results from the original PassPoints studies [135–137] show that PassPoints may be quicker than many other other graphical password schemes. To investigate whether this extra time is due to time taken to physically move the mouse and target the click-points in PassPoints, we also examined the "click time", i.e., the portion of time taken from the first click-point to the last click-point. Considering all images, it took a median time of 11 seconds to click on the five points during the Create phase, and 7 seconds during Confirm and Login. Figure 3.5 presents the median times for each phase on each image. Some images were obviously more difficult to use than others since participants took considerably longer to enter passwords on some of the images. As shown using ANOVAs, the differences in

Table 3.2: Differences between images in terms of timing (lab)

	ANOVA for Total time	ANOVA for Click time
Create	$F(16, 486) = 2.48, p < .01$	$F(16, 486) = 2.63, p < .001$
Confirm	n.s.	n.s.
Login	n/a	$F(16, 486) = 3.30, p < .0001$

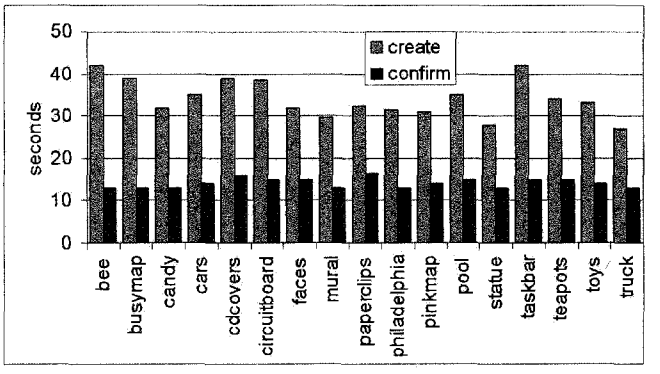


Figure 3.4: Median total times per phase (lab)

timings between images were statistically significant for the Create and Login phases (see Table 3.2); this indicates that the difficulty with some images occurred in both these phases.

Image Preference and Click-point Selection

Participants had strong opinions of which images they liked, and especially of those they disliked. Many voiced preference for images that had "clickable points" - small,

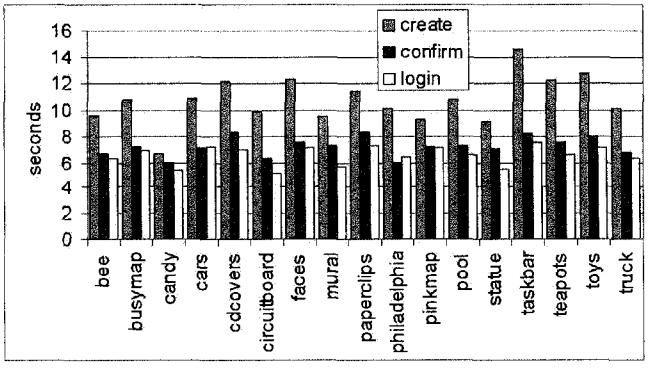


Figure 3.5: Median click-times per phase (lab)

distinct areas that could easily be identified and targeted with a mouse. Structural features such as lines, repeating items, and patterns seemed to be helpful. Many people also reported using letters or numbers if they appeared on the image.

They generally disliked images that were visually cluttered or that had too much repetition (such as the jumbled Paperclips or the close-up image of a uniformly coloured Circuit-board). They had trouble with the Bee image because it was mostly similar flowers and leaves with few distinct edges or distinguishing features. Most wanted to avoid clicking on the bee since it was "too obvious" but found little else that they thought they could accurately remember.

Many reported using patterns to select their click-points, for example geometric patterns such as "four corners and the middle" or contextual patterns such as "five red cars". Some used visible angles or intersections in the image and many selected objects of distinct colours. Points with personal meaning were often selected as well; one participant commented "I have to pick something that means something to me, if I just pick something at random, it'll be much harder to remember". There was a recurring theme of needing "clickable points", although exactly what made a point clickable varied.

3.1.3 Summary of lab study results

Overall, the login success rates are generally high (mean of 94% across all images) and the timings (median login click-time of 7 seconds) are reasonable, i.e., would appear to be quite acceptable for many login applications. There is little published research on comparable measures for text passwords. In a study [45] of text password variants following similar methodology, 19 participants with regular 8-character text passwords had a 94% login success rate if considering only their first login attempt, and a 98% success rate when considering a trial successful if users eventually entered their password correctly, regardless of how many attempts it required. The median login time for text passwords was 11 seconds. Given that users typically have many years of experience with text passwords and only a few minutes experience with PassPoints, we suggest that these differences are not unreasonable. With respect to accuracy, PassPoints participants performed extremely well, indicating that the

tolerance around the original click-points could potentially be reduced further than the 14×14 tolerance suggested by Wiedenbeck et al. [137] (see Section 2.4.5) without negatively affecting usability.

Our results indicate that the choice of image had a significant impact in all areas of usability. Besides the measurable aspects, some of the more difficult images led participants to sigh and sit back on their chair, just staring at the image, obviously frustrated at trying to select points.

3.2 PassPoints Field Study

To examine the effectiveness of PassPoints in a real-world setting, we conducted a field study where students used PassPoints to access their class notes during the Fall 2006 semester for 7 to 9 weeks.

3.2.1 Methodology for the field study

A web-based PassPoints system was built where students logged in to access the instructor's class notes. The system was available from mid-October to mid-December, with students logging on whenever they wanted to access their class material. Students who preferred not use a graphical password could opt-out and create a text password instead. In total, 376 students created graphical passwords and 25 created text passwords.

Students were introduced to the system through a combination of demonstrations during class time and tutorials, email instructions, and FAQ/Help documentation on the system's web page. We received only a handful of requests for technical support throughout the study.

The first time students accessed the system, they entered secondary identification information, created a secret question in case they needed to change their password, and proceeded to create and confirm their PassPoints password on an assigned image. A small square directly above the image reminded them of the accepted tolerance for their points. Passwords consisted of an ordered series of five unique points, as in our lab study.

Participants

Students from three first-year undergraduate Computer Science (CS) classes were invited to participate in this study. One class was for students who were not CS majors while the other two were primarily for students intending to major in CS. We received consent from 191 unique students to use their data in our study (124 CS students and 65 non-CS students). Of these, 37 students were in two of the classes and had two different accounts (with different images). Therefore we have data from 228 different accounts. These 228 accounts will be used for all further analysis.

Study Design

A two-dimensional between-participants design was used (see Table 3.3). Participants were randomly assigned to different experimental conditions with no consideration given to which class they were enrolled, except in the cases where participants were in two classes. Both the image and the required accuracy were varied. One group was given a tolerance square of 13×13 pixels and the other a tolerance of 19×19. The 19×19 square was consistent with our lab study. Students who were in both CS classes were assigned a different image for each class but the size of their tolerance square was kept consistent. Only two images were selected from our earlier lab study: the Pool and Cars images (Figures 3.6 and 3.7). These images had reasonable usability results and differed in their number of hotspots based on a separate security analysis by our colleagues [119]. The Pool image contained several intense hotspots while the Cars image did not. The Pool image was also selected because we wanted to test one of the original PassPoints images.

The number of participants per group is given in Table 3.3. The sizes of the experimental groups are uneven because participants were assigned to groups at the beginning of the study, before we knew who would give consent to use their data.

The two images were the same size as in previous studies, namely 451×331 pixels. However, since students were allowed to log in from anywhere with web access, we could not control screen size or resolution. Similarly to the lab study, the system stored passwords and user input in the clear so that we could analyze the passwords selected and the types of errors made by users as they tried to log in. Although

Figure 3.6: The Cars image [11]

Figure 3.7: The Pool image [90]

necessary to collect detailed data, this design decision has security implications, so we opted to protect only general class notes with PassPoints and not any personal or private information.

At the end of the semester, we asked students to complete an online questionnaire. The questionnaire included demographic questions and questions about their perception and opinion of click-based graphical passwords. This data was used only for post-hoc analysis and to informally provide insight for future designs and hypotheses.

Table 3.3: Number of students per experimental condition (field)

	13 × 13 Tolerance	19 × 19 Tolerance
Pool image	63	53
Cars image	61	51

Table 3.4: Attempts per participant for each phase (field)

	Create	Confirm	Login
Mean	2.6	3.6	18
Median	2	2	15
Maximum	11	17	65

3.2.2 Collected results for field study

Table 3.4 summarizes the usage data for the field study. Participants attempted to login an average of 18 times throughout the semester and created an average of 2.6 passwords (i.e., changed their password 1.6 times). Usage was relatively consistent throughout the entire semester. The student who attempted to login most frequently did so 65 times. It should be noted that these numbers take into account all attempts, including those that were unsuccessful. As users who chose to use text passwords had opted-out of the study, we do not have comparison data for text passwords.

Success Rate

Participants were allowed to change their passwords at any point, provided that they entered their secondary identification information and answered their preset secret question. For the purpose of our analysis, change password attempts are treated the same as original Create attempts since the result in both cases is a new password. Once again, an attempt to create a password was only accepted once five click-points were selected, so 100% of attempts to create a password were considered successful. In total, 265 passwords were created for Pool and 216 for Cars. Of these, 149 (56%) were a result of changing a password on the Pool image in comparison to 104 (48%) for the Cars image. On the Pool image, 49% of participants created only one password and 18% created four or more. Of those using the Cars image, 43% kept the same password all term while 11% created four or more passwords. We did not ask users why they were changing their password; possible reasons may include forgetting their password or testing out the system by trying various passwords since this was a novel password system for these users.

Table 3.5: Success rate per phase (field)

	Pool image successful / total attempts		Cars image successful / total attempts	
Confirm	207/388	(53%)	170/293	(58%)
Login	1461/1880	(78%)	1301/1563	(83%)

Success rates for the field study were calculated as the number of successful attempts across all attempts for a given phase. We decided that this was a more representative measure than calculating success rates on a per participant basis since a participant who logged in only once throughout the term could have a success rate of 100% which is rather misleading. Overall success rates for both images are provided in Table 3.5. There was no statistically significant difference in success rates between users of the two images during the Confirm phase. During Login, however, users of the Cars image had higher success rates than those who had the Pool image, and the difference was statistically significant ($\chi^2(1, N = 3443) = 16.42, p < .001$), perhaps indicating that the choice of image does affect the memorability of passwords over time. Here, the success rates seemed to indicate that Cars was more memorable than Pool.

Success rates were considerably lower than in the lab study. Upon closer examination of the Login attempts, we found that success rates did improve with practice, although never reaching the levels attained in the lab study. For example, the initial success rate across all students was 76%, rising to 88% when considering only login attempts beyond the 30th attempt for students who logged in at least 30 times.

Accuracy

Participants were once again remarkably accurate in entering their passwords. As with the lab study, we analyzed click-points individually rather than looking at whole passwords.

As shown in Figure 3.8, 78% of clicks on the Pool image for the Login phase were within 4 pixels (approximately 1.5mm, although this varied with the specific screen resolution and screen size used which was no longer in our control) of the original point (i.e., within a 9×9 pixel tolerance square), while 80% of clicks on the Cars

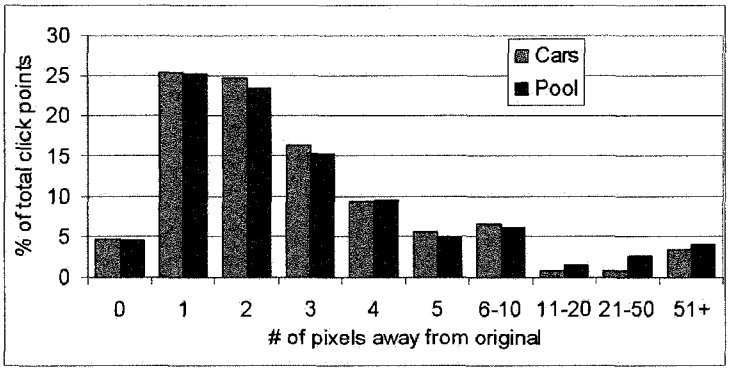


Figure 3.8: Accuracy for Login phase (field)

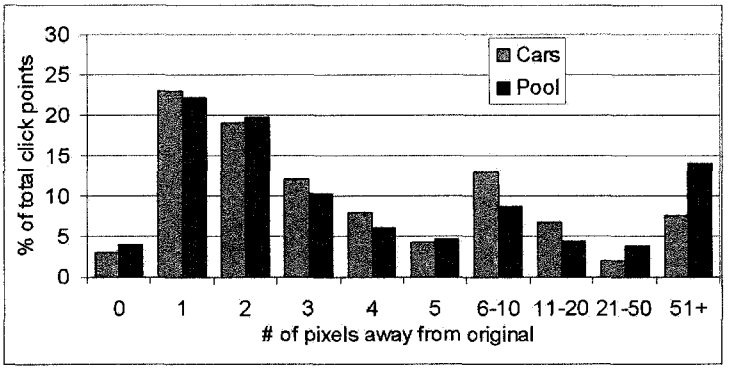


Figure 3.9: Accuracy for Confirm phase (field)

image fell within 4 pixels. Assuming that clicks further than 50 pixels away were forgotten points, only 4% and 3% of points were forgotten for the Pool and Cars images respectively.

Looking at Figure 3.9, it is apparent that confirming the password is part of the learning process as participants were considerably less accurate in entering their passwords than in the Login phase (reported above). For the Confirm phase, 62% and 65% were within 4 pixels (i.e., within a 9×9 pixel tolerance square) for the Pools and Cars images respectively. People were also more likely to forget their points altogether in the Confirm phase: 14% of points were forgotten on the Pool image and 8% of points were forgotten on the Cars image during Confirm.

There is no statistically significant difference in terms of accuracy between the two images for the Confirm phase. During the Login phase, we found a higher degree

Table 3.6: Effect of size of tolerance square on success rate (field)

		13×13 Tolerance		19×19 Tolerance		χ^2
Confirm	Pool	126/245	(51%)	81/143	(57%)	n.s.
	Cars	95/170	(56%)	75/123	(61%)	n.s.
Login	Pool	790/1018	(78%)	671/862	(78%)	n.s.
	Cars	640/790	(81%)	661/773	(85%)	$\chi^2(1, N = 1583) = 5.67, p < .05$

of accuracy for the Cars image than the Pool image ($U = 3.01, p < .01$) [1]. This result relates to the login success rates discussed in the previous section. Users who fail to login, by definition, have entered at least one inaccurate click-point; since the login success rate for the Pool image is lower, we would expect that the accuracy for the Pool image to also be lower.

Effect of Tolerance Square Size

Since participants were so accurate in entering their passwords, the size of the tolerance square had little impact on success rates. For the Pool image, having different sized tolerance square had no impact on the success rates for either the Confirm or Login phases (see Table 3.6). The Cars image similarly showed no difference for the Confirm phase, but for the Login phase participants were significantly more likely to succeed with the larger 19×19-pixel square; however both tolerances still had success rates of above 80%.

Interestingly, participants were more accurate in entering their click-points during the Login phase when they had a smaller tolerance square. Telling them that they needed to be accurate actually improved their accuracy in the field while having little impact on their success rates. As accuracy distributions are similar to those reported for the lab study, only the number of click-points within 4 pixels is reported in Table 3.7 although the Mann-Whitney tests take the entire data set into account.

To further examine whether the size of the tolerance square had an effect on performance, we looked at the click-time from the first to last point. If those who had a smaller tolerance square were actively trying to be more careful in targeting, we

[1] The non-parametric Mann-Whitney test was used because the distributions were skewed, and therefore normal distributions could not be assumed.

Table 3.7: Effect of size of tolerance square on accuracy (field)

		13 × 13 Tolerance: <= 4 pixels		19 × 19 Tolerance: <= 4 pixels		Mann-Whitney
Confirm	Pool	781/1225	(64%)	431/714	(60%)	n.s.
	Cars	549/850	(65%)	405/615	(66%)	n.s.
Login	Pool	4174/5090	(81%)	3164/4305	(73%)	$U = 13.60, p < .001$
	Cars	3289/3950	(84%)	3008/3860	(78%)	$U = 5.14, p < .001$

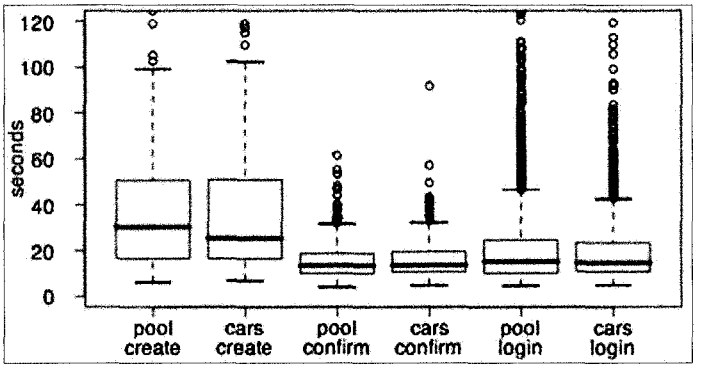


Figure 3.10: Median total times per phase (field)

would expect to see increased click-times. However we found no statistical differences in the click-times between the two tolerance groups for either image, further indicating that participants' performance was not impacted by having a smaller tolerance square.

Times for Password Entry

Participants were able to create their passwords relatively quickly, with a median total time of 25 seconds for Cars and 30 seconds for Pool. Total times for the Confirm and Login phases were surprisingly consistent, with median times varying between 13 and 15 seconds across both phases. Figure 3.10 presents the total times for each phase of the Cars and Pool images using Box-and-Whisker graphs. The box indicates the Inter-Quartile Range (IQR - the interval between the 25th and 75th percentiles) while the whiskers represent the first and fourth quartiles respectively. The thick line within the box indicates the median time for each phase. Outliers are values beyond the whiskers that lie further than $1.5 \times IQR$ from either end of the IQR box. Outliers are shown as empty circles.

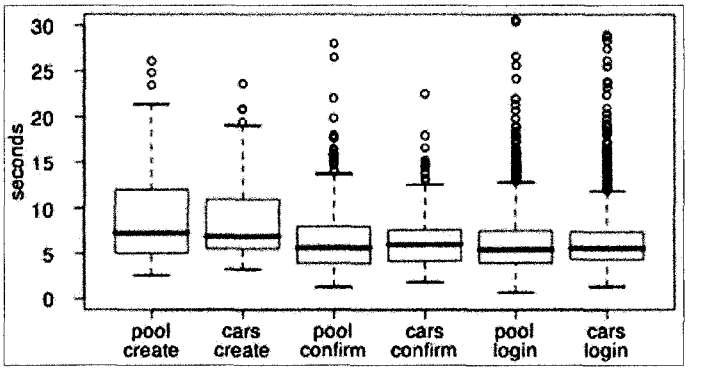


Figure 3.11: Median click-time per phase (field)

Mean times did not provide an accurate snapshot of the data in this case due to outliers with very high times. For example, a few Login attempts were measured in days rather than seconds. Since participants were not using the system in a controlled setting, they may have opened the login page, turned their attention elsewhere, and later returned to continue logging in. For this reason, median times are more representative.

As shown in Figure 3.11, participants were very quick in actually targeting and entering their click-points. When considering only the median click-time from the first to the fifth point, the Create phase took 7 seconds, while the Confirm and Login phases had median click-times of between 5-6 seconds.

Interference

Having multiple passwords affected performance. Students who had two passwords had higher success rates in the Confirm phase (statistically significant for the Pool image; see Table 3.8). It appears that the extra practice at creating and confirming a password improved their performance.

During the Login phase however, interference negatively affected success rates. Students were more likely to log in correctly when they only had one password to remember. As shown in Table 3.8,[2] the difference in success rates due to the presence or lack of interference is statistically significant for both images. For example,

[2]One participant had the same image for both classes. His data is excluded from our analysis of interference.

Table 3.8: Effect of interference on success rate (field)

		No Interference		Interference		Statistical Test
Confirm	Pool	139/284	(49%)	63/99	(64%)	$\chi^2(1, N = 383) = 6.36, p < .05$
	Cars	108/193	(56%)	62/100	(62%)	n.s.
Login	Pool	1224/1541	(79%)	226/319	(71%)	$\chi^2(1, N = 1860) = 11.33, p < .001$
	Cars	1053/1216	(87%)	248/347	(71%)	$\chi^2(1, N = 1563) = 44.26, p < .001$

those who only had a password on the Cars image had a success rate of 87% but those who had two passwords had a success rate of only 71%. This indicates that having to remember two unique passwords on different images negatively affects long-term memorability; this finding is troublesome if graphical passwords were to become widely used. However, in a more recent study [18], discussed in Section 9.3, we have found that multiple password interference was significantly worse for text passwords than for PassPoints.

We examined more closely the data from the interference group. Specifically, we looked at the initial password created on each image to see whether users' ability to confirm their password improved for the second image since they had already practiced the process with the first image. Looking only at the initial password created on each image, we uncovered that students had higher success rates for the Confirm phase for their second image (67% success rate) than on their first image (60% success rate). However, the difference did not reach statistical significance.

Usability versus Security

Our colleagues [119] carried out a security analysis of hotspots within the images and examined whether passwords created by a small subset of users can be leveraged to generate a successful attack against other users. Hotspots are areas of the image that are more likely to be selected by users as part of their password. Collected passwords from 35 users (Pool image) and 33 users (Cars image) in the lab study were used to determine hotspots, from which a dictionary of candidate passwords was generated. The dictionary entries were then compared to the set of final user-created passwords in the field study (i.e., if users changed their passwords during the semester, only the latest password was examined), after removing any passwords where users failed to log in at least once. The rationale for examining this subset was that final passwords may

be more indicative of what people would eventually select as memorable passwords.

The results are worrisome from a security viewpoint: the attack [119] correctly guessed 41/112 (Pool image) and 22/109 (Cars image) passwords (hotspot dictionaries, and also pattern dictionaries, are discussed further in Section 8.2). We focus on the usability implications of these results and examine whether those passwords that were easily guessed are also those that are most memorable. Taking into account all login attempts for the tested passwords, we see a statistically significant difference in the success rates between those passwords that were cracked and those that were not ($\chi^2(1, N = 2781) = 4.67, p < .05$). Contrary to our expectations, however, the guessed passwords actually had a lower login success rate (84%) than those that were not guessed (88%).

If success rate is taken as a measure of memorability, our small sample indicates that more memorable passwords (as measured by login success rate) were not any easier to guess than less memorable passwords. However, a larger sample or different attack strategies may reveal different results.

3.2.3 Summary of field study results

Most people chose to use their graphical passwords throughout the semester rather than opting-out and selecting a text password, something we found encouraging in terms of usability. However the lower success rates and accuracy results when compared to the lab study are a cause for concern. As noted earlier, login success rates did improve over time which may indicate that with continued usage, users may reach levels of expertise similar to text passwords.

The effect of interference is cause for concern since it is likely that in a real-world setting, people would have more than one password. Independent of interference, it is likely that users would resort to coping strategies that would further weaken security as they do with text passwords. In response to open-ended questions on the end-of-term questionnaire, many reported that they would be more likely to use geometric patterns to try and have similar passwords on each image. Our later examination of patterns is discussed in Chapter 7 (see also Section 8.2 for pattern dictionaries). We show that the security of PassPoints is questionable since many passwords do follow

simple geometric patterns. We expect that the passwords guessed in attacks based on such patterns would correlate with those passwords that have higher success rates and that are more memorable.

Interference is discussed by Wiedenbeck et al. [136] and by Monrose and Reiter [77] as a potential concern; our field study provides the first empirical evidence that interference is in fact a problem. The results of our more recent work on interference [18] is summarized in Section 9.3.

3.3 Discussion

The usability results of our two studies revealed interesting differences. The lab study provided much more positive results than the field study, calling into question the validity of only conducting lab studies for security interfaces, although they are definitely an important first step.

As shown in Table 3.9, there were statistically significant differences in the success rates and accuracy results between the lab and field studies, with less favourable results for the field study in both cases. This indicates that the lab study is not a reliable predictor of these usability aspects. With respect to password-entry times however, the field study had similar or shorter times than the lab study. For example, click-times were shorter for the Login phase in the field study (mean of 5.5 seconds) than in the lab study (mean of 7 seconds), a result that is statistically significant ($t(3403) = 2.02, p < .05$).

There are a few possible reasons for the discrepancies between the lab and field studies. The lab study gave participants more concentrated practice with creating and confirming passwords since they performed these tasks several times within an hour. Our lab participants also had two "practice" trials where they could ask questions and become accustomed to the process before starting the real trials. In contrast, participants in the field study received an explanation and instructions, but did not have a chance to rehearse on practice images before attempting to create and confirm their real password. We felt that requiring participants to create "practice" passwords before creating their own was impractical in a realistic setting and this may partially

Table 3.9: Differences in success rate and accuracy: lab vs. field study

		Success Rate χ^2	Accuracy Mann-Whitney
Confirm	Pool	$\chi^2(1, N = 427) = 14.07, p < .001$	$U = 13.81, p < .001$
	Cars	$\chi^2(1, N = 326) = 16.19, p < .001$	$U = 10.74, p < .001$
Login	Pool	$\chi^2(1, N = 1913) = 9.42, p < .001$	$U = 13.64, p < .001$
	Cars	n.s.	$U = 10.47, p < .001$

account for the discrepancies in success rates when compared to the lab study. However our analysis also showed that while field study success rates did improve with practice, they still did not reach the levels observed in the lab study. Another factor that may have affected performance was that users in the field study could log in from any computer and different system configurations may have had an impact. Users with laptops, for example, were likely using smaller screens set at higher resolution than users from our lab study.

Secondly, the Login phase for the lab study occurred shortly after the password was created and confirmed. Although we attempted to distract participants with MRT puzzles, the immediacy of logins likely contributed to the high success rates. As logins for the field study spanned across several weeks, participants had ample time to forget their password between login attempts.

Finally, passwords were the focus of the lab study. Participants were actively engaged in the process and it was their primary task. In the field study, the primary task was accessing class notes, while logging on was a secondary task. This shift to a secondary task likely affected the amount of attention paid to the task and the importance accorded to getting it correct, even though errors hindered progress towards the primary task. This also likely partially accounts for the faster click-times as participants were trying to quickly move on to accessing their class notes.

3.4 Conclusion

In this chapter, we present the results of two usability studies of PassPoints. The initial lab study revealed mostly positive results and led to a larger field study to see how PassPoints worked in practice.

The lab study confirmed earlier work that the usability of these passwords was good in terms of success rates and password-entry times. We additionally showed that participants were more accurate in targeting their click-points than previously suggested, indicating that smaller tolerance squares may be acceptable. Finally, contrary to previous work, we found that the choice of image significantly influenced success rates.

The field study represented the first large-scale, real-world study of click-based graphical passwords presented in the literature. Password entry times were acceptable, accuracy was not quite as high as in-lab but still very good, success rates improved with practice, and participants continued to use the system even though they could easily have switched to a text password. However, we found several legitimate concerns with adopting PassPoints as a means of authentication. We provided the first empirical evidence that interference from having to remember multiple graphical passwords is problematic. Participants also reported using patterns in selecting their passwords, suggesting increased susceptibility to guessing attacks.

The differences between the lab and field studies also raise methodological concerns in usable security. So far, lab studies are the most common form of usability evaluation and while others have cautioned that these were inadequate in providing realistic usability data, our two studies provide empirical evidence of this problem. We do not suggest that lab studies be eliminated. They offer a relatively quick and cost-effective way to test new ideas to find which are more promising by allowing for initial usability and security evaluations. However we caution that real usage may vary from lab usage and that field studies are an important second step to confirm results found in the lab.

These user studies of PassPoints form the basis for several parts of the remaining work in this thesis. We have conducted further analysis examining geometric patterns within passwords and the clustering of click-points; these results are presented in Chapters 5 and 7. We have also focused on improving the memorability and security of click-based graphical passwords through the design and evaluation of new schemes presented in Chapters 4, 5, and 6.

Chapter 4

Cued Click-Points

Since our initial user studies on PassPoints, several publications [35, 50, 101, 119, 126] have discussed the issue of "hotspots" in PassPoints. Hotspots are areas on the image that users are more likely to select; they are tied to the background images used, the nature of the password selection task (such as having to select 5 points on one image), and the degree of user choice during password selection. If this phenomenon is too strong, the likelihood that attackers can guess a password significantly increases.

Security analyses show that it would be possible for attackers to discover hotspots and use this information to successfully mount an attack against PassPoints passwords in a reasonably short time. Thorpe and van Oorschot [119, 126] show that dictionary attacks can crack a significant number of passwords with a relatively small dictionary for PassPoints, using a dictionary based on either sample passwords collected from actual users or likely hotspots as determined through automated image processing techniques. Dirik et al. [35] also had some success using automated image processing to guess PassPoints passwords; see also Salehi-Abari et al. [101]. Furthermore, Golofit [50] manually categorized different areas of three images based on prominent features (e.g., flat, structural, commonplace, block edges) and shows that user-selected click-points cluster within the areas of the images categorized as "commonplace" or "block edge" based on his classification scheme.

To partially address the problem of hotspots and further improve the memorability of click-based graphical passwords, we propose a new click-based graphical password scheme called Cued Click-Points (CCP). A password consists of one click-point per image for a sequence of images. The next image displayed is based on the previous click-point so users receive immediate *implicit feedback* as to whether they are on the correct path when logging in. As explained herein, CCP offers both improved usability and security.

We conducted an in-lab user study with 24 participants and a total of 257 trials. Users had high login success rates, could quickly create and re-enter their passwords, and were very accurate when entering their click-points. Participants indicated that they preferred CCP to a PassPoints-style system. They also said that they appreciated the immediate implicit feedback signalling them whether their latest click-point was correctly entered.

A preliminary security analysis of this new scheme is also presented in this chapter. CCP uses a large set of images that will be difficult for attackers to obtain. For our proposed system, hotspot analysis requires proportionally more effort by attackers, as each image must be collected and analyzed individually. CCP appears to allow greater security than PassPoints because the workload for at least some phases of attacking CCP can apparently be proportionally increased by augmenting the number of images in the system. As with most graphical passwords, CCP is not intended for environments where shoulder-surfing is a serious threat. The work presented in this chapter was published at ESORICS 2007 [21].

4.1 Cued Click-Points (CCP)

Cued Click-Points (CCP) is our first proposed alternative to PassPoints. In CCP, users click one point on each of $c = 5$ images rather than on five points on one image. It offers *one-to-one cueing*, where each image acts as a cue for the one corresponding click-point, and introduces *implicit feedback*, where visual cues instantly alert legitimate users if they have made a mistake when entering their latest click-point (at which point they can cancel their attempt and retry from the beginning). It also makes attacks based on hotspot analysis more challenging, as we discuss later. As shown in Figure 4.1, each click results in showing a next-image, in effect leading users down a "path" as they click on their sequence of points. A wrong click leads down an incorrect path, *with an explicit indication of authentication failure only after the final click*. Users can choose their images only to the extent that their click-point dictates the next image. If users dislike the resulting images, they may create a new password involving different click-points to get different images.

We envision that CCP fits into an authentication model where a user has a client

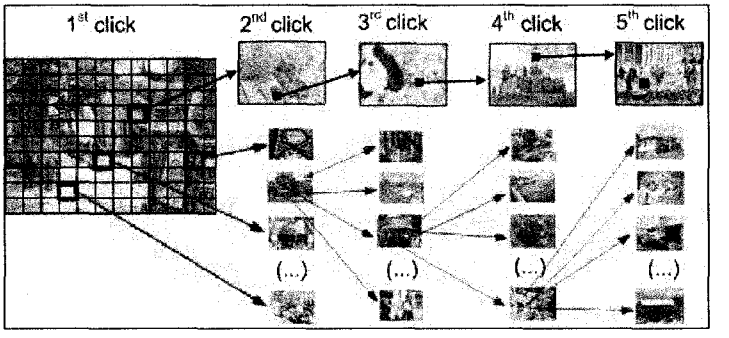


Figure 4.1: CCP passwords can be regarded as a choice-dependent path of images

device (which displays the images) to access an online server (which authenticates the user). We assume that the images are stored server-side with client communication through SSL/TLS. For further discussion, see Section 4.4.

For implementation, CCP initially functions like PassPoints. During password creation, a discretization method (e.g., robust discretization [9] or centered discretization [19], discussed in Chapter 6) is used to determine a click-point's tolerance square and corresponding grid. For each click-point in a subsequent login attempt, this grid is retrieved and used to determine whether the click-point falls within tolerance of the original point. With CCP, we further need to determine which next-image to display.

Similar to our PassPoints studies, our example system had images of size 451×331 pixels and tolerance squares of 19×19 pixels, giving 432 squares per grid. We note that our calculation for the number of squares per grid differs from that of Wiedenbeck et al. [136] because their calculation assumed that tolerance squares were 20×20 pixels and does not account for partial squares on the edges of the image. To uniquely map each tolerance square to a next-image, we use a function

$$f(username, currentImage, currentToleranceSquare).$$

The *currentImage* and *currentToleranceSquare* are identifiers for the current image and the grid square corresponding to the user's most recently entered click-point, respectively. This suggests a minimum set of 432 images required at each stage. One argument against using fewer images, and having multiple tolerance squares map to the same next-image, is that this could potentially result in misleading implicit feedback in (albeit rare) situations where users click on an incorrect point yet still see the correct next-image.

Each of the 432 next-images would have 432 tolerance squares and thus require 432 next-images of their own. If we map each possible grid-square to a unique image, the number of images would quickly become quite large. So we propose re-using the image set across stages. By re-using images, there is a slight chance that users see duplicate images. During the 5 stages in password creation, the image indices $i_1, ..., i_5$ for the images in the password sequence are each in the range $1 \leq i_j \leq n$, where n is the total number of images in the set. When computing the next-image index, if any is a repeat (i.e., the next i_j is equal to i_k for some $k < j$), then the next-image selection function f is deterministically perturbed to select a distinct image so that users do not see a duplicate image.

A user's initial image is selected by the system based on some user characteristic (as an argument to f above; we used *username*). The sequence is re-generated on-the-fly from the function each time a user enters the password. If a user enters an incorrect click-point, then the sequence of images from that point onwards will be incorrect and thus the login attempt will fail. For an attacker who does not know the correct sequence of images, this cue will not be helpful. As previously mentioned, shoulder-surfing is a concern with CCP (and other click-based graphical password systems), and our discussion focuses primarily on attackers who are not in a position to observe or capture login information from the legitimate user. However, it should be noted that obtaining only the sequence of images does not provide enough information to log in directly; considerable additional effort is required to identify where to click on the images to obtain this sequence. Further security discussion is provided in Section 4.4 and Chapter 8.

We expect that hotspots will appear as in PassPoints, but analysis will require more effort because the number of images is significantly increased; this increase varies proportionally with the configurable number of images in the system. For example, if attackers identify 30 likely click-points on the first image, they then need to analyze the 30 corresponding second images (once they determine both the indices of these images and get access to the images themselves), and so on, growing exponentially.

A major usability improvement over PassPoints is the fact that legitimate users get immediate feedback about an error when trying to log in. When they see an incorrect

image, they know that the latest click-point was incorrect and can immediately cancel this attempt and try again from the beginning. The visual cue does not explicitly reveal "right" or "wrong" but is evident using knowledge only the legitimate user should possess. Text passwords and PassPoints can only safely provide feedback at the end and cannot reveal the cause of error. Providing explicit feedback in PassPoints before the final click-point could allow PassPoints attackers to mount an online attack to prune potential password subspaces, whereas CCP's visual cues should not help attackers in this way. Another intended usability improvement is that being cued to recall one point on each of five images appears easier than remembering an ordered sequence of five points on one image. Each image triggers the memory of one click-point and there is no need for users to remember the order of the click-points.

4.2 CCP Lab Study

We conducted an in-lab user study of CCP with 24 participants. The methodology was identical to the methodology for our PassPoints lab study (Section 3.1.1) other than modifications to the instructions to explain how CCP worked rather than PassPoints. The participants (12 females and 12 males) were university students with diverse backgrounds. None were specifically studying computer security but all were regular web users. They ranged in age from 17 to 26 years. In total, 257 CCP trials were completed.

When time remained in the one-hour session, participants were given one further task: to complete a trial with our earlier PassPoints system, where they selected five points on one image. The experimenter was careful to identify the second system as "the other system we are looking at" rather than the "old" system, to not bias participants into thinking that they should rate CCP more favourably. Users were then asked which version they preferred.

A prototype application was developed in J#. A set of 330 images was compiled from personal collections as well as from websites providing free-for-use images. The prototype system did not hash the passwords or use a discretization method as would a real system, but simply stored the exact pixel coordinates of each click-point so that the users' choice of click-points and accuracy on re-entry could be examined. The

system also implemented an improvised image selection process to reduce the size of the required image set, since with several unique trials per participant we would have needed several thousand images to implement the proposed scheme since each trial would require at minimum 432 images. Furthermore, with the number of users participating in a lab study, we would not have been able to collect enough click-point data on each image to allow for reasonable analysis of hotspots if the click-points were distributed across thousands of images. The first time a user clicked on a point, a new image was associated with that point. If a user clicked within the tolerance region of that point again, either for re-entering or for resetting a password, the same image was shown. Once associated with a click-point, an image was not re-used for any other click-point during the entire session. The software prototype was built such that only areas where the user clicked had images associated with them, thereby reducing the total number of images required while still behaving in a manner consistent with the actual proposed scheme from the user's perspective.

4.3 Collected Results

4.3.1 Success rates and restarts

Although it occurred infrequently, users were allowed to restart (similar to pressing the backspace key in text passwords) if they changed their mind while creating a password. This accounts for the restarts listed in Table 4.2 for the Create phase.

During the Confirm and Login phases, participants typically used the reset button as soon as they saw an incorrect image and realized they were on the wrong path. This effectively cancelled the current attempt and returned them to the first image where they could start entering their password again. A few times, participants restarted even when they saw the correct image because they had forgotten the image. Failed login attempts (where users pressed the login button and were explicitly told that their password was incorrect) occurred only when users clicked on the wrong point for the last image since they did not receive any implicit feedback for that click-point. Because these were so few, failed login attempts are included in the restart counts since ultimately both failed login attempts and restarts are considered incorrect entries.

Table 4.1: Success rates for CCP on first attempt (over 257 trials). Only trials where the password was entered correctly on the first attempt, with no restarts, are considered successful.

	Create	Confirm	Login
Success Rates (first attempt)	251/257 (98%)	213/257 (83%)	246/257 (96%)

Table 4.2: Total number of restarts for CCP over 257 trials (note that it was possible to restart multiple times per trial)

	Create	Confirm	Login
Total Number of Restarts	7	101	14

Success rates were calculated as the number of trials completed without errors or restarts over the total number of trials. Our method of calculating success rates per attempt, as used in the PassPoints studies, needed modification to reflect the additional interim feedback provided by CCP that allowed users to determine if their partially entered password was correct. With CCP, users restarted on their own each time they thought they had made a mistake, and very rarely pressed the login button before their password was entered correctly. To avoid misrepresenting the success rates, we moved to calculating success rates "per trial", where a password was considered successful only if entered correctly *on the first attempt, with no restarts or errors.*

Participants said that confirming the password helped them to remember it and that it was part of the learning process. Once they had successfully confirmed the password, logging on even after the distraction task was relatively easy. This fact is reflected in Table 4.2 which shows that the vast majority of restarts occurred during the Confirm phase.

Four participants completed all of their trials without any restarts, i.e., they made no errors during the entire session. In total, 201 of 257 trials (79%) were completed without restarts in any phase. The success rates were high for all phases, as shown in Table 4.1.

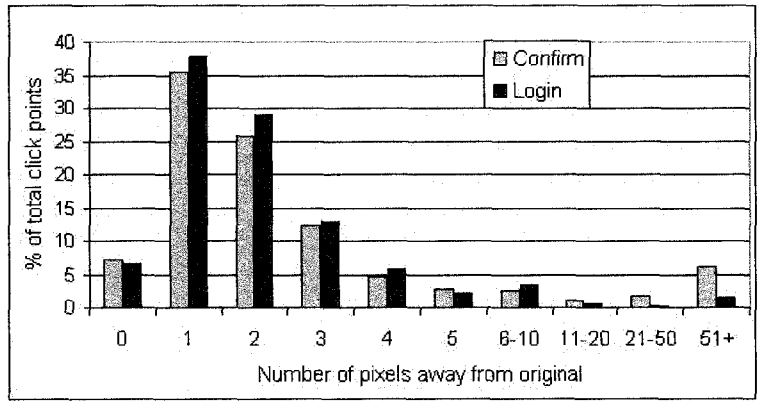


Figure 4.2: CCP accuracy for the Confirm and Login phases

4.3.2 Accuracy

Although CCP participants were less accurate in re-entering their passwords than our PassPoints participants from Chapter 3, the accuracy rates remain quite high. As a measure of accuracy, all individual click-points in the Confirm and Login phases were evaluated. This totalled 1569 click-points for the Confirm phase and 1325 click-points for the Login phase. For each point, the accuracy was computed as the maximum of $|x_{original} - x_{current}|$ and $|y_{original} - y_{current}|$. All click-points were considered in the analysis, even those that were unsuccessful. A few times, participants reached an incorrect image and still proceeded to click on a point. These were included in the 51+ category since the point was obviously forgotten. As indicated in Figure 4.2, 86% of points were within 4 pixels of the original click-point for the Confirm phase compared to 92% for the Login phase. Falling within 4 pixels of the original point means that these click-points would have been accepted within a tolerance square of 9×9 pixels. The lower accuracy during Confirm compared to during Login reflects the same pattern as seen for the success rates (Section 4.3.1) since inaccurate click-points lead to incorrect password entries.

4.3.3 Times for password entry

As expected, participants took longest to create their password and then were progressively quicker in entering it during the Confirm and Login phases. The reported

Table 4.3: Times for password entry per phase for CCP, in seconds

	Create	Confirm	Login
Mean Time (SD)	24.7 (16.4)	10.9 (13.1)	7.4 (5.5)
Median Time	19.1	7.4	6.0

times encompass from the first click in a phase until the last click (i.e., the "click-time"), including any restarts. The mean and median times reflected in Table 4.3 are slightly elevated because some participants paused to comment as they were entering their password, which slowed their performance. Despite this fact, the median login time is 6 seconds and the total time to create and confirm a CCP password is approximately half a minute, which we expect would be quite acceptable in many applications or environments. In comparison, a similar study requiring the creation of regular 8-character text passwords [45] shows a median time of 35 seconds for the password creation and confirmation combined.

4.3.4 Preference between CCP and PassPoints

When time permitted, participants were introduced to a PassPoints system and asked which they preferred. Ten participants attempted a trial with the PassPoints system. Of these 10 people, 7 strongly preferred CCP, one preferred PassPoints, and two felt that PassPoints was easier but that CCP was more secure.

4.3.5 User choice

Users were told in the preamble to the session to pretend that their passwords were protecting bank information and as such they should choose points that were memorable to them but difficult for others to guess. Users apparently took these instructions seriously; for example, many commented on how they were avoiding certain areas because these would be too easy to guess or because they felt that others would select the same points.

Users developed strategies for selecting their points. Some tried to pick geometric patterns that applied across images such as selecting items along the bottom of the images, but most talked about picking things that have meaning to them such as their

initial from a sign or a familiar toy. One participant made up elaborate stories about each of the click-points. Users indicated that they preferred to click on things that were small and "clickable", such as letters or small circles. However, as we discovered later, users of CCP were much less likely than users of PassPoints to select their passwords in simple geometric shapes. These results are presented in Chapter 7.

As with PassPoints, participants felt strongly about the suitability of some images, with strongest reactions to images they disliked. They preferred images that were not too cluttered, that contained a variety of distinct items, that had small well-defined areas, and that featured contrasting colors. The most disliked images were uniform and repetitive, such as a circuit board or field of flowers, were highly cluttered, or had few items with well-defined borders.

4.4 Preliminary Security Analysis

We begin by clarifying our target scenario for CCP and the particular assumptions made about the system. We recommend that CCP be implemented and deployed in systems where offline attacks are not possible, and where any attack made against an online system will be allowed only a limited number of guesses made per account in a given time period (this limit should include restarts as well). This follows related comments by Davis et al. [27] regarding Faces and Story, even though we expect the security of CCP to be substantially stronger than those schemes. We further assume that all communication between the client and server will be made securely through SSL, maintaining secrecy of selected click-points and corresponding images, therefore avoiding simple attacks based on network sniffing.

We suggest that the image mappings (the mapping of tolerance squares to next-images based on f) be done on a per-user basis as a function of the username, as a form of salting to complicate the construction of general attack dictionaries. We also suggest that the image set across all users is a superset containing a very large number of images and that individual users are assigned a subset of these images for their image-maps.

General attacks against such a system, where attackers try to break into any account [93], are slowed by the precautions mentioned above. We assume that the

function f would be (or become) known to attackers. Hotspot analysis might be used to increase the efficiency of an attack dictionary but images would need to be collected, and such a dictionary would need to be re-generated on a per-user basis due to salting. For the best pattern-based attack against PassPoints in Salehi-Abari et al. [101], the pattern-based dictionary is image-independent, eliminating the need for image analysis. However, as shown in Chapter 7, such patterns are not found in CCP, so this image-independent strategy would not lead to efficient attacks on CCP.

Online attacks against specific users are more worrisome and require further examination. Even for online systems where the account is locked after t failed login attempts, non-trivial security is still necessary to guard against system-wide attacks over W accounts since an attacker gets $t \times W$ guesses per time window [93].

4.4.1 Shoulder-surfing and other information capture from users

Most graphical passwords are vulnerable to shoulder-surfing attacks [117]. With today's small cameras, camera phones, or even cameras with telephoto lenses [7, 70], it is easy to video-capture a user's screen or keyboard as they are logging in. CCP is also susceptible to such attacks and indeed in its present form the change in images may be easier to see from further away than mouse pointer movements in PassPoints. With knowledge of which images to look for in systems allowing sufficient numbers of online guesses, attackers could try a brute-force attack of clicking on points until the correct next image appears and use this in a divide-and-conquer password recovery.

If the username, the image sequence, and the click-points are observed through shoulder-surfing then an attacker has all of the information needed to break in to the account, as is the case with PassPoints and most other password systems. Having a compromised computer is also a threat because malware may capture the login information and relay that information elsewhere. Whereas a keylogger suffices for text passwords, for graphical passwords somewhat more sophisticated malware is needed to capture both the images and the cursor positions.

When only some of the information is known, it can be used to narrow the search for a correct guess. With PassPoints, knowing the username is enough to retrieve the user's sole image from the live system. With CCP, the username allows for retrieval

of only the first image, which provides only limited information to an attacker.

Knowing some images and their position in a user's sequence allows pruning of an attack dictionary. The attacker's job is made easier as more images from the user's password are known. Thus CCP is not suitable in environments where shoulder-surfing is a realistic threat, or environments where user images can otherwise be recorded (e.g., by insiders, malicious software on the client machine, etc.).

4.4.2 Hotspots and dictionary attacks

In cases where attackers are not in a position to capture information from the user, they are limited to what they can deduce through image analysis or through other predictable behaviour on the part of users. If attackers can accurately predict the hotspots in an image, then a dictionary of passwords containing combinations of these hotspots can be built. Hotspots are known to be problematic for PassPoints [35, 50, 119, 126]. Users may also select passwords with other common characteristics, such as selecting click-points that follow geometric patterns. However, we expect that pattern-based attacks [101, 126] are likely not a concern for CCP since our analysis of patterns (see Chapter 7) revealed that geometric patterns did not occur when the password was constructed across 5 images (as opposed to one image for PassPoints).

Our example system uses images of size 451×331 pixels, with tolerance squares of 19×19 pixels, which gives 432 tolerance squares per grid for a given image. Because the grid identifier for each click-point are stored during password creation (as discussed in Section 4.1), the correct grid is always retrieved by the system during login, so the fact that there are several grids does not come into play in online attacks. This means that for each image, there is a $1/432$ chance of clicking within the correct tolerance square. However, due to hotspots some of these have a much higher probability of being correct than others. Knowing the hotspots would allow an attacker to modify an attack dictionary to test passwords with higher probability first. For example, re-examining the data from our PassPoints lab study we found that, as a general result across 17 images used, the 30 largest hotspots on an image cover approximately 50% of user-chosen click-points. Assuming that attackers are first able to extract the necessary images and perform hotspot analysis, there is approximately

a 3% ($.5^5$) chance that a user-chosen password is contained in a dictionary of 2^{25} entries built entirely from hotspots. As discussed in Chapter 7, CCP images have approximately the same number of hotspots as PassPoints.

A key advantage of CCP over PassPoints is that attackers need to analyze hotspots on a large set of images rather than only one image since they do not know the sequence of images used for a given password. Secondly, using different subsets of images for different users means that an attacker must somehow gather information about the specific subset assigned to the current user.

When presented with the same images, users selected similar points in both our CCP and PassPoints user studies. However, for CCP only one click-point is selected per image as opposed to 5 click-points for PassPoints. Further testing is required to gather a larger sample of click-points per image for CCP, but preliminary analysis provides evidence that users are no more likely to select a popular hotspot as their click-point in CCP than with PassPoints.

A powerful attack method against graphical passwords involves looking at dependencies between two adjacent components in passwords [27, 126]. For example, in PassPoints this involves how a click-point may depend on the previous click-point in a user-chosen password. Van Oorschot and Thorpe [126] show how this can be exploited for efficient attacks against PassPoints. This type of attack seems unlikely to be effective for CCP since having only one click per image appears to destroy obvious relationships between click-points (see Chapter 7 for evidence of the lack of patterns between click-points).

4.5 Discussion

From a usability point of view, CCP appears quite successful. Success rates were high, with 96% of logins being successful on the first attempt (see Table 4.1). These success rates and the median password entry times (see Table 4.3) are similar to results for text passwords under similar conditions [45]. Users also felt that it got progressively easier to use CCP passwords as they progressed through the session.

Based on comments and feedback provided by users during the sessions, we believe that users appreciated the implicit feedback. As soon as they saw an unfamiliar image,

they knew they were on the wrong path and restarted. They liked being able to narrow down exactly which click was erroneous, a feature that is lacking in PassPoints. They also told us that seeing each image triggered the memory of where they had clicked.

Participants were accurate in their targeting of click-points. During the Login phase, 92% of click-points fell within a 9×9 pixel square of the original click-points. The accuracy of CCP click-points provide further evidence that tolerance squares as small as of 9×9 pixels may be acceptable in terms of usability.

We can also compare results of CCP with our PassPoints studies. When comparing only the lab studies, participants performed similarly well in terms of login accuracy and success rates. The median login click-time for our PassPoints system was 7.0 seconds while for CCP it was 6.0 seconds, despite CCP's time including the time to re-orient as each image appeared (as opposed to PassPoints where the majority of thinking occurred before the first click, when users first saw the image, and as such is not included in these click-time results). Of those participants who tried both systems, a preference for CCP was evident. In this limited sample, the most common reasons for preferring CCP were because seeing each image triggered their memory of their click-point, there was no need to remember the order of the click-points, and they received implicit feedback about the correctness of their latest click. This comparison is somewhat biased since users had much more practice with one system than the other, but these responses do correspond to what would intuitively be expected.

With any password-based authentication scheme, a common goal is to maximize the theoretical password space in order to make it more resistant to attack. A few alternatives are presented below to increase the theoretical password space for CCP. Of course, the usability of the system must also be considered when such changes are made to a system. A study examining these issues is discussed in Section 9.3. These strategies for increasing the theoretical password space are examined further in Section 8.1.

Adding more click-points (variable password length)

As with PassPoints, one way to increase the password space is to increase the number of click-points contained in a password. This comes at the cost of increasing the memory burden on users. Although this would need to be empirically tested, it seems that the negative impact may be less with CCP than with PassPoints since a one-to-one mapping between images and click-points in CCP may be easier for users to manage. Therefore moving to 6 click-points may be a reasonable strategy for CCP.

Alternatively it is possible to enforce a minimum number of clicks (images) but allow users to decide for themselves how many clicks their password contains, similar to minimum password lengths for text-based passwords. In this case, the system would continue to show the next image in the sequence but the user would determine at which point to stop clicking and press the login button. Granted, most users would probably pick the minimum length, but a user concerned about security could build a longer password. If k bits of security are assumed per image used, then for a password using c click-points, the security would be $c \times k$.

Adjusting the image and tolerance sizes

A simple way of enlarging the theoretical password space is to use larger images or reduce the tolerance. Both have the effect of adding squares to the grids. Tolerance cannot be reduced past a certain threshold because it becomes impossible for users to accurately re-enter their passwords. Results of this CCP study and our earlier PassPoints studies, however, indicate that it may be possible to reduce tolerance more than was originally believed [137] (at least on full-sized monitors) since users were very accurate in targeting their click-points. For example, with images of size 451×331, as used in these studies, there are 432 19×19 pixel grid squares, giving $432^5 \approx 2^{44}$ 5 click-point passwords in the theoretical password space. If we reduced the tolerance squares to 9×9 pixels, this would increase to 1887 squares per image and increase the size of the theoretical password space to $1887^5 \approx 2^{54}$. The second way of increasing the theoretical password space is to enlarge the image. Enlargement is restricted by the size of the screen used. Increasing the size of the image may also make it more susceptible to shoulder-surfing. Zooming, which has been suggested

elsewhere, including by Wiedenbeck et al. [136] for PassPoints, often has usability problems of its own, and thus we hesitate to propose it here.

Using a larger set of images

At minimum, the size of CCP's total image set should match the number of squares in a tolerance grid (i.e., 432 in our example system). This strategy would imply that the set of images in the system is re-used across users and at each stage in the password for each user.

In this case, if users make a mistake during login, there is a small chance that they accidentally see an image belonging somewhere else in their password sequence. They may realize the mistake immediately or subsequently when an unknown next-image appears. The possibility of such collisions can be reduced or eliminated if the number of images is increased to reduce (or at the extreme, entirely eliminate) the overlap between password stages. However, depending on implementation details, this could imply that the entire sequence could be deduced from knowing only the last image in a password, as discussed below.

As suggested earlier, it is possible to have a larger set of images in the system and to use a subset for each user. Additionally, the subset for each user may include enough images so that not every image is re-used at each stage. For example, if only 25% of images are re-used per stage, then 1405 images would be required per user for our example system of 5 click-points and 432 grid squares per image (for further discussion, see Section 8.2.1). This can reduce the possibility of collisions during incorrect login. It also increases the work required by attackers to identify images and determine hotspots as this work increases proportionally with the number of images used in the system. In comparison, with PassPoints only one image needs to be analyzed per user and this image is accessible by knowing the username. If attackers are using an offline brute-force attack where all possible combinations of images and click-points are used, then there are $(totalImages \times totalGridSquares)^{totalClicks}$ potential passwords with CCP since the image identifier for each click-point is included in the hashed password. For example, with 1405 images, 432 grid squares, and 5 click-points, there are $(1405 \times 432)^5 \approx 2^{96}$ candidate passwords.

If attackers know the image-mapping function f and the set of images used, then using more images has no effect on the password space beyond requiring more processing time to determine hotspots if a dictionary attack based on hotspots is used, since the correct next-image can be determined for each grid square. However, even if attackers know f, collecting the set of images still poses a challenge because they must either have insider access to the system or they must discover the images one at a time by selecting different click-points during login attempts on the particular account. This can prove time-consuming since the number of unsuccessful login attempts allowed on a particular account can be restricted (e.g., see [125]). When both f and the image set are known, the password space is determined by the number of paths through the image-map tree (generated by f), based on the number of squares in the tolerance grid, not the number of images available. If a dictionary was built containing all paths through the tree, the number of entries would be the same (2^{44} for grids containing 432 squares and 5 clicks) regardless of the number of images used (although the entries would be different).

In cases where attackers know f and the set of images used, as well as one or more images in the password (gathered through shoulder-surfing or malware installed on the client machine), then having a very large set of images for a given user can leak some information about the password to the attacker (although the amount of work required for hotspot analysis is still increased). This is because if not all images will be used within each image-map, then attackers can use this information to eliminate branches of the image-map tree that do not contain the known image at the correct stage. At the extreme case where there are no duplicate images (i.e., 432×5 different images are used for a given user), then knowing the last image of a sequence would identify a unique path through the tree and reveal the password. Conversely, when all images are re-used at each stage, then no branches can be eliminated and knowing the last image will not result in a unique path. See Section 8.2 for further discussion.

Another alternative for increasing the number of images available is to use larger images but crop them differently for each user. This would complicate hotspot analysis for attackers because the coordinates of hotspots determined for one account could not be applied directly to other accounts.

4.6 Conclusion

The proposed Cued Click-Points scheme shows promise as a usable and memorable authentication mechanism. By taking advantage of users' ability to recognize images and the memory trigger associated with seeing a new image, CCP has advantages over PassPoints in terms of usability. In one-to-one cueing, each image acts as a cue for the corresponding click-point. Having to remember only one click-point per image appears easier than having to remember an ordered series of clicks on one image. The inclusion of implicit feedback, which signals to users whether their previous click-point was entered correctly, also appears helpful to users. In our small comparison group, most users strongly preferred CCP.

We also believe that CCP offers a more secure alternative to PassPoints. CCP increases the workload for attackers by forcing them to first acquire image sets for each user, and then conduct hotspot analysis on each of these images. The system's flexibility to increase the overall number of images in the system allows us to increase this workload. Also, certain pattern-based attacks possible on PassPoints, and attacks exploiting dependencies between click-points, do not appear applicable to CCP.

Chapter 5

Persuasive Cued Click-Points

We have evidence that users in both PassPoints and CCP tend to select click-points from common areas of the image, forming hotspots. Visual attention research [139] shows that different people are attracted to the same predictable areas when looking at an image, which may partially explain why hotspots occur. This suggests that if users select their own click-based graphical passwords without guidance, hotspots will remain an issue. Davis et al. [27] suggest that user choice in all types of graphical passwords is unadvisable because users will always select predictable passwords. To the best of our knowledge, no research exists on helping users select better graphical passwords, nor on how to avoid hotspots in click-based systems during password creation. In this chapter, we present Persuasive Cued Click-Points (PCCP), a system that influences users to select better passwords. The work presented in this chapter was published at the 2008 British HCI conference [16].

5.1 Persuasive Technology

Persuasive Technology was first articulated by Fogg [42] as using technology to motivate and influence people to behave in a desired manner. He discusses how interface cues can be designed to actively encourage users to perform certain tasks. We propose how these may be condensed into a set of core persuasive principles for computer security, in a paper co-authored by Forget, Chiasson, Biddle, and van Oorschot [44].

An authentication system which applies Persuasive Technology should guide and encourage users to select stronger passwords, but not impose system-generated passwords. To be effective, the users must not ignore the persuasive elements and the resulting passwords must be memorable. As detailed in the next section, our proposed system accomplishes this by making the task of selecting a weak password more tedious and time-consuming. The *safe-path-of-least resistance* for users is to select

a stronger password (not comprised entirely of known hotspots or following a predictable pattern). As a result, the system also has the advantage of minimizing the formation of hotspots across users since click-points are more randomly distributed.

5.2 Persuasive Cued Click-Points (PCCP)

We investigated whether password choice could be influenced by persuading users to select more random click-points while still maintaining usability. Our goal was to encourage compliance by making the less secure task (i.e., choosing poor or weak passwords) more time-consuming and awkward. In effect, behaving securely became the safe-path-of-least-resistance.

Using Cued Click-Points (CCP) as a base system, we added a persuasive feature to encourage users to select more secure passwords, and to make it more difficult to select passwords where all five click-points fall within hotspots. Specifically, when users created a password, the images were slightly shaded except for a randomly positioned viewport (see Figure 5.1). The viewport is positioned randomly, rather than specifically to avoid known hotspots, because this could also lead to the formation of new hotspots and such information could be used by attackers to improve guesses. The viewport's size was intended to offer a variety of distinct points but still cover only an acceptably small fraction of all possible points. Users were required to select a click-point within this highlighted viewport and could not directly click outside of this viewport. If users were unwilling or unable to select a click-point in this region, they could press the "shuffle" button to randomly reposition the viewport. While users were allowed to shuffle as often as they wanted, this significantly slowed the password creation process. The viewport and shuffle buttons only appeared during password creation. During the Confirm and Login phases, the images were displayed normally as in CCP, without shading or the viewport, and users were allowed to click anywhere.

Our hypotheses were:

1. PCCP users will be less likely than users of PassPoints or CCP to select click-points that fall into known hotspots.

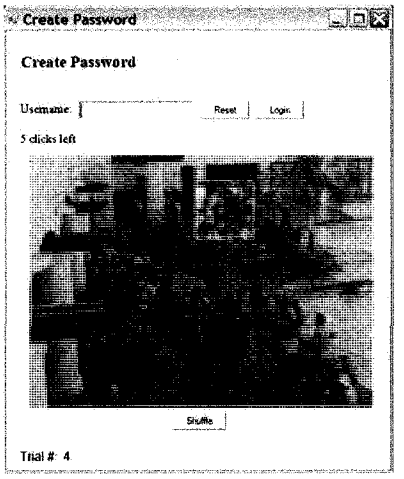


Figure 5.1: Screenshot of the PCCP Create Password interface with the viewport highlighting a portion of the image. (Pool image from [90])

2. The PCCP click-point distribution across users will be more randomly dispersed than those from PassPoints and CCP and will not form new hotspots.

3. The Login success rates for PCCP will be similar to those of the original CCP system.

5.3 PCCP Lab Study

We tested Persuasive-CCP (PCCP) in a lab study with 39 participants. The usability study followed the same methodology as our previous lab studies (see Section 3.1.1).

Participants ranged in age from 17 to 37. Most were university students from various fields. All were regular computer users who were comfortable with passwords and using a mouse. In total, data from 307 trials was collected.

The PCCP system was identical to that of the CCP study, except for the addition of the viewport in the Creation phase. In our test system, the viewport was a 75×75 pixel square. System logs also recorded the location of the viewport for each shuffle.

We used a between-participants design, with all participants from this study assigned to the viewport condition. For comparison, we used data collected from our previous PassPoints and CCP studies where participants created passwords without the viewport. The methodology, including instructions to participants (other than

explaining the viewport), questionnaires, equipment, software (other than the addition of the viewport), and images were identical to those used for CCP. Although recruited at different times, participants were all university students studying in various fields and were all recruited using the same methods. Data collected from CCP can therefore be used as a control group against which to measure the effects of the viewport in PCCP. We recognize that ideally, data from a new control group would have been collected at the same time as the PCCP dataset to further ensure that no external factors affected the results.

5.4 Collected Results

To analyze PCCP's performance, we compared the data from this user study to the following three datasets collected in our previous studies (Chapters 3 and 4: PassPoints Lab (PPLab), PassPoints Field (PPField), and Cued Click-Points (CCP).

The system used in our initial CCP study randomly selected which of the 330 images to display and this led to a small number of click-points per image. To more accurately compare the effects of the viewport, we needed more CCP click-point data. We modified the CCP image selection algorithm to ensure that the 17 images used in the PassPoints lab study(Figure 3.1) were randomly displayed within the first 6 trials completed by each participant. We collected data from an additional 33 CCP participants to ensure that we had enough CCP click-point data for comparison with PCCP in our hotspot analysis. This weighted image selection algorithm was also used for all PCCP participants. Methodologically, collecting all data using this improved selection algorithm would have been better, but time constraints prevented us from repeating the entire CCP study.

We had the most data available for the two images used in the field study: the Pool image (Figure 3.7) and the Cars image (Figure 3.6). In most cases, the click-points collected in the PPField study will be used as the reference dataset since they were gathered in a realistic usage scenario and included the most samples.

Our data analysis examines several aspects of the system in order to address each of our previously stated hypotheses. We first look at the general usability of PCCP, then focus on the issue of hotspots.

Table 5.1: PCCP success rates on the first attempt out of 307 trials. Only trials that were correct on the first attempt, with no restarts, are considered successful.

	Create	Confirm	Login
PCCP Success rate	305/307 (99%)	211/307 (69%)	278/307 (91%)

5.4.1 Success rates

As shown in Table 5.1, participants were able to successfully use PCCP. Success rates were calculated as the number of trials completed without errors or restarts, over all trials (i.e., successful on the first attempt). Participants had some difficulty during confirmation while learning their password, but had little problem logging on afterwards. The success rates in Table 5.1 were calculated using the most stringent criteria: only passwords that were entered correctly on the first attempt without pressing the reset/clear button were considered successful. With a broader interpretation of "success", there are only 3 instances (1% failure) where users were unable to eventually log in correctly and had to create a new password.

In comparison, CCP's Confirm and Login success rates were 83% and 96% respectively (Chapter 4). We suspect that PCCP participants had more difficulty initially learning their password because they were selecting click-points that were less obvious than those chosen by CCP (and PassPoints) participants. However, PCCP participants were ultimately able to remember their passwords with a little additional effort. The Login success rates of CCP and PCCP are not significantly different ($\chi^2(1, N = 564) = 0.07, p = .796$) , thus suggesting that the gain in security (reduction in the number of hotspots, as shown in Section 5.4.4) was not at the expense of usability, at least not in the lab environment.

5.4.2 Times for password entry

Password creation was the longest of the three phases (Table 5.2). Users got progressively quicker with each phase. This is consistent with the pattern seen in our previous graphical password studies. We report the total time taken to complete a phase: from the time the first image was displayed to the time that they pressed the

Table 5.2: PCCP lab study completion times for each phase (in seconds)

	Create	Confirm	Login
Total time: mean	50.7	29.9	16.2
Total time: median	41.4	18.9	14.0
Click-time: mean	36.3	24.9	10.6
Click-time: median	28.5	11.6	7.8

Table 5.3: PCCP effect of shuffling on success rates for 307 trials

Shuffles	# of trials	Login Success Rate
Low (0-5)	194 (63%)	89%
High (>5)	113 (37%)	94%

Login button, which included time spent thinking about their password. We also report the "click-time": the time taken from the first click-point to the fifth click-point. This represents the time taken to actually enter their password.

PCCP participants had a median click-time of 7.8 seconds for the Login phase, which is slower than CCP's 6.0 seconds (Chapter 4). This difference is likely due to the slightly steeper learning curve from memorizing a password that is not comprised of hotspots.

5.4.3 Shuffles

The shuffle button was used moderately during password creation (Table 5.3). During the Create phase, 63% of trials had 5 or fewer shuffles across all 5 images within a password (i.e., an average of at most 1 shuffle per image). We found that users who shuffled a lot had higher Login success rates than those who shuffled little, but the difference was not statistically significant ($t(305) = 1.89, p = .06$). Using linear regression, we further found that shuffling did not correspond to selecting click-points falling into known hotspots for the Pool and Cars images, the two images for which we had hotspot information from the PassPoints field study ($F(1.65) = 0.2068, p = 0.7$).

Most participants devised a shuffling strategy and used it throughout their session. They either consistently shuffled a lot at each trial or barely shuffled during the entire session. Those who barely shuffled selected their click-point by focusing on the section of the image displayed in the viewport, while those who shuffled a lot scanned

the entire image, selected their click-point, and then proceeded to shuffle until the viewport reached that area. When questioned, participants who barely shuffled said they felt that the viewport made it easier to select a secure click-point. Those who shuffled a lot felt that the viewport hindered their ability to select the most obvious click-point on an image and that they had to shuffle repeatedly in order to reach this desired point.

5.4.4 Hotspots

The primary goal of PCCP was to increase the effective password space by guiding users to select more random passwords. To gauge our success, we therefore needed to determine whether PCCP click-points were more randomly distributed across the image and whether they successfully avoided known hotspots from previous studies.

To begin our analysis, we represented the click-point data graphically on the images themselves. The PPField study yielded a large volume of data about where users clicked on the Pool and Cars images. We used a Gaussian kernel smoothed intensity function [34] to summarise this data for each image. We then created heat maps to depict this summary on the image area, using several colour bands to represent varying intensities of click-point concentration. The most intense areas thus correspond to hotspots. This heat map of hotspots was used as the basis for comparing whether PCCP was better at avoiding known hotspots than CCP.

Figure 5.2 shows the heat map for the PPField click-points on the Pool image. White areas are the least click-point intensive and cover most of the image area. The five colour bands from red to yellow indicate progressively more intense areas thus revealing severe hotspots. The figure shows the same heat map twice: on the left, overlaid with the individual click-points (shown as small circles) from the CCP study (34 click-points), and on the right for our PCCP study (35 click-points). Figure 5.3 shows the corresponding information for the Cars image. Visually, it appears that PCCP click-points are more randomly distributed across the image, and not as concentrated on the heat map hotspots. Since visual inspection alone does not provide an accurate measure, we further tested to see whether this was true by conducting a dictionary attack on the click-points and by conducting some spatial statistics tests

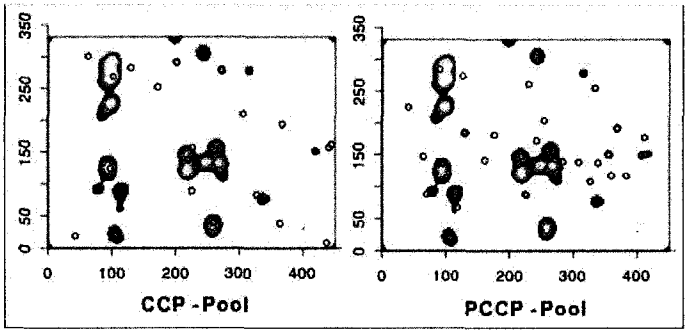


Figure 5.2: Displays individual click-points from CPP and PCCP respectively for the Pool image. The base heat map shows the location of known hotspots derived for the PPField dataset and thus is identical on both plots. The heat map is included to illustrate how many of the CCP and PCCP click-points fall near or within known hotspots. (Best viewed in colour).

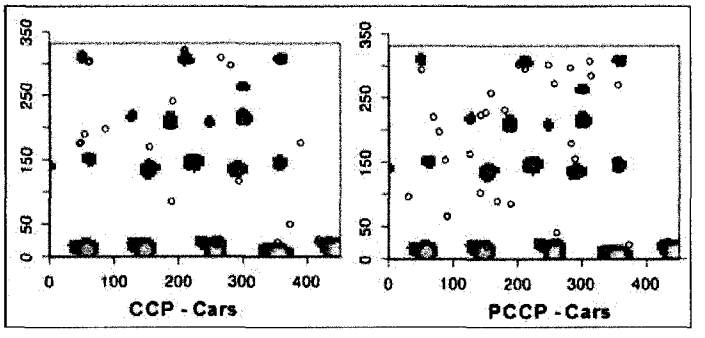


Figure 5.3: Displays individual click-points from CPP and PCCP respectively for the Cars image. The base heat map shows the location of known hotspots derived for the PPField dataset and thus is identical on both plots. The heat map is included to illustrate how many of the CCP and PCCP click-points fall near or within known hotspots. (Best viewed in colour).

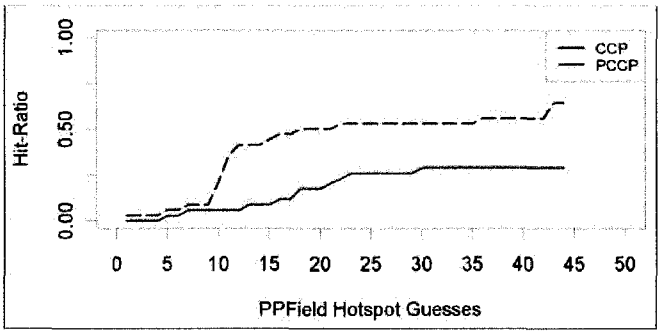


Figure 5.4: Individual click-points "guessable" using hotspots from the PPField study on the Pool image

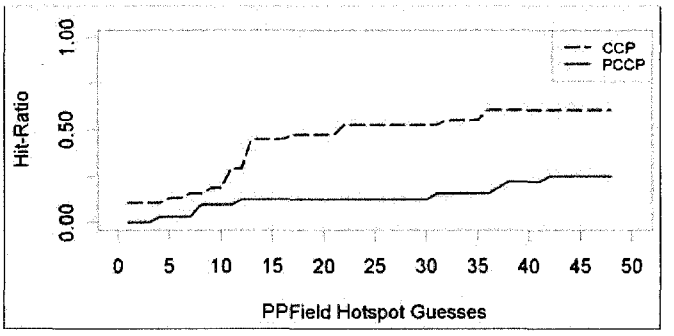


Figure 5.5: Individual click-points "guessable" using hotspots from the PPField study for the Cars image

which confirm that PCCP click-points are more randomly distributed on the images.

To determine whether PCCP helped users avoid hotspots, we used the data from the earlier PPField study to compile a list of hotspots for the Pool and Cars images. The PPField datasets included 580 click-points for Pool and 545 click-points for Cars. The hotspots were determined by finding the number of neighbouring click-points that were within tolerance of each click-point, sorting in decreasing order on this number of neighbours, then greedily assigning each click-point to the largest hotspot for which it was within tolerance. The result was a list of hotspot coordinates sorted in decreasing order by number of click-points they encompass.

We compared these hotspots to the click-points gathered for PCCP and CCP. Figure 5.4 and Figure 5.5 show the cumulative percentage of individual click-points that were "guessable" (i.e., the click-point fell within tolerance of a hotspot) for the Pool and Cars images respectively. PCCP click-points were much less likely to fall

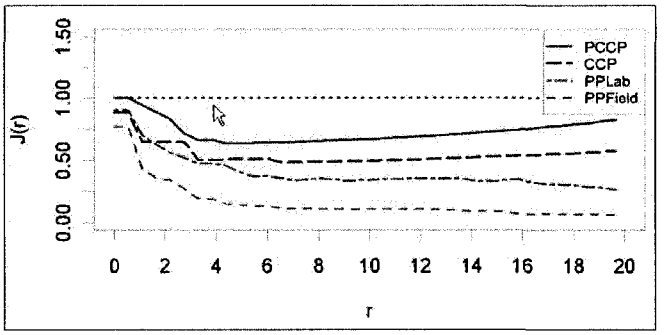


Figure 5.6: J-function showing amount of clustering at different radius values measured in pixels for PCCP, CCP, PPLab, and PPField on the Pool image. PCCP has the least clustering.

within hotspots than CCP's. For example, in the dataset for the Pool image, the 12 largest hotspots correctly identify 40% of CCP click-points but only 8% for PCCP. It should be noted that these are individual click-points, not passwords. An attacker would need to correctly identify all five of a user's click-points and images in order to successfully guess a password.

Due to the large set of images used in PCCP and CCP, we currently do not have hotspot information on all images and thus could not build an attack dictionary for entire passwords. However, we can use the same method as in the CCP study (see Section 4.4.2) as an estimate. For CPP (and PassPoints), the top 30 hotspots on an image cover approximately 50% of click-points (see Figure 5.4 and Figure 5.5). Assuming that a password consists of 5 click-points, the probability that a given password is found in an attack dictionary built from these hotspots would be $0.5^5 = 3\%$. For PCCP, the top 30 hotspots cover between 12% and 25% of click-points on the Pool and Cars images, so using an estimate of 20%, the probability that a password is in the same attack dictionary becomes $0.2^5 = 0.03\%$.

Standard statistical methods were inappropriate for this analysis because of the 2-dimensional nature of the click-point data. We instead applied point pattern analysis from spatial statistics [34] to measure the occurrence of hotspots and to evaluate whether click-points from the current PCCP study largely avoided hotspots established in the PPField study. We used the R programming language for statistical analysis and the spatstat package [8] to conduct our analysis.

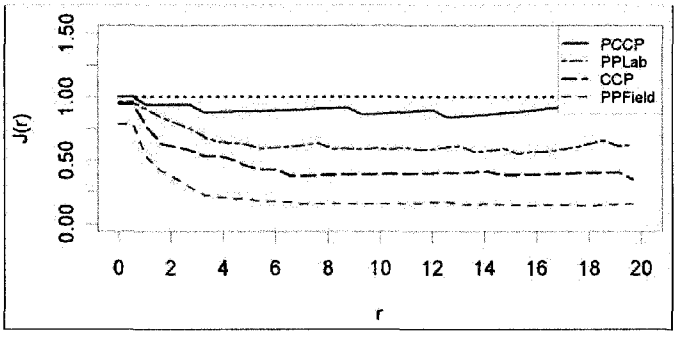


Figure 5.7: J-function showing amount of clustering at different radius values measured in pixels for PCCP, CCP, PPLab, and PPField on the Cars image. PCCP has the least clustering.

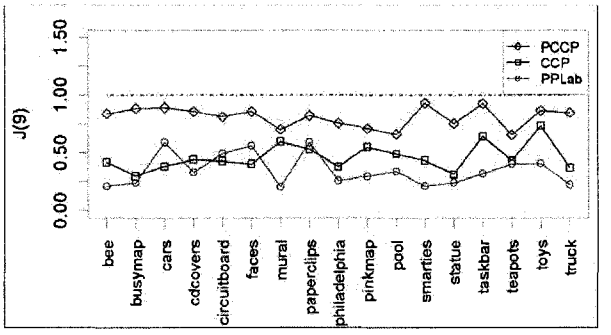


Figure 5.8: J-function at r=9 pixels for the set of 17 core images

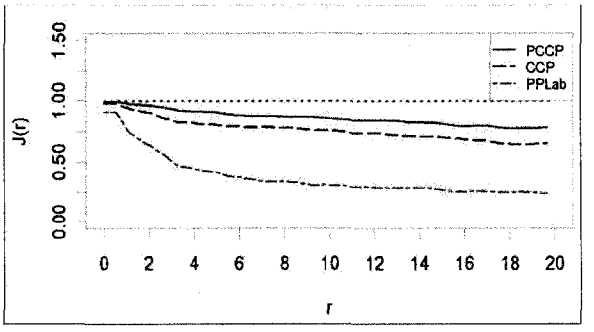


Figure 5.9: Cross J-function comparing PCCP, CCP, and PPLab to PPField reference dataset for the Pool image. PCCP is most dissimilar.

To measure the level of clustering of click-points within datasets (the formation of hotspots), we used the J-function [123] statistic from spatial analysis. The J-function combines nearest-neighbour calculations and empty-space measures for a given radius r in order to measure the clustering of points. A result of J closer to 0 indicates that all of the data points cluster at the exact same coordinates, $J = 1$ indicates that the dataset is randomly dispersed, and $J > 1$ shows that the dataset is uniformly distributed. Ideally, we want the results to be near 1, indicating that the click-points are nearly indistinguishable from randomly generated points. Figures 5.6 and 5.7 show that click-points on the Pool and Cars images are more randomly dispersed for PCCP than the other three datasets, indicating that the persuasive viewport was successful at guiding users to select more random click-points.

We further looked at the J-function measures at $r = 9$ pixels for the set of 17 core images used in all of our lab studies (see Figure 3.1). A radius of 9 approximates the size of the tolerance squares (19×19 pixels) used to determine whether a click was correct during password re-entry. Figure 5.8 shows that PCCP approaches complete spatial randomness for all 17 images (near $J = 1$). A line graph was used for clarity, but in reality these are discontinuous points.

The Cross J function [124] is a multivariate summary statistic measuring the interaction between two spatial datasets. We use it as a measure of whether the PCCP click-points differ from those collected in previous click-based graphical password studies. Cross J close to 0 indicates that the two datasets are taken from the same population, Cross $J = 1$ shows that the datasets are distinct, and Cross $J > 1$ means that the datasets "repulse" each other. Figure 5.9 shows the Cross J values comparing each of the lab studies to PPField for the Pool image. The values for PCCP are approaching 1, indicating that the PCCP dataset is distinct from the PPField reference set. Similar results were found for the Cars image. As results for PCCP are closest to 1, the Cross J function supports the assertion that the PCCP dataset is most dissimilar (among the three lab datasets) to our reference dataset of PPField.

5.4.5 Validation of hypotheses

We now revisit our hypotheses to evaluate whether to accept or reject them in light of the data analysis.

1. PCCP users will be less likely than users of PassPoints or CCP to select click-points that fall into known hotspots. *Hypothesis supported:* This was confirmed by using known hotspots from the PPField data to attack the PCCP and CCP datasets. Click-points were significantly less predictable for PCCP (recall Figure 5.4 for Pool and Figure 5.5 for Cars), indicating that they did not fall within known hotspots. The Cross J-function results also provide statistical evidence that the PCCP dataset is more distinct from the PPField dataset than PPLab or CCP.

2. The PCCP click-point distribution across users will be more randomly dispersed than those from PassPoints and CCP, and will not form new hotspots. *Hypothesis supported:* The results of the J-function tests show that the PCCP dataset is more random (less clustered) than the previous PPLab, PPField and CCP datasets.

3. The Login success rates for PCCP will be similar to those of the original CCP system. *Hypothesis supported:* The difference in Login success rates between PCCP and CCP are not statistically significant, despite apparently more secure passwords in PCCP.

5.5 Discussion

A common goal in authentication systems is to maximize the size of the effective password space. When user choice is involved, this also becomes a usability issue since users will be responsible for selecting their password. We have shown that it is possible to allow user choice while still increasing the effective password space.

A few users shuffled a lot (the user who shuffled the most did so 201 times on one image), until they reached a desired area of the image. These passwords may be more vulnerable to attack. This would be especially problematic in a multiple

account attack scenario where attackers target large numbers of accounts in hope of guessing any password. We could further deter users from selecting obvious click-points by limiting the number of shuffles allowed during the creation of a password or by progressively slowing system response in repositioning the viewport with every shuffle past a certain threshold. These approaches present a middle-ground between insecure but memorable user-chosen passwords and secure system-generated random passwords that are difficult for users to remember. While user choice is influenced with PCCP, the low number of shuffles for the majority of users indicates that users were willing to accept the system's suggestion. We believe that this design decision is justified by the increased security it offered and the apparently minimal usability drawbacks.

Furthermore, tools such as PCCP's viewport are only used during password creation so they cannot be exploited during an attack on an existing account. PCCP also does not need any modification to the verification component of the system. Although outside the scope of this thesis, we have been investigating ways of applying Persuasive Technology to text passwords [45](see Section 9.3). Both of these features are especially advantageous for text passwords because they require minimal modification to existing authentication systems and thus would be easier to adopt.

Providing instructions on how to create secure passwords, using password managers, or providing tools such as strength-meters to gauge the strength of a password have had only limited success [41]. The problem with such tools is that they require additional effort on the part of users who are creating passwords and often provide little useful feedback to guide the user's actions. In PCCP, creating a more secure password (by selecting a click-point within the first system-suggested viewport position) is the easiest course of action and requires little additional cognitive effort. Users still make a choice but they are influenced in their selection. Reducing complexity within a task and providing guidance through tunneling [42] are both recommended strategies in Persuasive Technology for encouraging users to behave in the desired manner. PCCP demonstrates one possible application of Persuasive Technology but other strategies could also be applied, even for graphical passwords.

Another often cited goal of usable security is helping users form accurate mental models of security. Through questionnaires and conversations with participants in authentication usability studies, it is apparent to us that in general, users have little understanding of what makes a good password and how to best protect themselves online. Furthermore, even those who are more knowledgeable usually admit to behaving insecurely (such as re-using passwords, or providing personal information online even though they are unsure about the security of a website) because it is more convenient, because it is the only way they can cope with the memory load of too many passwords, and because they do not fully understand the possible consequences of their actions.

We believe that guiding users in making more secure choices, such as using the viewport during graphical password selection, can help foster more accurate mental models of security [20, 39, 134, 144]. Rather than providing vague instructions such as "pick a password no one will guess", we are actively showing users how to select a more random password as they perform the task. CCP and PCCP additionally offer one-to-one cued recall, which may help ease the memory burden, and implicit feedback that helps users recognize when they have made a mistake during password entry.

Although these initial results are promising, further work is needed to test the long-term memorability of PCCP passwords, test the effect of interference when users must remember multiple passwords, and observe user behaviour in a real-world setting. A field study where participants use PCCP passwords, instead of text passwords, to access online resources over a few months would provide insight into these issues.

5.6 Conclusion

An important usability and security goal in authentication systems is to help users select better passwords and thus increase the effective password space. Our earlier PassPoints studies revealed memorability issues and security concerns because users selected click-points that formed hotspots, making it possible to conduct successful dictionary attacks with minimal effort. CCP was designed to address these issues by using one-to-one cueing, adding implicit feedback, and increasing the number of

images used to proportionally increase the effort required to perform hotspot analysis. However, hotspots were still occurring in CCP.

We designed PCCP to encourage and guide users in selecting more random click-based graphical passwords. A key feature in PCCP is that creating a secure password is the "safe-path-of-least-resistance", making it likely to be more effective than schemes where behaving securely adds an extra burden on users. The approach has proven effective at reducing the formation of hotspots and avoiding known hotspots, thus increasing the effective password space, without significantly affecting the memorability of passwords.

Chapter 6

Centered Discretization

When testing authentication mechanisms with prototypes, it is reasonable for an implementation to differ from that of a deployable system. The prototype is instrumented to record user behaviour and other modifications, such as storing passwords unencrypted, may be necessary to evaluate the usability and security of the system more easily. However, it is also important to consider the impact of the proposed deployable implementation because it may introduce new usability or security problems.

In this chapter, we show that the implementation of PassPoints proposed by the original PassPoints authors [9] would have a significant negative impact on both the security and usability of the system. These problems were not apparent in the Wiedenbeck's et al. [135–137] studies or our studies reported in Chapter 3 because these prototypes used a simplified implementation [12]. This work on centered discretization was published at UPSEC 2008 [19].

6.1 Discretization

In our user testing of PassPoints, CCP, and PCCP on prototype systems, we stored click-point data in the clear, making it easy to compute whether a login click-point was within the acceptable tolerance square. For example, with a 19×19 tolerance square centered around a click-point, any login entry within 9 pixels in the x- or y- direction of the original coordinates should be accepted as correct. These systems must allow for some level of inaccuracy when re-entering passwords because it is unrealistic to expect users to always identify and target the exact same pixel. However, for a real implementation, graphical password coordinates should not be stored "in the clear" but rather they are ideally cryptographically hashed to provide an additional layer of security in case the password file is compromised, similar as with regular text

passwords. For click-based graphical passwords, this means that an approximately correct entry must result in the same hash value as the original password so that the system can recognize it as correct. A simple solution is to overlay a static grid (potentially invisible to users) onto the image and associate each pixel with the grid-square that contains it. The hashed password consists of the identifiers of the grid-squares rather than the original pixels. During re-entry, if a click-point falls within the same grid-square as the original point, then the entry is accepted since its hashed value matches the original. However, using a static grid leads to the "edge problem": if an original click-point is very close to a grid line, then during re-entry a click-point may be within tolerance but fall in an adjacent grid-square, and thus be rejected by the system because the hash values of the two points do not match. Therefore, more sophisticated discretization methods are required.

Robust discretization was proposed by Birget et al. [9] in conjunction with PassPoints as a means of performing this discretization of click-points. As shown in this chapter, robust discretization results in "false accepts" and "false rejects" when re-entering passwords because the tolerance region is not guaranteed to be centered on the original click-point. Through post-hoc analysis of our long term field study of PassPoints, we provide empirical evidence that this likely causes significant problems in practice.

We propose *centered discretization*, an alternative scheme that eliminates false accepts and false rejects as defined herein, providing system behaviour consistent with users' likely mental model of the system. It also allows for a larger theoretical password space because the tolerance squares can be smaller while still providing the same guaranteed minimum tolerance as robust discretization. We compare the usability and security of centered discretization and robust discretization using data collected from our field study of PassPoints.

6.2 Robust Discretization

To address the edge problem discussed in Section 6.1, Birget et al. [9] proposed *robust discretization*. This approach involves using three offset grids to guarantee that every point in the image is a "safe" distance away from the edges of at least one grid. It

was shown that three grids were necessary and sufficient to guarantee that for any given point in a 2-dimensional space, the system: (1) "guarantees the acceptance of approximately correct passwords", i.e., if a login click-point is within distance r from the original click-point then the input is accepted; and, (2) "guarantees the rejection of significantly wrong passwords": if a login click-point is at a distance greater than r_{max} (see Section 6.2.1) from the original click-point for some specified tolerance, the input is guaranteed to be interpreted as different from the original click-point.

Parameter r represents the minimum tolerance level desired. To achieve the stated objectives, the three grids are diagonally offset from each other by a distance of $2r$ and each grid-square is of size $6r \times 6r$. When an original click-point is selected, one of the three grids is chosen such that the click-point falls at least distance r from the grid's edges. They say that the user-entered click-point is *r-safe* in this particular grid.

For each point, the system stores the grid identifier in the clear, and determines which grid-square contains the click-point. The coordinates of this grid-square are cryptographically hashed and the hash is stored along with the grid identifier. For each click-point in future login attempts, the system overlays the pre-selected grid onto the image and finds the coordinates of the grid-square containing the click-point. The resulting password is hashed to see if it matches the stored hash value.

6.2.1 Definition of false accepts and false rejects

While robust discretization guarantees at least an r-safe tolerance around each point, it does not guarantee that this tolerance is *exactly* r-safe. For example, with grid-squares of size $6r \times 6r$, a reasonable interpretation by users might assume that a uniform $3r$ tolerance buffer exists around the original click-point. We define a uniformly distributed buffer as the *centered-tolerance*. However, in robust discretization, an original click-point is only guaranteed to be at least distance r from edges of the $6r \times 6r$ grid-square. So in the worst case, a click-point is of distance r from one edge, but is consequently a distance of $5r = r_{max}$ from the opposite edge. Figure 6.1 shows this discrepancy between centered-tolerance and a robust discretization grid-square in the worst case. This means that users clicking $r + 1$ pixels away in one

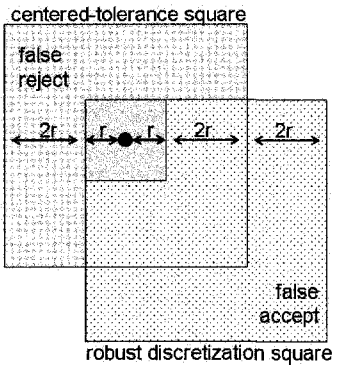


Figure 6.1: The small circle is the original click-point. The centered-tolerance square is the uniformly distributed tolerance likely expected by a user. The dotted square is the grid-square used by robust discretization in the worst-case. The non-overlapping region of the centered-tolerance square is the area where false rejects would occur in robust discretization, while the non-overlapping region of the robust discretization square indicates false accepts in robust discretization.

direction could have their login attempt rejected, but could click as far as $5r$ pixels in the opposite direction and be successful, which may confuse users. Furthermore, to have a usable implementation, r needs to be sufficiently large to allow a reasonable minimum tolerance around an original click-point. This means that the grid-squares will be correspondingly large (at $6r \times 6r$), reducing the theoretical password space for attackers.

In light of these circumstances, we introduce the terms *false rejects* and *false accepts* in the context of PassPoints implemented using robust discretization (see Figure 6.1). False rejects occur when a user clicks within the centered-tolerance area of a point but the click is rejected because it falls outside of the robust discretization grid-square (as little as $r + 1$ away from the original point). False accepts describe the opposite scenario, where a click falls outside of the centered-tolerance area but is accepted because it is still contained within the correct robust discretization grid-square (as far as $5r$ pixels from the original point). In the best case, the robust discretization square and the centered-tolerance square are perfectly aligned and the click-point is centered in the grid-square, but in practice the squares are offset, to some degree, 99% of the time for 19×19 pixel grid-squares.

6.2.2 Size of grid-squares

To be usable, the grid-squares must be sufficiently large to tolerate reasonable inaccuracies in targeting the original click-points. For example, to guarantee at least a 6-pixel tolerance around the original click-point using robust discretization, grid-squares must be 36×36 pixels ($6r \times 6r$). This will avoid rejects for login click-points that fall within 6 pixels of original click-point, but it will increase the potential for false accepts as a large area outside of the 13×13 pixel[1] centered-tolerance square will also be accepted. Furthermore, requiring such large grid-squares significantly reduces the theoretical password space for attackers. For example, a 640×480 pixel image contains only 252 36×36 grid-squares per grid, giving a theoretical password space of only 39.9 bits for a 5-click password, as opposed to 54.3 bits if centered-tolerance and 13×13 grid-squares ($r = 6$) were used. In comparison, the theoretical password space for a randomly generated 8-character text password is 52.5 bits for a standard 95-letter alphabet.

In essence, a usable implementation of robust discretization reduces security by significantly reducing the theoretical password space. This contradicts one of the major goals of a graphical password scheme [61], i.e., to achieve a larger theoretical password space (assuming large images are used).

6.3 Centered Discretization

Motivated by these observations, we propose *centered discretization*, which offers usability and security improvements. It offers centered-tolerance, which increases security because the size of grid squares can be reduced (to $2r \times 2r$ instead of $6r \times 6r$), thereby increasing the theoretical password space without negatively impacting usability since the same minimum tolerance r is guaranteed. It further increases usability by behaving in accordance with users' likely mental models and eliminating false rejects and false accepts. We first introduce centered discretization in 1-dimension, and then show how it can be expanded to 2-D for click-based graphical passwords or to higher dimensions.

[1]The extra pixel is to ensure an even 6-pixel tolerance around the original point.

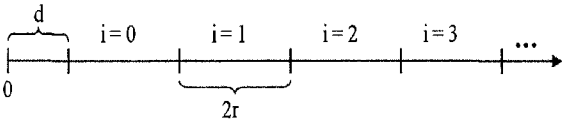


Figure 6.2: The continuous line L is divided into segments of length $2r$.

6.3.1 1-D centered discretization

Consider a 1-dimensional line, L, with a continuous set of data points. A particular point on this line is represented by a real number x. Our initial objective is to discretize this line into equal segments where x falls in the center of the segment containing it. This ensures an even tolerance on both sides of x. A tolerance r is selected based on system or user preferences. Each segment is of length $2r$ as shown in Figure 6.2. To ensure that x is centered in its segment, segment 0 may need to be offset from the origin. This offset is represented by parameter d.

First assume that a 1-D password consists of a single click-point x. To store this password, we must discretize the point by calculating its offset d (where $0 \leq d < 2r$) and its corresponding segment identifier i (where $i \geq -1$, with $i = -1$ occurring if x is within r of the origin). Offset d is stored in the clear, while i is stored in protected form as its hash value $h(i, d)$. The offset d is included in the hash to uniquely identify the segment. The system must also be aware of tolerance r that specifies the acceptable inaccuracy during password re-entry. The segment identifier i is computed by $i = \lfloor (x - r)/2r \rfloor$, identifying the segment containing x. The offset $d = (x - r) \bmod 2r$ determines the distance between the origin and the left boundary of segment 0.

To verify if a re-entered click-point x' is acceptable, the system computes $i' = \lfloor (x' - d)/2r \rfloor$. This calculates which segment contains x' using the same offset as the original point. Note that x' is not necessarily centered within its segment; we are simply calculating which segment contains x' based on x's pre-determined segments. If x' is within tolerance r of x, then $i' = i$ and hence $h(i', d)$ equals the stored value of $h(i, d)$ and system accepts the entry. If x' is outside of the accepted tolerance r, it falls in a different segment and $i' \neq i$, thus $h(i', d) \neq h(i, d)$ and the system rejects it.

For example, assume $x = 13$ and $r = 5.5$. We compute $i = \lfloor (x - r)/2r \rfloor =$

$\lfloor(13 - 5.5)/11\rfloor = 0$ and $d = (x - r) \bmod 2r = (13 - 5.5) \bmod 11 = 7.5$. Offset $d = 7.5$ is stored in the clear along with protected $h(i, d) = h(0, 7.5)$. If a user enters $x' = 10$ during login, the system calculates $i' = \lfloor(x' - d)/2r\rfloor = \lfloor(10 - 7.5)/11\rfloor = 0$. It then compares $h(i', d)$ and $h(i, d)$ and the click-point is accepted since they match. In practice, if a password consists of more than one click-point, all segment indices and their offsets are concatenated and hashed together as one. This stops attackers from matching individual points, and thus carrying out an efficient divide-and-conquer attack.

6.3.2 Applicability to 2-D spaces

Centered discretization can also be applied to click-based graphical passwords on a 2-D image. This is achieved by taking a point (x, y) in 2-D and discretizing each coordinate value individually along its corresponding axis. The segments along the x-axis can be combined with those of the y-axis to form a grid.

For example, if we use a tolerance value of $r = 9.5$ pixels,[2] then $2r = 19$ pixels. Thus the grid-squares will be 19×19 pixels. If we treat the click-point as coordinates on two 1-D lines, then the grid identifier will be composed of the offset for each dimension (d^x, d^y). Here, there are $19^2 = 361$ possible grids.

For a 5 click-point graphical password, each of the 5 click-points (x_1, y_1), ..., (x_5, y_5) will have an associated grid-square index (composed for the two 1-D segment indices) (i^x, i^y) and grid identifier (composed of the two 1-D offsets) (d^x, d^y). Grid identifiers $(d_1^x, d_1^y, \ldots, d_5^x, d_5^y)$ are stored in the clear, while the encrypted portion consists of:

$$h(d_1^x, d_1^y, i_1^x, i_1^y, \ldots, d_5^x, d_5^y, i_5^x, i_5^y).$$

To prevent a pre-calculated dictionary attack, a user identifier could be added to the hash (and also stored in clear-text), essentially serving as a salt. To address any concerns that offline attacks might be mounted to match hashed password values, the cost of such an attack could be increased by using iterated hashing, e.g., using h^{1000}

[2]In practice when dealing with graphical passwords and pixels, we add 0.5 to r to arrange for an odd number of pixels. For example, if the desired tolerance is 9, we need the width of the grid-square to be $(r+1+r)$ where 1 represents the original click-point's pixel centered in the grid-square. Adding 0.5 to each r accounts for this pixel.

effectively adds 10 bits of security ($1000 \approx 2^{10}$). By definition, original click-points in centered discretization are centered in their grid-square. The security implications of this design decision are discussed in Section 6.5.

Centered discretization may be expanded to n-dimensional objects for $n \geq 3$ by computing results for each dimension separately and then combining them to form an n-dimensional grid. While this paper discusses the applicability to 2-D images, other proposed graphical password schemes are based on 3-D spaces [2]. Such schemes currently allow users to select predefined objects in a virtual environment as possible click-points, limiting the theoretical password space to the number of predefined clickable objects. Moving to a scheme that allows discretization of an entire 3-D space could significantly enlarge the theoretical password space, depending on system parameters.

6.4 Usability Analysis

To understand the severity of false rejects and false accepts in practice, we implemented both robust discretization and centered discretization to analyze a large data set containing coordinates of passwords and login attempts for these passwords on a PassPoints system. This data was collected during the field study described in Chapter 3. The original prototype system implemented a centered-tolerance scheme without hashing to allow for the collection of information about the actual click-points. In total, 481 passwords were created and 3339 login attempts were recorded. Two different 451x331-pixel images were used; approximately half of the participants saw the Cars image (Figure 3.6) and the others used the Pool image (Figure 3.7).

For this current analysis, we used reconstructions to determine whether the actual login attempts in the collected data set would have been accepted if the system implemented each of the two discretization schemes discussed herein with various sizes of tolerance grid-squares. Our centered discretization scheme was fairly straightforward to implement since it involves centered-tolerance; if a login click-point was within centered-tolerance for some tolerance r of the original click-point, it was accepted, otherwise it was rejected.

Robust discretization proved more challenging. Implementation decisions, such as

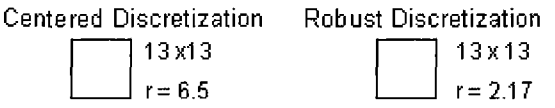


Figure 6.3: When the grid-square sizes are kept constant, r (the minimum guaranteed tolerance) is larger for centered discretization.

which grid to select when a click-point is r-safe in more than one grid, and how to deal with rounding when moving from real numbers to pixels, were not addressed in the earlier literature [9]. To avoid misrepresenting the scheme, we sought clarification from the original authors, and learned [12] that robust discretization was not implemented in their prototype system. Since they were not concerned with protecting password confidentiality in their usability studies [136, 137], their prototype stored all details in the clear and used essentially a centered-tolerance algorithm to determine whether a login attempt was successful. It is therefore an open question as to how false rejects and false accepts as defined herein would have affected usability and user success rates in earlier publications [136, 137], had robust discretization actually been used.

We attempted to implement an optimal robust discretization algorithm that minimized the occurrence of false accepts and false rejects. In cases where more than one grid was r-safe, we calculated the distance from the click-point to the grid edges and selected the grid where the point was closest to the center of the grid-square. To minimize rounding errors, we used real numbers for our computations and comparisons.

Occurrence of false accepts and false rejects

With centered discretization, the rate of false accepts and false rejects is zero by definition since centered-tolerance implies that the system will only accept click-points that are within r from the original point. With robust discretization, false positives occur when a click-point is accepted by the system but falls outside of the centered-tolerance grid square of the original point. Conversely, false negatives occur when a click-point falls within the centered-tolerance grid square of the original point but is rejected by the system.

There are two approaches to measuring false negatives and false positives. The

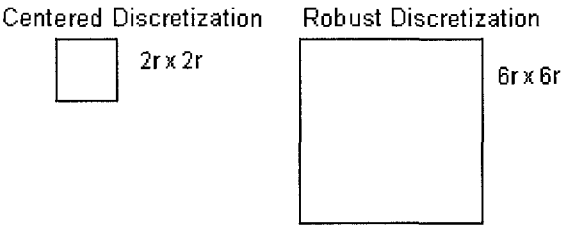


Figure 6.4: When r is kept constant, the grid-squares for centered discretization are smaller, so the theoretical password space is larger.

first is to assume that the centered discretization square is the same size as the robust discretization square (see Figure 6.3), but the robust discretization square may not be centered on the click-point. Table 6.1 shows the percentage of passwords that would have been falsely accepted and falsely rejected with robust discretization, with tolerance squares of the same size as centered discretization. For example, using the dataset as described in Section 6.4 with a tolerance square of 13×13 pixels, 21.1% of passwords are falsely rejected during login using robust discretization, but would have been accepted by centered discretization using a 13×13 grid (see Table 6.1). This likely indicates serious usability issues if a click-based graphical password scheme was implemented using robust discretization, since more than a fifth of passwords were falsely rejected.[3]

The second approach is to keep parameter r constant rather than the size of tolerance squares (see Figure 6.4). This means that the minimum guaranteed tolerance around a click-point is kept constant between centered discretization and robust discretization, but it also means that the robust discretization squares are much larger than the centered discretization squares. For this comparison, there can be no false rejects in robust discretization because everything within r is guaranteed to be accepted. However, the larger squares required by robust discretization lead to false accepts. For example, with $r = 6$, 14.1% of passwords are falsely accepted as correct in our dataset (see Table 6.2).

[3]Note that a false accept can only occur when a login click-point falls outside of the centered-tolerance grid-square, but because users contributing to the collected dataset [15] were very accurate in targeting their click-points, only a small fraction of login points fell outside of centered-tolerance and thus had the potential for being a false accept. When considering false accepts across all logins, the percentages (Table 6.1) may seem disproportionately low.

Table 6.1: False accept and reject rates for robust discretization when grid-square for both schemes are of equal size.

Grid Size	Robust Disc. (r in pixels)	False Accept	False Reject
9×9	1.50	3.5%	21.8%
13×13	2.17	1.7%	21.1%
19×19	3.17	0.5%	10.0%

Table 6.2: False accept and reject rates for robust discretization when r is the same as for centered discretization.

r (in pixels)	Robust Discr. Grid Size	False Accept	False Reject
4	24×24	32.1%	0%
6	36×36	14.1%	0%
9	54×54	4.3%	0%

The number of false accepts and false rejects seen with robust discretization raise usability concerns since the system will appear to perform erratically: accepting some clicks as correct when they are far from the original click-point and rejecting other clicks that should have been accepted from the users' perspective. The discrepancy between user expectations and system behaviour may lead users to feel frustrated and mistrust of the system. Furthermore, if a robust discretization system is implemented with reasonable-size grid-squares such as those recommended in the literature [15,21, 136,137], then the value of r becomes unreasonably small (in the range of 1-2 pixels), meaning that it is increasingly likely that click-points very near the original point are rejected. These problems have not been identified earlier because, as mentioned in Section 6.4, none of the original user studies [136, 137] were conducted on systems that implemented robust discretization.

6.5 Preliminary Security Analysis

Although the usability advantages are clear, to be acceptable centered discretization should provide at least comparable security as robust discretization. We examine how click-based graphical passwords implemented using both schemes withstand various

Table 6.3: Bitsize of the theoretical password space for 5-click passwords

Image Size (pixels)	Grid Size	Centered Discr. r (pixels)	Robust Discr. r (pixels)	# of Squares per Grid	Password Space for 5-clicks (bits)
451x331	9×9	4	1.50	1887	54.4
	13×13	6	2.17	910	49.1
	19×19	9	3.17	432	43.8
	24×24	11.5	4	266	40.3
	36×36	17.5	6	130	35.1
	54×54	26.5	9	63	29.9
640×480	9×9	4	1.50	3888	59.6
	13×13	6	2.17	1850	54.3
	19×19	9	3.17	884	48.9
	24×24	11.5	4	540	45.4
	36×36	17.5	6	252	39.9
	54×54	26.5	9	108	33.8

attacks and how the theoretical password space is affected.

The theoretical password space depends on both the size of an image and the size of the tolerance grid-squares, with larger images and smaller tolerances leading to a larger theoretical password space. Table 6.3 shows how these two variables affect the theoretical password space. While the table is organized by grid size, it is also possible to see the smaller password space for robust discretization when r is equal in both schemes, due to robust discretization's larger grid squares. For example, on a 640×480 image the theoretical password space is 59.6 bits for $r = 4$ using centered discretization but only 45.4 bits for robust discretization.

6.5.1 Human-seeded dictionary attacks

We attempted to crack PassPoints passwords from our field study (from Chapter 3, with important details summarized in Section 6.4) using passwords collected from our PassPoints lab study (described in Chapter 3). We used the click-points collected in the lab study and generated a dictionary containing all possible 5-click-point permutations as entries. Thirty lab passwords were used for each image, giving dictionaries with $\binom{150}{5} \approx 2^{36}$ entries for the Cars and Pool images separately. Our dictionaries

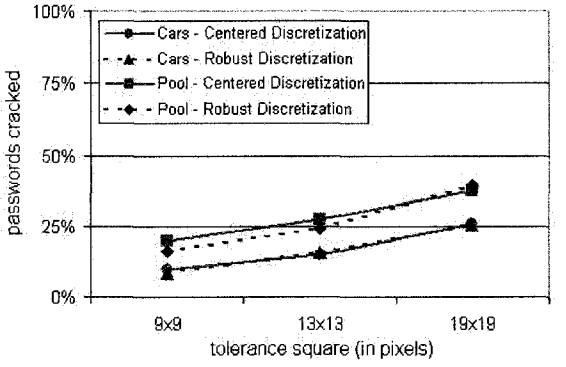


Figure 6.5: Offline dictionary attack with known grid identifiers for robust and centered discretization with a 36-bit dictionary and equal grid-square sizes assumed.

represented the simplest attack dictionary that could be built with 30 collected passwords per image. This is similar to the approach of Thorpe and van Oorschot [119].

Offline dictionary attack with known grid identifiers

The first scenario assumes that attackers have access to the clear-text grid identifiers and hash values stored by the system. In a targeted attack against a specific user, this reduces the theoretical password space since each guess can be mapped directly to the user's stored grid identifiers to compute the hash rather than having to iterate through all possible grid combinations. For example, if an attacker knows that user A's grid-identifier for the first click-point is $(d^x, d^y) = (10, 10)$, all guesses for that click-point can be discretized using this grid. This may occur in an offline attack if attackers gain access to the server-side files containing the grid identifiers and hashed passwords.

Using our dictionary of 5-click-point passwords, we searched for matches to passwords collected in the field study (which collected 162 passwords for the Cars image and 187 for Pool). For a successful match, all click-points in a dictionary entry had to be within the grid-squares of the user's click-points. The grid-squares were computed using either robust discretization or centered discretization and we calculated how many matches were made under each scheme.

We initially kept the size of the grid-squares constant (as shown in Figure 6.3) for both schemes. As expected, they performed similarly under this condition (see

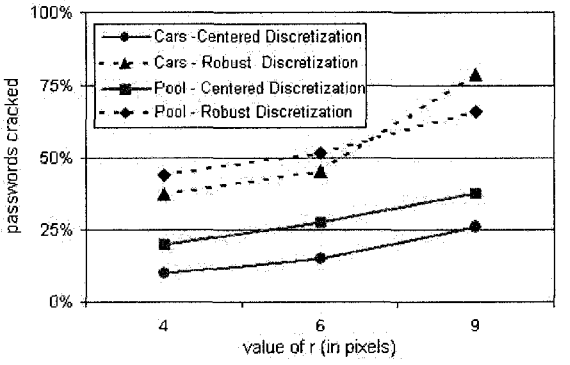


Figure 6.6: Offline dictionary attack with known grid identifiers for robust and centered discretization with a 36-bit dictionary and equal r-values assumed.

Figure 6.5) since having grid-squares of similar size means that roughly the same number of guesses would be accepted as correct.

Conversely, if we keep r constant across both schemes as in Figure 6.4 (to ensure similar usability in terms of the guaranteed size of the tolerance around a click-point), then centered discretization is significantly more secure in the face of this particular attack strategy since its grid-squares are much smaller (with comparable usability). Many guesses that are successful within robust discretization's larger grid-square are rejected by centered discretization. For example, Figure 6.6 shows that with $r = 6$, 14.8% of Cars passwords are cracked with centered discretization, as compared to 45.1% for robust discretization. With $r = 9$, robust discretization reaches 79% of passwords cracked. For this flavor of dictionary attack where the grid identifier is known, centered discretization can be more secure than robust discretization because smaller grid squares can be used without negatively affecting usability.

As mentioned earlier, this type of attack may be slowed or stopped by including a user identifier as a salt for the hashed values, forcing attackers to re-compute all of the hash values for every user. This can be made even more computationally expensive by using iterated hashing so that each password guess requires more computational effort.

We assume that if attackers gain access to the password file, they will have access to both the hash values and the clear-text grid identifiers. However, in the unusual case where only the hashed passwords are known, the size of attack dictionaries to

have the same attack efficacy would have to increase significantly. For each dictionary entry, attackers would need to compute a hash for each possible grid identifier combination. This would require significantly more work for centered discretization since the number of grids is proportional to the size of the grid-squares (13×13 grid-squares implies $13^2 = 169$ grid identifiers). Conversely, robust discretization has only 3 possible grids.

Online dictionary attack

Alternatively, attackers without access to the password file may attempt an online attack. While attackers may not explicitly know the grid identifiers, these are not necessary since the system will automatically use the correct grids when interpreting the login attempt. The attacker need not worry about pre-determining hash values. The attacker enters each guessed password through the regular login user interface to see if the system accepts it. The system may limit the number of incorrect login attempts for individual accounts, slowing or stopping the attack, but multi-account attacks are still possible. As with the offline attacks, smaller grid-squares mean that guessed click-points must be much closer to the real password click-points in order to be accepted so the theoretical password space is increased.

6.5.2 Information revealed

Robust discretization requires 2 bits of information to store one of its three grid identifiers, whereas centered discretization as proposed herein needs $log_2(2r * 2r)$ bits (e.g., for $r = 8$, this equals 8 bits). As the grid identifiers are (by design) stored in the clear for both schemes, they may be accessible to an attacker. This may have security implications, however, to our knowledge this does not lead to weaker security for the attacks discussed so far.

In the case where attackers have gained access to both the grid-identifier and the image, visual information may be leaked. Attackers may overlay the grid onto the image to see which parts of the image fall near the center of the grid-squares and thus may be able to predict which squares have a more likely click-point (either by using knowledge of hotspots or by personally evaluating the image). This may allow

prioritization of entries in the attack dictionary to test more likely entries first. With centered discretization, a single pixel at the center of each grid-square is identified, while for robust discretization, a central region is revealed. Knowing the center pixel does not appear to provide much advantage for attackers over knowing the center region since guessed click-points are correct as long as they are within the correct grid-square and the items targeted by users as click-points are usually much larger than a single pixel. However, we have not yet pursued this attack strategy sufficiently to have full confidence, and it is possible that combining this information with knowledge of hotspots may lead to new attacks on centered discretization. Our future work includes a study examining this issue (see Section 9.3).

6.6 Conclusion

So far, usability testing of click-based graphical password systems has used a centered-tolerance discretization approach. Robust discretization, as proposed by the creators of PassPoints, may well make these schemes less usable. Our results suggest that this would be the case, but since our analysis was conducted post hoc, it is unknown whether users of a robust discretization system would resort to some kind of compensatory behaviour. This still indicates usability issues, however, since users would be responsible for coping with the system's behaviour.

This chapter provides the first analysis of how the usability and security of click-based graphical passwords are affected by the type of discretization implemented. We identified weaknesses in robust discretization that lead to false rejects and false accepts, which we expect makes the system appear unreliable from the users' perspective. To compensate, robust discretization must use larger tolerance squares, which reduces the theoretical password space considerably, thus making it more susceptible to attack. Our proposed centered discretization scheme guarantees centered-tolerance, increases the theoretical password space since smaller grid squares can be used, and makes graphical passwords more usable in real systems by making system behaviour more predictable, since the tolerance square is centered on the original click-point (avoiding false accepts and false rejects). It remains open to further study whether centered discretization opens the door to new types of password attacks.

Chapter 7

Patterns in Graphical Passwords

In this chapter, we focus on how the design of the user interface influences users and may encourage either secure or insecure behaviour. Our post-hoc analysis looks at click-point patterns within passwords and shows that PassPoints passwords follow distinct patterns. Surprisingly, these patterns occur *independently of the background image*. Conversely, Cued Click-Points (CCP) and Persuasive Cued Click-Points (PCCP) passwords are nearly indistinguishable from those of a simulated dataset.

To better understand effective password spaces and the characteristics of user interfaces that can influence users towards more secure behaviour, we analyzed datasets collected through our PassPoints (Chapter 3), Cued Click-Points (Chapter 4), and Persuasive Cued Click-Points (Chapter 5) user studies and compared them to simulated datasets. The simulated datasets represent passwords that would occur if all passwords were equally likely and thus used the full theoretical password space. Our analysis is not driven by specific hypothesis, but rather by exploratory post-hoc questions aiming to identify patterns in click-points, and distinguish their presence in the different variant schemes. In post-hoc analysis, it is important to avoid the misleading situation where many directions are pursued, but only those which lead to significant results are reported. Therefore, we report on all of our pattern investigations, regardless of their results. The work from this chapter is available as a technical report [17], and has been submitted to an academic journal.

From previous chapters, we know that hotspots are a problem in PassPoints and in CCP. We now investigate whether users select their click-points in geometric patterns. In parallel work, Salehi-Abari et al. [101] recently found that automated dictionary attacks where click-points are ordered according to horizontal or vertical lines, or general diagonal direction were successful on PassPoints passwords. They used their approach to attack the Pool and Cars data from from our field study of PassPoints.

Table 7.1: Number of participants, click-points, and passwords per lab study. Note that only passwords where users were successfully able to confirm and login are used in our analysis and included in this table.

Study	Number of participants	Total number of click-points	Total number of passwords
PassPoints (PP)	43	2800	560
CCP	57	2520	504
PCCP	39	1500	300

7.1 Methodology

Our analysis compares data from our three lab studies: PassPoints (PP), Cued Click-Points (CCP), and Persuasive Cued Click-Points (PCCP). Table 7.1 summarizes the number of participants, passwords, and individual click-points collected. More points per image were collected for PassPoints (PP) since each user password gave 5 click-points on an image, whereas for CCP and PCCP, there was only one click-point per image.

We also analyze data from the PassPoints Field (PPField) study discussed in Chapter 3. In the field study, we collected 116 passwords (580 click-points) on the Pool image (Figure 3.7) and 109 passwords (545 click-points) on the Cars image (Figure 3.6).

Besides analyzing the datasets for patterns, we wanted to see whether the datasets differed from randomly-generated datasets. For this, we used a modified Monte-Carlo approach of generating simulations. For each study (PP, CCP, PCCP, PPField), we generated 100 simulated datasets, each containing the same number of passwords as the corresponding original dataset. Each password consisted of 5 pairs of (x,y) coordinates, corresponding to 5 click-points. These simulated datasets approximate passwords taken from the full theoretical password space, where all passwords are equally probable. They were generated using R's [58] random number generator function for uniform distributions (*runif()*).

In the present chapter, we are using these datasets to explore a new question: *how does user interface design affect security in these similar graphical password schemes, and what patterns of user choice emerge as a result of the different interfaces?*

7.2 Analysis of User Choice

Patterns in user choice reduce the effective password space and are advantageous to attackers who can use this knowledge to modify their attack strategy and increase the likelihood of success. Previous studies [16, 35, 50, 119, 126] show that when attackers know the images used to create passwords, they can determine likely hotspots and use this information to successfully attack PassPoints and CCP passwords. In the following sections we show that patterns emerge even without knowing the images. We look at several different password characteristics to see which ones reveal patterns that could help attackers fine tune their attack strategy.

We focus mainly on data from the lab studies because the methodologies are the same and the studies cover a wide range of images, reducing the risk of getting results that are an artifact of a particular image. In the following analysis, data from the three lab studies (PassPoints, CCP, and PCCP) are examined and compared against the randomly-generated datasets. The number of passwords and individual click-points for each dataset is available in Table 7.1. Unless otherwise indicated, all analyses of PassPoints refers to the dataset from the lab study (not the field study also mentioned in Section 7.1).

For each measure in the following analysis, we also calculated the results for each of the simulated datasets. We then determined the maximum and minimum median values among the 100 simulated datasets corresponding to a given study. These minima and maxima indicate the range of random values. Any collected result that falls outside of this range did not occur by chance, with a 99% probability. This is because each simulation represents a chance to include the observed value. If this does not happen after 100 simulations, this suggests that there is less than one chance in 100 that it might do so at random. Therefore if median values for our real datasets fall outside of this minimum-maximum range, it is likely because some pattern exists in the dataset that did not occur by chance. In all of the subsequent figures, we have represented these minima and maxima as lines to more clearly observe patterns, but the data is not continuous.

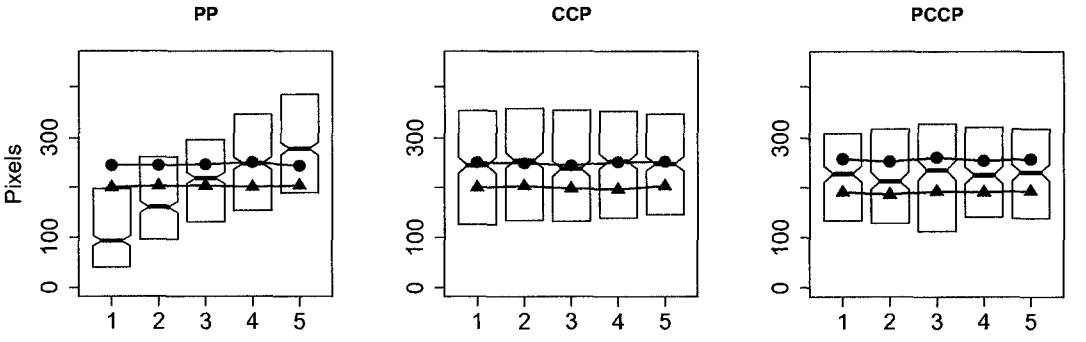


Figure 7.1: The box plots show the distribution of click-points along the x-axis of the image, grouped and ordered by click-point number for the three original datasets. The image dimensions were 451 × 331, therefore 451 is the maximum possible x-coordinate. The red line (with circles) and the blue line (with triangles) represent the maximum and minimum median values among the simulated datasets, respectively.

7.2.1 Click-point distribution

Are click-points distributed in some recognizable manner independent of the background image? We found that when selecting 5 click-points on a single image (as in PassPoints), users tend to select their first point towards the top-left of the image and progressively move towards the bottom-right with each subsequent click-point. This was not the case when users only selected one click-point per image (as per CCP and PCCP).

Figure 7.1 shows the distribution of click-points along the x-axis of the image.[1] The origin (0,0) is at the bottom-left of the image. The box plots represent the original datasets, while the blue and red lines respectively represent the minimum and maximum median values for the random simulated datasets. If the medians for the real datasets fall outside of the lines, then this pattern did not occur by chance with 99% probability. With PassPoints, there is a clear progression from the left side of the image for the first click-point towards the right for fifth click-point. The same occurs for the y-axis, as demonstrated in Figure 7.2; PassPoints click-points progress from the top of the image towards the bottom. Note that our participants

[1]Notched box plots can be interpreted as follows. The thick line in the narrowest part of the box represents the median. The box represents the center quartiles (25th to 75th percentile). The notches surrounding the median represent the confidence intervals. If the notches of two boxes do not overlap, then they are significantly different from each other at $p < 0.5$.

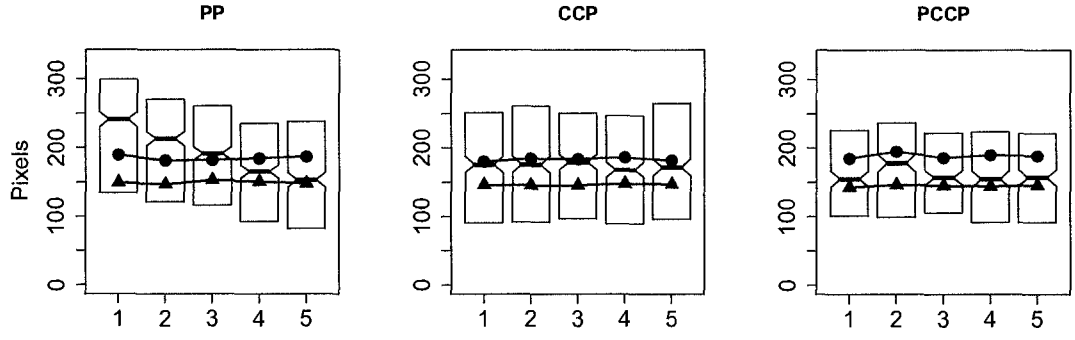


Figure 7.2: The box plots show the distribution of click-points along the y-axis of the image, grouped and ordered by click-point number for the three original datasets. The image dimensions were 451 × 331, therefore 331 is the maximum possible y-coordinate. The red line (with circles) and the blue line (with triangles) represent the maximum and minimum median values among the simulated datasets, respectively.

were volunteers from an environment where Western (top-down, left-right) writing and reading is dominant; we suspect that a tendency towards right-to-left or other distributions may be evident in other cultures. With CCP and PCCP, the click-points are quite uniformly distributed along both the x- and y-axes, regardless of the click-point number, as Figures 7.1 and 7.2 also illustrate. For PassPoints, the medians fall outside of the random range for three of the five click-points, while all of CCP and PCCP's medians fall within range of the simulated datasets.

Regression analysis shows that for PassPoints, there exists a strong relationship between the click-point number and its position on the x- and y-axes. For the x-axis, $F(4,2795)=123.7$ and $p < .0001$, and $F(4,2795)=30.2$ and $p < .0001$ for the y-axis. No such relationship exists for CCP, PCCP, or the simulated datasets. With PassPoints, it is possible to determine which areas of the image are more likely to contain click-points based entirely on the click-point number, without knowledge of the image used. For example, looking at Figure 7.1 we see that 75% of the first click-points fall within the first 200 pixels (out of 451 pixels) on the x-axis. Contrarily, the click-point number is not a predictor of click-point location for CCP and PCCP.

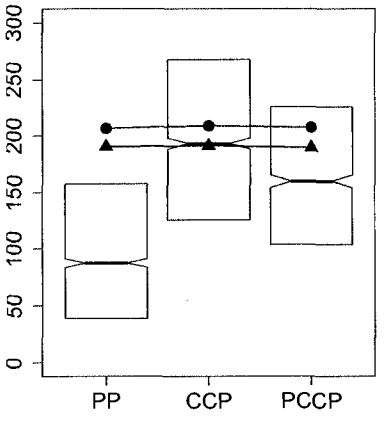


Figure 7.3: The box plot shows the distance in pixels between two adjacent click-points in a password (segment length) for the 3 original datasets. The red line (with circles) and the blue line (with triangles) represent the maximum and minimum median values among the simulated datasets, respectively.

7.2.2 Segment lengths

We next looked at the length of the segments formed between two adjacent click-points. If attackers can predict the likely distance between click-points, they could prioritize guesses containing click-points that are approximately that distance apart.

Figure 7.3 illustrates the distance in pixels between adjacent click-points in each dataset. For example, in PassPoints, the median segment length is 87 pixels while the median for CCP is 193 pixels. Adjacent click-points in PassPoints are more closely positioned, with very few individual segments spanning the entire image. This PassPoints click-point distribution is statistically different from the simulated datasets ($t(2288.92) = 45.30$, $p < .0001$)). An attacker may be able to use this information to predict higher probability click-point combinations, again even without knowledge of the specific image.

On the other hand, CCP segment lengths are more evenly distributed and are indistinguishable from those of the simulated datasets. The PCCP dataset, however, appears distinct from the simulated datasets for segment lengths ($t(1231.89)=14.17$, $p < .0001$). Figure 7.3 confirms that PCCP segments are shorter than those of the random sets. We were surprised by this result and suspect that it may have occurred

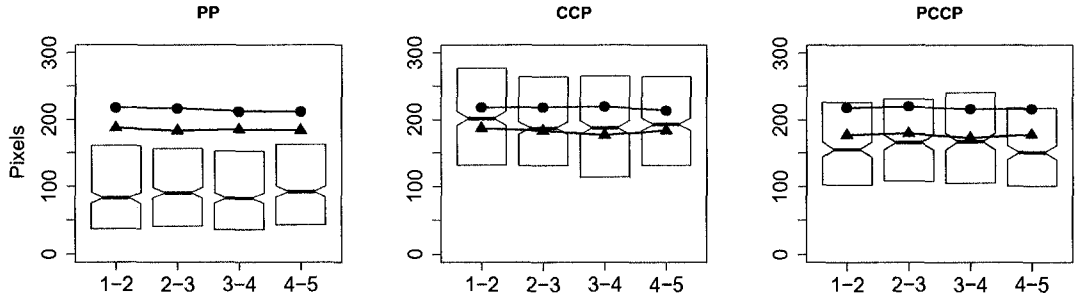


Figure 7.4: The box plots show the segment lengths grouped by segment number for the 3 original datasets. The red line (with circles) and the blue line (with triangles) represent the maximum and minimum median values among the simulated datasets, respectively.

as a side-effect of the viewport positioning algorithm, or it may be that users were more likely to select a click-point towards the center of the viewport and so the edges of the image were less likely to be selected.

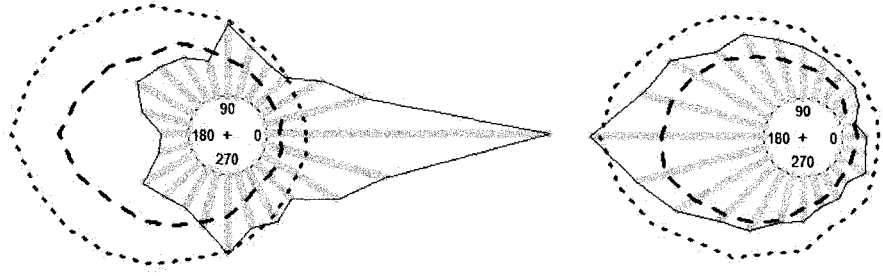
We also examined whether the segment number had any effect on segment length. Segment lengths appear consistent regardless of their position within the password (Figure 7.4). Regression analys confirmed that there were no statistically significant relationships between segment number and segment length for any of the datasets.

7.2.3 Angles and slopes

Users of PassPoints tend to create a straight line with their click-points, as evidenced in Figure 7.5.[2] The PassPoints diagram shows that the most common angles formed between two line segments are near 0 degrees, indicating that the users often selected click-points in a straight line, heading in the same direction. In comparison, CCP, PCCP, and the simulated datasets favour large angles resulting from back and forth motion between click-points.

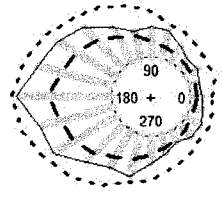
The distribution of segment slopes relative to the x-axis in PassPoints (Figure 7.6) shows that users strongly favour horizontal lines (0 degree slopes), followed by vertical segments in the downward direction (270 degree slopes). The slopes for the CCP

[2]Figures 7.5, 7.6, and 7.12 use circular diagrams to summarize angle data. These can be interpreted as circular frequency distribution diagrams. The distributions appear flattened because of the rectangular shape of the images from which this data was collected (451 × 331 pixels).



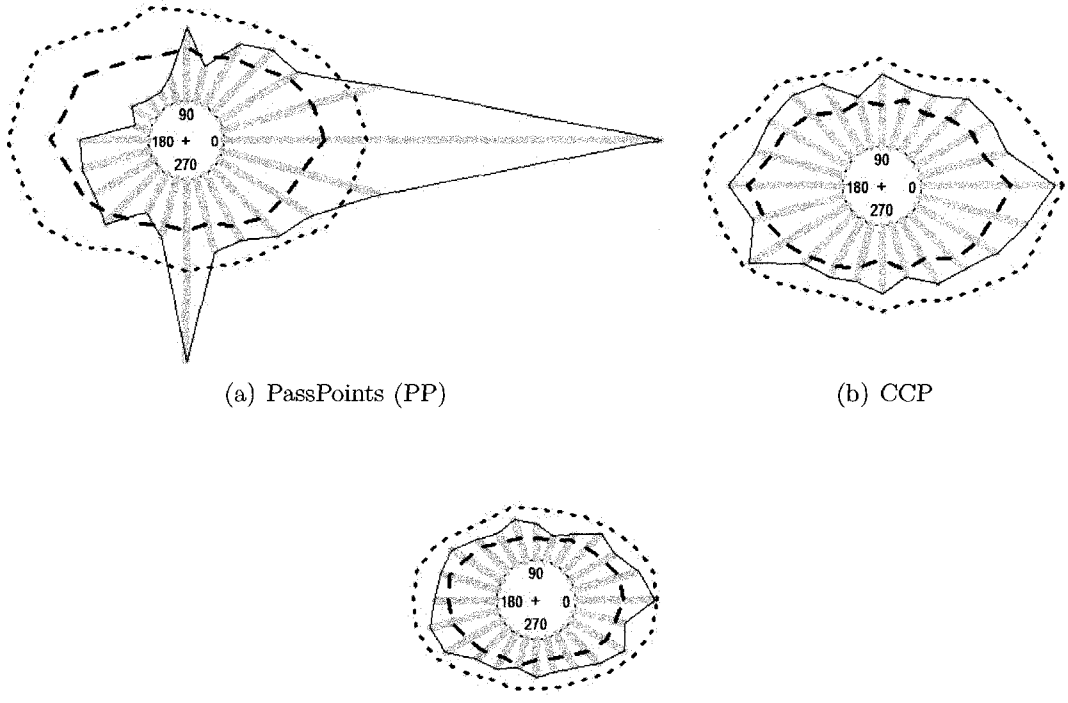
(a) PassPoints (PP)

(b) CCP



(c) PCCP

Figure 7.5: Frequency distribution of the angle (in degrees) formed between two adjacent line segments. These line segments are formed by joining two consecutive click-points in a password. The grey bars and black line represent the original dataset. The red dotted line and the blue dashed line show the maximum and minimum median values among the simulated datasets, respectively.



(a) PassPoints (PP)

(b) CCP

(c) PCCP

Figure 7.6: Frequency distribution of the slope (in degrees) of each line segment, relative to the x-axis. Line segments are formed by joining two consecutive click-points in a password. The grey bars and black line represent the original dataset. The red dotted line and the blue dashed line show the maximum and minimum median values among the simulated datasets, respectively.

Table 7.2: Shape classification scheme

Shape	Description
Line	The sum of the absolute values for all 3 angles is less than 15 degrees.
W	Angle 1 and angle 3 have the same sign (turn in the same direction) and angle 2 has the opposite sign.
Z	Two of the angles have opposite signs (turn in opposite directions) and the third angle is less than 15 degrees (forms a straight line).
V	Two of the angles are less than 15 degrees and the third angle is greater than 15 degrees.
C	All 3 angles have the same sign (turn in the same direction) and the sum of the absolute values for all 3 angles is greater than 180.
Other	Anything that does not fall into another pattern described above, i.e.,"no pattern".

and PCCP datasets are quite evenly distributed, which matches the slopes from the simulated datasets. Of the three systems, only PassPoints is distinct from the simulated datasets.

We further investigated whether angle number or slope number had any effect on the angle or slope respectively. We found no evidence of such interaction. In other words, the likelihood of finding a given angle (or slope) was not impacted by its ordinal position within the password.

7.2.4 Shapes

We also looked at shapes formed by all 5 click-points and the line segments between adjacent points. Our classification scheme identified 5 different categories of patterns, as detailed in Table 7.2 and Figure 7.7. For example, click-points may form a W pattern. A password was classified into this category if the line segments formed this particular pattern, regardless of orientation; a sideways or upside down W was still considered a W, as illustrated in Figure 7.7. The password shapes were identified by following the path formed from the first to last click-point sequentially, as entered by the user.

Once again, we found that the PassPoints dataset was easily distinguishable from

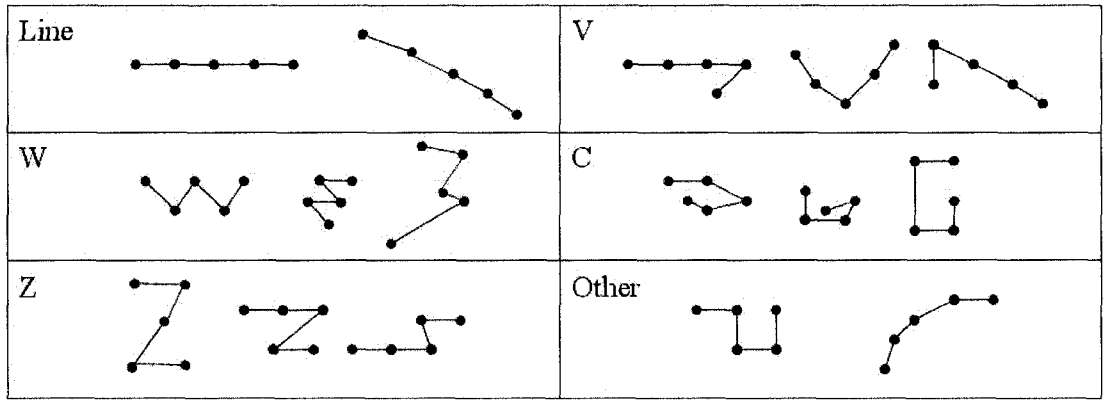


Figure 7.7: Example click-point patterns for each category. These represent the path formed by the sequence of points as entered by the user, proceeding in constant direction from one end of the pattern to the other.

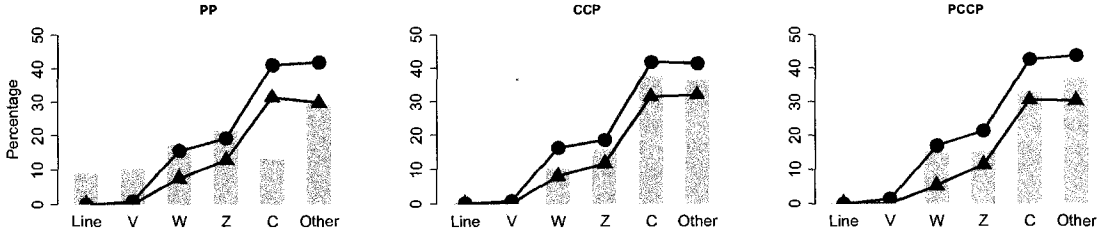


Figure 7.8: The bar graph shows the percentage of passwords in each shape category for the 3 original datasets. The red line (with circles) and the blue line (with triangles) represent the maximum and minimum median values among the simulated datasets, respectively.

the simulated datasets ($\chi^2(5,56560)=6798.67, p < .0001$). PassPoints includes simpler shapes, with far more passwords forming lines and V-shape patterns. Figure 7.8 reveals how PassPoints is distinct from CCP, PCCP, and the simulated datasets. Chi-square tests revealed no statistically significant difference between either of the CCP and PCCP datasets and their corresponding simulated datasets.

7.2.5 Analysis of the PassPoints field study (PPField)

The PassPoints field study [15], as previously mentioned, offers an opportunity to look at "real-world" passwords used over an extended period of time. It provides evidence of the types of passwords that one may expect to see if such a system was deployed. However since only two images were used, the patterns may be a direct result of the Pool (Figure 3.7) and Cars (Figure 3.6) images. We present the patterns found, but caution that further work is required to determine whether these occur across different images as well.

Figure 7.9 reveals that in the PassPoints field study, the click-point number has an effect on the x-coordinates of the click-points but not on the y-coordinates. The lack of interaction for the y-axis is likely a result of the Cars image since users frequently selected their click-points in a horizontal line across a row of cars. This is further supported by Figure 7.10 which shows that 24% of passwords followed a straight line. A further 17% had only one bend, forming a V-shape. Figure 7.12 also shows users' preference for straight lines since the most popular angles and slopes are very near 0 degrees. The slopes diagram (Figure 7.12(b)) further highlights that users preferred horizontal or vertical directions, with peaks near 0, 90, 180, and 270 degrees.

The median segment length for the PassPoints field study matches the median for the PassPoints lab study (Figure 7.11). This shows that even in the field study, users still tended to select adjacent click-points in close proximity to each other.

The PassPoints field data certainly exhibits click-point patterns; although some of these may be side-effects of the Pool and Cars images. We expect that they may also be partially attributed to users trying to select more memorable and simple passwords for two reasons: they had to remember PassPoints passwords over a longer period of time, and they had to actually use their passwords on a regular basis to access their

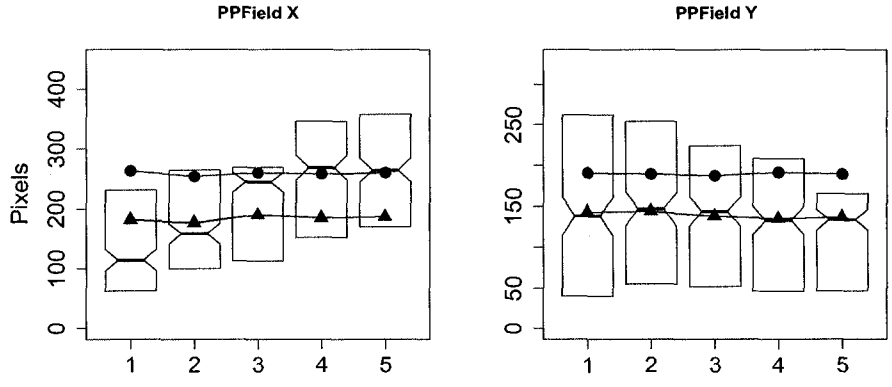


Figure 7.9: The box plots show the distribution of click-points for the PassPoints field study along the x- and y-axes of the image, grouped and ordered by click-point number. The image dimensions were 451×331, therefore 451 is the maximum possible x-value and 331 is the maximum y-value. The red line (with circles) and the blue line (with triangles) represent the maximum and minimum median values among the simulated datasets, respectively.

class notes. This serves as further cautionary evidence that user behaviour tends towards the easiest path when using these systems in a practical setting.

7.3 Discussion and Conclusion

Previous studies [15, 35, 50, 119, 126] have shown that hotspots occur in PassPoints and some mild evidence of click-point patterns [101]. Our present analysis provides considerably more evidence of click-point patterns. Our analysis revealed that click-point coordinates, segment lengths, angles between segments, segment slopes, and shapes formed by click-points can all be used to identify patterns in user passwords when all click-points are on a single image. Interestingly, these same patterns were not apparent when click-points within a password were based on separate images. For example, users of PassPoints prefer straight lines, with click-points that are roughly evenly spaced across the image, starting from left to right, and either completely horizontal or sloping from top to bottom. These patterns were apparently independent of the specific image used. Conversely, CCP and PCCP do not display these same patterns and are very similar to the randomly-generated datasets based on the

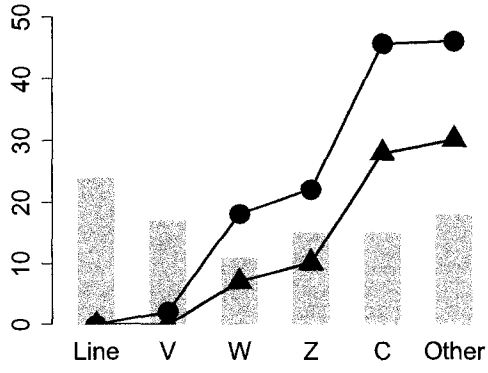


Figure 7.10: The bar graph shows the percentage of passwords in each shape category for the PassPoints Field study. The red line (with circles) and the blue line (with triangles) represent the maximum and minimum median values among the simulated datasets, respectively.

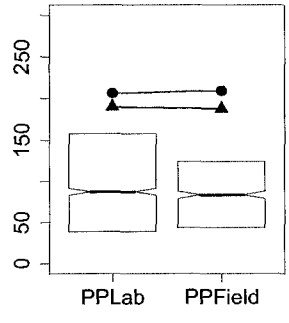
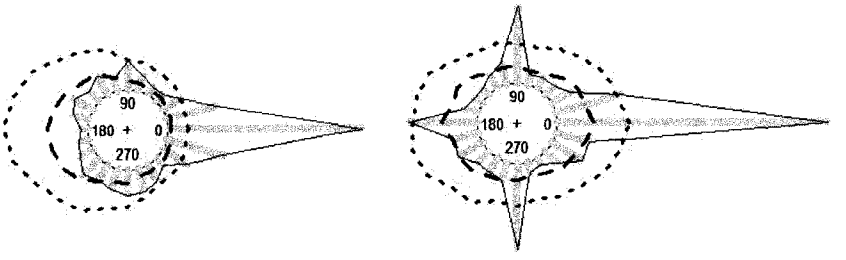


Figure 7.11: The box plot represents the line segment lengths for the PassPoints lab (PPLab) and PassPoints field (PPField) studies. Line segments are formed by joining two consecutive click-points in a password. The red line (with circles) and the blue line (with triangles) represent the maximum and minimum median values among the simulated datasets, respectively.



(a) PassPoints Field Angles (b) PassPoints Field Slopes

Figure 7.12: Frequency distributions of angles between segments and segment slopes for the PassPoints field study. There are more data points in the slopes diagram since each password contains 4 slopes and only 3 angles, making the slopes diagram appear slightly larger than the angles diagram. The grey bars and black line represent the PPField dataset. The red dotted line and the blue dashed line show the maximum and minimum median values among the simulated datasets, respectively.

Table 7.3: Summary of hotspots and patterns in click-based graphical passwords

Measure	PP	CCP	PCCP
Hotspots	Yes	Yes	No
Patterns	Yes	No	No

pattern characteristics analyzed in this paper. We note that there may exist other patterns, which we have not examined.

In click-based graphical passwords, hotspot information may be combined with knowledge of common click-point patterns. We expect that knowledge of likely patterns could be effectively used to prioritize a dictionary of passwords comprised entirely of (or biased towards) component click-points found to attract attention, e.g., hotspots. As shown in recent work [101], dictionary of passwords could also be constructed based solely on the patterns, without knowledge of the particular image. Table 7.3 summarizes the susceptibility of each scheme to hotspots and patterns.

All three schemes (PassPoints, CCP, and PCCP) are based on the same fundamental idea that a password consists of 5 ordered click-points while the image (or images) acts as a cue to remember the click-points. Nonetheless, our results indicate important differences in usage which lead to patterns that a conservative defender must expect to be exploitable by attackers.

With PassPoints, users receive one image as a cue and must recall 5 click-points.

This may be a more challenging cognitive task and it may be that users resort to click-point patterns in an effort to cope. Alternatively, asking users to select 5 click-points on one image may simply afford the creation of patterns because it is the easiest strategy. If this is the case, the mere fact that a password consists of 5 clicks on one image leads to insecure behaviour and design choices such as "what type of images" become less significant, since the system is inherently less secure.

With CCP and PCCP, each image provides a cue for the corresponding click-point. The one-to-one relationship may be easier for users to remember, therefore reducing the tendency towards selecting an overall geometric pattern formed by the click-points. Also, as each image appears on the screen, it forces users to refocus and take in the new stimulus which may interrupt the thought process for forming a pattern. PCCP further tries to persuade users to select more random points through the viewport, making it much less convenient to select hotspots. Consequently, the easiest path is most secure.

Overall, we note that the implications of design choices need to be carefully considered when making security-related modifications to a graphical password design or user interface. For example, adding a sixth click-point to PassPoints may provide less of a security improvement than adding a click-point to PCCP. With PassPoints, our results suggest that an extra click-point is likely to extend an existing click-point pattern, whereas in PCCP the extra click-point would add considerably more randomness to the password. This is discussed further in Section 8.1.

User choice is heavily influenced by the design of the system. Previous work focused on how image choice led to the formation of hotspots. We show that relatively minor changes in the type of cueing used and feedback provided by the system can lead to a significant reduction in the occurrence of patterns, regardless of image choice. In the case of click-based graphical passwords, it appears that having multiple images within a password is a main factor in reducing patterns in user-selected passwords.

Chapter 8

Security Discussion

In previous chapters, we focused on susceptibility to dictionary attacks because of their relationship with user choice in password selection. This is the primary type of attack we sought to defend against in our design of Cued Click-Points (CCP) and Persuasive Cued Click-Points (PCCP). In this chapter, we step back and address how CCP and PCCP resist various other forms of attacks, as well as summarizing their vulnerability to dictionary attacks. For the purposes of this discussion, we assume that the images are stored server-side and that all communication is done through SSL/TLS.

8.1 Exhaustive Attacks

In an exhaustive (brute-force) attack, every possible password combination is tried, until a match (or matches if cracking multiple accounts) is found. Exhaustive attacks can be rendered infeasible by having a large theoretical password space so that they become too costly or resource-intensive.

The risk of online exhaustive attacks against a live system can be decreased by limiting the number of incorrect login attempts allowed on individual accounts before lockout, or by progressively slowing system response with incorrect login attempts. To circumvent this online defense, attackers may conduct a multiple account attack where they target any account. In this case, attackers get a (relatively small) number of guesses at each account before lockout over a (potentially very) large number of accounts, increasing the likelihood that at least some passwords will be guessed.

Exhaustive attacks can also be conducted offline, after an attacker gains access to some verifiable text. In these cases, attackers are limited only by the computing resources and time at their disposal. Strategies for protecting passwords (including

click-based graphical passwords) can be applied to increase the effort and time required by attackers to guess passwords in an offline attack. Salting [66] concatenates a string of characters to a password before hashing it for storage by the real system. This salt is user-specific and stored in clear, along with the hashed password, so that it can be concatenated with the user's input password during login. The resulting string is hashed and compared for a match against the stored hash. This effectively forces attackers to compute the hash for each candidate password on a per-user basis. The hashing function can also be massively iterated, in conjunction to salting, to further slow the process of preparing a candidate password. The additional processing time is not noticeable on the live system for a legitimate user during login since only one password needs to be hashed; but for an attacker trying to process a large number of guesses, this can have a significant impact on the efficiency of the attack.

Obviously, increasing the size of the theoretical password space is a desirable goal to help reduce the chance of success for both online and offline exhaustive attacks. The theoretical password space for CCP, PCCP, and PassPoints depends on number of pixels in the image (N), the area of the tolerance squares (M) in pixels, and the number of click-points in a password (c). The password space is determined by $(N/M)^c$; Table 8.1 provides the sizes of the theoretical password spaces for various parameter values.

In our user testing of CCP and PCCP, we have used images of $N = 451 \times 331$ pixels, tolerance squares of $M = 19 \times 19$, and $c = 5$ click-points per password. We chose these parameters to remain consistent with earlier PassPoints studies by Wiedenbeck et al. [135–137]. As initially discussed in Section 4.5, we can increase the theoretical password space by adjusting the parameters N, M, and c but these may have trade-offs of decreasing usability or increasing susceptibility to shoulder-surfing.

8.1.1 Increasing image size

Increasing the image size is a simple way of increasing the theoretical password space. With larger images, there will be more grid squares per image (with constant size of grid squares); thus potentially increasing the number of guesses required by an attacker to find the correct click-point location, especially using a naive exhaustive

Table 8.1: Size of theoretical password space for CCP and PCCP with different parameters

Image Size (M) in Pixels	Grid Square Size (N) in Pixels	Number of Grid Squares	Number of Click-points (c) per Password	Number of Passwords
451×331	9×9	1887	5	2^{54}
451×331	13×13	910	5	2^{49}
451×331	19×19	432	5	2^{44}
640×480	9×9	3888	5	2^{60}
640×480	13×13	1850	5	2^{54}
640×480	19×19	884	5	2^{49}
1024×768	9×9	9804	5	2^{66}
1024×768	13×13	4740	5	2^{61}
1024×768	19×19	2214	5	2^{56}
451×331	9×9	1887	6	2^{65}
451×331	13×13	910	6	2^{59}
451×331	19×19	432	6	2^{53}
640×480	9×9	3888	6	2^{72}
640×480	13×13	1850	6	2^{65}
640×480	19×19	884	6	2^{59}
1024×768	9×9	9804	6	2^{80}
1024×768	13×13	4740	6	2^{73}
1024×768	19×19	2214	6	2^{67}

approach to guessing. Increasing the theoretical password space can be a useful strategy, but large images could increase the threat of shoulder-surfing because larger images may be easier to distinguish from further away, or if only part of the screen is visible to attackers. However, attackers who learn only the correct sequence of images for CCP or PCCP still need to determine the exact click-points leading to that sequence. Shoulder-surfing is discussed further in Section 8.3.

On a client-server system where the images are being transmitted from the server, consideration should also be given to the size of the image files. With CCP and PCCP, the images must be requested one at a time, depending on the user's click-points, so transfer rates may be a concern with large images.

Although we have not yet tested larger images, we might optimistically predict that there would be little impact on usability and memorability. Increases in login time due to longer mouse movements may have little practical effect, but could be estimated using Fitts' law [71] for different image dimensions and our current knowledge of patterns in click-point distributions on an image. The memorability of

click-points on a larger image would also need to be examined more closely. An initial investigation of the effects of larger images is planned, as discussed in Section 9.3.

8.1.2 Decreasing size of tolerance squares

We have evidence in Sections 3.1.2, 3.2.2, and 4.3.2 that users are very accurate when entering their click-based graphical passwords. Therefore, large tolerance areas may not be necessary for adequate usability, especially if using centered discretization (as discussed in Chapter 6).

By decreasing the size of tolerance squares, the grid becomes finer and the number of grid squares increases; thus increasing the theoretical password space. For example, moving from 19×19 squares to 9×9 squares increases the number of passwords from 2^{44} to 2^{54} on images of 451×331 pixels (see Table 8.1). Attackers conducting exhaustive searches will need more guesses to cover all possible passwords.

8.1.3 Increasing the number of click-points

Requiring that passwords contain more click-points can also increase the theoretical password space. This has a usability and memorability cost, however, as users are now responsible for choosing, remembering, and entering more click-points. An alternative would be to enforce a minimum password length, but allow for passwords of varying length. Under this configuration, a user who is concerned about security, and is willing to memorize extra click-points, could create a longer password.

Despite the extra usability cost, we suspect that adding a click-point to CCP or PCCP may be less strenuous for users than adding a click-point to PassPoints. CCP and PCCP offer one-to-one cued recall, so an additional click-point would also include an additional cue to help remember it. Furthermore, we saw in Chapter 7 that users of PassPoints were more likely to select click-points in a geometric pattern. Adding a click-point under these circumstances likely continues the pattern and, as such, offers a smaller relative boost in security. As an example, we consider our sample system with an image of 451×331 pixels and tolerance squares of 19×19 pixels (giving 432 grid squares). The theoretical password space for a 5 click-point password using these parameters is $432^5 \approx 2^{44}$ and $432^6 \approx 2^{53}$ for a 6 click-point password.

To illustrate the effect on security, we look at the case where passwords form a line. To simplify the analysis, we consider that after the first two click-points (which set the direction of the line), the line segment formed by each subsequent click-point can deviate by a maximum of 5° in either orthogonal direction from the previous click-point (10° total). For the first two click-points, any of the 432 grid squares are possible. For each of the remaining 3 click-points, we have (as a very rough approximation, ignoring that the image is rectangular-shaped) $432 \times (10°/360°) = 12$ choices because only tolerance squares falling within an arc of 10° are available if the click-points form a line. The total number of 5 click-point passwords forming a straight line, therefore, is $432^2 \times 12^3 \approx 2^{28}$. By the same logic, if we add an additional click-point and the password still forms a line, we have $432^2 \times 12^4 \approx 2^{32}$ candidate passwords. Under these parameters, we see that adding a 6^{th} click-point results in only a 4-bit gain in security (from 2^{28} to 2^{32}) when passwords form lines, compared to a 9-bit gain (from 2^{44} to 2^{53}) if no patterns are present and the full theoretical password space is used.

Geometric patterns were not evident in CCP or PCCP, so we expect that additional click-points in these two systems would offer more of a security enhancement than for PassPoints. It remains to be investigated whether the additional memory aids found in CCP and PCCP would be sufficient to avoid the use of geometric patterns when extra click-points are added to the password. As described in Section 9.3, we are in the process of examining the usability and memorability effects of varying parameters such as increasing the number of click-points. However, this is beyond the scope of this thesis.

8.2 Dictionary Attacks

Attackers conduct dictionary attacks by identifying passwords with higher probability of being chosen by users and using this list to systematically try and guess passwords; in effect, attackers try to identify the effective password space (or portion thereof). This can dramatically improve the success ratio compared to an exhaustive attack, by lowering the expected number of guesses required for success. Dictionary attacks can be especially successful if entries are prioritized to test the most probable passwords

first. The disadvantage of dictionary attacks is that they require more design and pre-computation than exhaustive attacks since some preliminary work must be done to identify candidate entries for the dictionary. Dictionary attacks can be conducted online or offline, in a similar manner to exhaustive attacks and the same security precautions apply.

When creating text passwords, users typically select real words and use variations such as adding digits to the beginning or end of the word, or replacing some letters with symbols. Forming an attack dictionary that includes entries with these characteristics is likely to yield some success. Programs such as John the Ripper [30] employ these types of "word mangling" rules to create their dictionaries. In incremental mode, John the Ripper allows attackers to define additional rules to help prioritize guesses based on the particular type of passwords being attacked. Automated programs for guessing click-based graphical passwords are not widely available (compared to programs such as John the Ripper for text passwords); this is probably because these types of passwords are not widely deployed. However, as discussed in previous chapters, we have found that many PassPoints and CCP users also behave in a predictable fashion, so it is not unreasonable to expect that such software would be made available if click-based graphical passwords were deployed in practice. We now look at two strategies for creating dictionaries for click-based graphical passwords: using hotspots and using geometric patterns.

8.2.1 Hotspot dictionaries

Researchers [35, 101, 119], including the author and colleagues, have been able to generate click-point dictionaries that yield some success at guessing user choices. Certain areas of a given image are more popular than others (i.e., hotspots); if an attacker can determine areas that have a higher probability of being selected, then an effective click-point dictionary can be created. While work has been done in automating the process of determining hotspots through image analysis [35, 119], the most effective method of determining hotspots appears to be gathering click-points from a few users to form the basis of the attack dictionary. Recent work by van Oorschot and Thorpe [126], reports 7-10% success rates within 3 guesses using an

improved human-seeded dictionary attack on data from our PassPoints field study (Chapter 3).

With PassPoints, only one image per user needs to be analyzed to determine potential passwords using hotspots, and this image is available to attackers through the live system upon entering the username (if known) because the system must provide the image before the user can enter their password. In the most secure case, each user would be assigned a different image so attackers would need to perform preliminary work to analyze the image and determine probable passwords on a per-user basis. The password would be hashed using the username as a salt for storage, so attackers conducting an offline attack would also need to hash the dictionary on a per-user basis.

For CCP and PCCP at least several hundred images need to be processed per user. For example, if an image contains $s = 432$ tolerance squares (as in Section 4.1), then s next-images are needed at each stage (after the first stage, since the first image must be displayed by the system before the user enters their first click-point). We assume that for each stage, there is a percentage p of images re-used from previous stages. For example, if $p = .25$, then 25% of images will be re-used from previous stages and, therefore, 75% will be new at each stage. The total number of images required for this user can be determined by $I = 1 + s \times (c-1) - (c-2) \times s \times p$ where c is the number of click-points (i.e., the number of stages, we assume $c = 5$). This equation effectively calculates the total number of images required if there was no re-use, then subtracts the total number of images that will be re-used, resulting in the total number of images needed per user. In our example, $I = 1 + (432 \times 4) - (3 \times 432 \times 0.25) = 1405$. As a comparison, if we assume no reuse of images across stages, $p = 0$, therefore $I = 1 + 432 \times 4 - 0 = 1729$.

An attacker can retrieve the first image for CCP or PCCP from the live system by entering the username, but the remaining images are unknown and must be systematically retrieved one at a time by clicking on the current image. An attacker performing such an attack would have to process each image to determine hotspots, before clicking on each of these hotspots to retrieve the next set of images; the number of images grows exponentially with each click-point. This work would need to

be done on a per-user basis because the algorithm for mapping click-points to next-images is dependent on the username as parameter (as discussed in Section 4.1), and a different subset of images is assigned to each user. Although, with some image reuse across users, some images may have already been analyzed for hotspots. One strategy for improving security when images are reused would be to use larger images that are cropped in a different way for each user, so that hotspot information may not be immediately transferable. Furthermore, hotspot dictionaries would apparently be ineffective for PCCP because we have shown in Chapter 5 that click-points tend not to form hotspots across users.

8.2.2 Pattern dictionaries

Users may also select click-based graphical passwords in other predictable ways. As discussed in Chapter 7, we found that PassPoints users frequently chose their click-points in simple geometric patterns. Combining this pattern information with information about hotspots could lead to even more refined click-point dictionaries, similar to the approach recently taken by van Oorschot and Thorpe [126]. Salehi-Abari, Thorpe, and van Oorschot [101] further report success with an attack based entirely on patterns (in this case, horizontal or vertical lines, or a general diagonal direction) for the two images of the PassPoints field study from Chapter 3. This result matches our findings from Chapter 7, where we show that passwords that form lines are a popular choice for PassPoints users, and that passwords tend to follow left-to-right and top-to-bottom directions. We further believe that similar attacks using the other patterns identified in Chapter 7 would also lead to successful attacks on PassPoints passwords. However, we expect that CCP and PCCP may be less susceptible to pattern-based attacks because we found that passwords on these systems did not follow geometric patterns.

Based on the observed lack of hotspots and the lack of geometric patterns, PCCP passwords appear to be much more resistant to the types of dictionary attacks discussed here. However, it is possible that user behaviour is predictable in other ways that could be exploited by attackers to form attack dictionaries for PassPoints, CCP, and PCCP. For example, users may be more attracted to objects of a certain colour

or of a certain size. We did not explore these characteristics in this thesis. It is impossible to predict every potential pattern; and some patterns may only emerge in the future, once users have extensive experience with such systems, or once other external factors have an effect (e.g., the pattern of including "internet-speak" in text passwords due to mass usage of the internet).

8.3 Shoulder-Surfing Attacks

Shoulder-surfing is a targeted attack against a specific user. It can occur when it is possible to observe someone entering a password, either through direct observation or through some external recording device such as a camera or video camera, perhaps with a telephoto lens. Recently published papers discuss the ability to gain information from computer screens through telephoto images of reflections on other items near the computer [7] and the ability to duplicate physical keys based on images from telephoto lenses 195 feet away [70]. Obviously, shoulder-surfing is a general security threat not unique to graphical passwords, but since they use visual output on the computer screen, graphical passwords are also susceptible to shoulder-surfing.

Some recognition-based graphical passwords require that multiple successful logins be observed before the full secret can be deduced because only some of the user's portfolio images are displayed at each login or because the scheme does not require that users explicitly reveal the shared secret at login (e.g., as in Weinshall's scheme discussed in Section 2.4.3). However, most other types of graphical passwords can be gathered from observing or recording one successful login; click-based graphical passwords fall into this category. In their present form, we do not recommend that CCP or PCCP be used in environments where shoulder-surfing is a serious threat.

With CCP and PCCP, an observer needs to record the images and the precise mouse-clicks on each of these images, then be able to accurately reproduce the series of click-points. Partial information, such as only capturing the images, does not reveal the password but does leak sufficient information to help attackers. Using these captured images, attackers can then mount a divide-and-conquer attack since they now know exactly what sequence of images they are trying to achieve. If conducting an online attack, this presumes that attackers have a sufficient number of guesses

available for the particular account before being locked out by the live system. For example, let us assume that an attacker has learned the entire sequence of $c = 5$ images within a password but not the exact click-point locations. If the number of tolerance squares per image is $s = 432$, and we assume that all pixels are equally likely to be selected by users, we would expect that an attacker would need to guess 50% of tolerance squares on average before finding the correct one. The total number of guesses, therefore, would be $G = .5 \times c \times s = 1080$. The advantage for attackers is that they know when to stop guessing at each stage, so only need to try as many guesses as necessary to find the correct image before moving on to the next stage. This attack might be made more efficient by using hotspot or pattern information to prioritize their efforts.

The sequence of images observed for one user is of little (or no) use to help attackers guess passwords for other accounts. This is because the subset of images and the mapping from one image to the next includes the username as a parameter, so knowledge from one account will not be transferable to other accounts.

PassPoints passwords are also susceptible to shoulder-surfing. Attackers must gather information about the image and the precise click-points. We suspect, however, that it may be somewhat more difficult for attackers to gain partial knowledge of PassPoints passwords from a distance, unless a telephoto camera or video camera is used. If an attacker is too far away to see the mouse cursor, only the one PassPoints image is visible, and this information would be available anyway by entering the username (if known) at the login screen of the live system. In this case, attackers are no further ahead, but can still mount exhaustive or dictionary attacks against this particular image. If an attacker can observe the mouse cursor movements and deduce where the user clicked, then the entire password is known and can be reproduced.

Smaller tolerance squares may also reduce the risk of successful shoulder-surfing by either a nearby attacker observing the screen, or an attacker recording the screen using a high-powered telephoto camera lens [7, 70] (since the captured image may be too blurry to accurately identify the mouse pointer tip). With smaller squares, attackers must repeat mouse clicks with greater precision to correctly enter the password. Furthermore, observing mouse cursor movements alone may not reveal exactly

where the user clicked since the user may not necessarily stop moving the cursor with every click, especially when familiar with the pattern of mouse clicks. With CCP and PCCP, attackers who can clearly see the mouse pointer may be able to identify the last position of the mouse immediately before the next image appeared; this could be partially addressed by adding a (short) random delay before the next image appears. While attackers may be able to approximate the password, they are more likely to require several guesses and to run out of login attempts before finding a match than if the system allowed for larger tolerance areas.

Existing shoulder-surfing resistant or shoulder-surfing immune graphical password systems [67, 131, 138] have major usability drawbacks, usually in the amount of time and effort it takes to log in; as such they are typically not viable alternatives for everyday authentication. Click-based graphical passwords could be made more shoulder-surfing resistant by reducing the size of the images (which, however, consequently also reduces the size of the theoretical password space) or by manipulating the image and cursor on the screen, such as reducing the amount of contrast, to reduce the risk that observers can identify them from far away. These would need to be user-tested to ensure that the usability of the system remains acceptable.

Eye tracking has also been proposed as a shoulder-surfing resistant method of user input [68]. By entering a password using only eye gaze, no mouse cursor needs to be visible on the screen. With CCP and PCCP, however, the sequence of images may still be observed even if eye-tracking was used as an input device. Preliminary (unpublished) experiments by members of our group have revealed that eye tracking is not yet sufficiently accurate to be a viable approach. Furthermore, it is unclear if advances in the technology will improve precision enough for graphical password entry using eye tracking, or whether characteristics of human vision make eye tracking inherently imprecise. We are pursuing this line of inquiry, but it is beyond the scope of this thesis.

8.4 Phishing Attacks

Phishing is type of attack where attackers convince users to reveal their credentials at a malicious website, typically designed to look like a legitimate site for which the

user has an existing account. Attackers can then use these credentials to impersonate the user at the real website. For text passwords, often only a reasonable copy of a website's login page is needed along with a means of luring users to the site. The attacker gathers the username and password from the phishing site and enters it at the legitimate site. Users are typically led to the phishing site by a forged email, appearing to come from the legitimate company.

For CCP and PCCP, a more active role is necessary to capture the user's credentials through phishing. The attackers need to know the correct sequence of images to display in response to user input; something they do not know ahead of time. This is most commonly accomplished through a "man-in-the-middle" attack: the phishing website gets the username from the user, enters this username into the real website, retrieves the user's first image from the real website, displays this image on the phishing website, captures the user's first click-point, transmits that information to the real website, and so on. In effect, the attacker acts as a relay, intercepting all information to and from the user and the real website, and in the process succeeds in logging on to the legitimate website.

Although CCP and PCCP are susceptible to phishing when used in conjunction with a man-in-the-middle attack as described above, this is a more challenging attack than for text passwords (and PassPoints, as shown next). With PassPoints, the attacker must also know which image to display to the user on a phishing site before the user can log in. However, this image can be retrieved by entering the username (if known) at the legitimate site. The attacker may do this in real-time, as soon as the user enters the username at the phishing site. Although this is also a man-in-the-middle attack, only one contact is needed with the legitimate site during the attack, to retrieve the one PassPoints image. Attackers can then collect the click-points for later (or immediate) use at the legitimate site. Alternatively, a phishing site could pre-fetch the PassPoints images of the users it is targeting, if the usernames are known, and store them on the server. If one of these users is lured to the phishing site, the system can display the correct image immediately, without having to use a man-in-the-middle attack or having to access the legitimate site in real-time.

8.5 Social Engineering Attacks

Phishing is a specific type of social engineering attack, but social engineering can include any means of manipulating users into revealing their credentials for malicious purposes, such as phone calls from a fake help desk or credit card company. While these types of calls may require some background work to seem legitimate to users, it is often easier to convince users to reveal their password or other confidential information than it is to break into the system through other means [74].

Text passwords and other types of alphanumeric information are relatively easy to share with attackers (or friends) since they can be spoken or written down. Click-based graphical passwords are more difficult to share, even if a user is tricked into trying to do so. First, the user and the attacker must coordinate a frame of reference, describing the image in enough detail so that the attacker (masquerading as a well-intentioned associate, in most cases) understands the descriptions of the click-points on the image. With PassPoints, the user must first remember the image, unless it is in front of them, describe the image, and identify the 5 click-points. Dunphy et al. [37] conducted on a preliminary study where the experimenter described a password to a participant, who tried to enter the password based on the description. They report that 4 out of 5 participants were successful. The scenario is somewhat artificial, however, since the experimenter and the participant were looking at the same screen, so had a common frame of reference.

CCP and PCCP passwords are more difficult to reveal; users must somehow explain the exact location of their click-points based on characteristics of the images, after first ensuring that the other party is in fact looking at the correct image. The user and attacker must reorient themselves with each image and click-point. And, unless the user is also entering their password, the user must remember 5 images in enough detail to provide accurate descriptions. Although this is a security advantage, it does have usability drawbacks because it also means that users cannot receive a reset password by phone, for example.

If we consider other means of sharing the password, obvious methods include drawing and taking photos or screen shots. It would be difficult to get the required accuracy by drawing, and it assumes that the user somehow shares the drawings with

the attacker. A more efficient way of accurately sharing a click-based graphical password is to take screen shots of the images with the mouse cursor (or other indicator) in the correct positions to identify the click-points. These would need to be passed on to the attacker, perhaps by email. If the password must be transferred through electronic means, then a phishing attack is likely to more believable and simpler to accomplish than other types of social engineering attacks. If taken offline, users could print screen shots of their images, mark the click-points with a pen, and share these printed copies (or put them away for backup purposes). Overall, it appears that CCP and PCCP passwords would be moderately more difficult to gather through social engineering attacks than PassPoints, and significantly more difficult than text passwords.

8.6 Malware Attacks

Malware includes any unauthorized programs running on a computer. These can collect information from the hard drive or directly from the user's input, and transmit this information back to attackers, or make it available for retrieval.

Key-loggers can capture and keep a log of the user's typing, and as such can record text passwords. Attackers can then look through the captured data file (log), identifying likely usernames and passwords. Key-loggers do not provide enough information to reveal most graphical passwords, unless the scheme exclusively uses keyboard entry (e.g., inkblot authentication, as described in Section 2.4.4).

To collect PassPoints, CCP, and PCCP account information, an attacker would need to capture the user's keystrokes to collect the username, screen information for determining the image and its position on the screen, and mouse clicks to know when a click-point has been selected since cursor movement alone may not reveal the exact location of the click-points. This information would then need to be synchronized to accurately determine which mouse clicks correspond to password click-points on specific images. Although feasible, this is a more difficult than simply recording the keyboard input. A screen-scraper would be needed to collect the screen information, a mouse-logger to record mouse clicks for the exact location of the click-points, and then the two would need to be synchronized in time. Alternatively, it may be possible

for mouse-loggers to also capture information about the position of windows on the screen and use this information to determine image positions without the need for a screen scraper. We expect that if click-based graphical passwords became popular, then malware collecting the necessary information would soon follow.

Compromised computers are a significant threat against all of a user's information and computer resources, not only against a user's login credentials. This is a general security problem that will affect every authentication mechanism if used from an unsecured computer or using insecure communication channels. If there is malware on the end-user computer, then it is safest to assume that all resources and communications are compromised.

8.7 Conclusion

Due to our interest in usable security, in this thesis we have focused our analysis on dictionary attacks because their success is a direct result of user choice in password selection. Our general intent in designing CCP and PCCP was to find ways of increasing memorability of passwords while decreasing predictability. The best measure of predictability is to examine the passwords for patterns (as done in Chapter 7) and common traits (such as hotspots) that may reduce the effective password space.

In this chapter, we identified and provided an overview of several other threats to authentication mechanisms and discussed how these may affect our proposed click-based graphical password schemes. Table 8.2 summarizes CCP and PCCP's features based on the same security characteristics as the other graphical password schemes reviewed in Section 2.4. For completeness, we include Table 8.3, which covers the usability characteristics also covered in the same section.

We find that CCP and PCCP appear to be more secure against dictionary attacks than PassPoints and text passwords. CCP and PCCP may require more sophisticated strategies than PassPoints for phishing attacks. With respect to other types of attacks, CCP and PCCP appear no more susceptible than other schemes, with the possible exception of shoulder-surfing.

Table 8.2: Security comparison of CCP and PCCP schemes.

Scheme	Theoretical Pswd Space	Effective Pswd Space	Offline Attack	Shoulder Surfing	Phishing	Social Engineering	Malware
L. Cued Click-Points (CCP)	2^{44} (with $c = 5$ clicks, 451×331 pixel images, and 19×19 squares)	Hotspots, may be personally identifiable	Can be hashed, but grid identifier and images must be available to system	One login	Man-in-the-middle to retrieve images, one login to repeat	Possible with complex description of each image or screen shots	Screen or Mouse
M. Persuasive Cued Click-Points (PCCP)	2^{44} (with $c = 5$ clicks, 451×331 pixel images, and 19×19 squares)	No known hotspots or patterns, may be personally identifiable, but less likely than CCP or PassPoints due to viewport influence	Can be hashed, but grid identifier and images must be available to system	One login	Man-in-the-middle to retrieve images, one login to repeat	Possible with complex description of each image or screen shots	Screen or Mouse

Table 8.3: Usability comparison of CCP and PCCP. The reported times represent the mean values in seconds.

Scheme	Type of Memory	Time to Create Pswd	Time to Login	Login Success Rate	Number of Images Needed	Types of User Studies
L. Cued Click-Points (CCP)	Cued recall (one-to-one)	24.7 sec (click-time)	7.4 sec (click-time)	96%	Per user: Minimum 433 images, with 451 × 331 image and 19×19 squares (Section 8.2)	Lab
M. Persuasive Cued Click-Points (CCP)	Cued recall (one-to-one)	36.3 sec (click-time)	10.6 sec (click-time)	91%	Per user: Minimum 433 images, with 451 × 331 image and 19×19 squares (Section 8.2)	Lab

Chapter 9

Design Strategies and Conclusion

To conclude this thesis, we look at design strategies derived from our work with click-based graphical passwords. We believe that these can help inform the design of other knowledge-based authentication schemes and may also be applicable to other types of usable authentication interfaces. We next summarize our research contributions and show how these met the objectives set forth in this thesis. In closing, we discuss research directions based on this work, and offer concluding remarks.

9.1 Design Strategies

Graphical passwords are not necessarily the best approach to authentication in all cases, but we find that they offer an excellent environment for exploring the effects of user interface design decisions and techniques for helping users select better passwords, since it is relatively easy to compare user choices. In this section, we step back and address the larger issue of design in knowledge-based authentication systems.

We have applied the following four design strategies to click-based graphical passwords. We believe that these are the main contributing factors for the enhanced security, memorability, and usability of our proposed graphical password schemes. Throughout this thesis, we have shown that improving usability leads to improved security because when the system is easier to use and there is less of a memory burden placed on users, then they are less likely to resort to unsafe coping strategies such as selecting weak, predictable passwords.

We further believe that some of the underlying design characteristics (described below) included in CCP, PCCP, and centered discretization could be generalized for application to other knowledge-based authentication mechanisms. Our recommendation is that new knowledge-based authentication schemes include analogous features based on the following design strategies to increase usability and security.

9.1.1 One-to-one cueing

> **Design Strategy 1:** *Knowledge-based authentication schemes should include one-to-one cueing to help with the memorability of passwords, and to make it possible for users to remember less predictable passwords.*

Psychology research has shown that cued-recall is an easier memory task than recall alone. Tulving and Pearlstone [120] discuss the possibility that items in human memory may be available but not accessible for retrieval. They show that information that was previously inaccessible in a pure recall situation can be retrieved with the aid of a retrieval cue. They further show that performance in the cued-recall condition is inversely related to the number of items associated with one cue. In other words, one-to-one cued-recall was an easier memory retrieval task than cued-recall where multiple items were associated with the one cue.

All three click-based graphical password schemes examined in this thesis used cueing to help users remember their password. However, CCP and PCCP have an advantage over PassPoints. One-to-one cueing increases the security of passwords by facilitating password choices that are less predictable. We found that CCP and PCCP users created passwords that were less likely to follow predictable click-point patterns than users of PassPoints; this is a result that enlarges the effective password space for CCP and PCCP. Memorability was not affected by this increase in security; login success rates were equally high with CCP and PCCP as they were with PassPoints. Furthermore, of the users who tried PassPoints and CCP, most said that they appreciated and preferred the one-to-one cueing offered by CCP. We believe the reason is that with one-to-one cueing, users did not need to resort to fabricated memory aids such as selecting their click-points in a predictable geometric patterns because the cues provided by the system were sufficient for retrieving the memory of the password.

The idea behind *one-to-one cueing* for knowledge-based authentication is to provide a memory cue for each component of a user's password. We believe that one-to-one cueing could be incorporated into other knowledge-based recall authentication systems with similar benefits: increased memorability which indirectly leads to increased security. One example of a text-based password system that successfully uses

(nearly) one-to-one cueing is Inkblot Authentication [113], discussed in Section 2.4.4. In this system, each inkblot acts as a cue for two text characters. Future work by members of our research group (discussed in Section 9.3) includes designing alternative types of one-to-one cueing for text passwords.

In adding one-to-one cueing to other knowledge-based authentication systems, designers must be careful to balance the security gained from less predictable passwords with the potential information revealed through the use of cues that may also be accessible to attackers. Ideally, carefully designed cueing systems would provide no meaningful information to those with no previous knowledge of the password.

9.1.2 Implicit feedback

> **Design Strategy 2:** *Knowledge-based authentication schemes should provide implicit feedback to users — feedback that is meaningful only to a legitimate user of a system.*

A common difficulty in designing usable security interfaces is that many of the established HCI design principles cannot be directly applied. Providing clear and meaningful feedback is widely accepted and important design principle [75, 109] in user interface design. As explained by Molich and Nielsen [75], "The system should always keep the user informed about what is going on by providing him or her with appropriate feedback within reasonable time."

Offering feedback in security interfaces is often problematic, however, because the feedback may also provide valuable information to attackers. In some cases this is unavoidable and necessary, such when the system accepts or rejects a login attempt; this explicitly tells both a legitimate user and an attacker if this is the correct password for this particular account. In other instances, however, additional feedback would be helpful for usability. For example, it would be useful if a system told a user how many characters were incorrect after a failed login attempt, or even immediately informed the user of an incorrect character as the password is being typed. This type of feedback would obviously be much more meaningful and timely to a legitimate user who simply mistyped a password than the typical "login failed" error message displayed in current systems. However, password systems that provide such feedback

would make it significantly easier for an attacker to determine the correct password for a given account.

The idea behind *implicit feedback* is to provide feedback that has meaning only to the legitimate user of the system. Ideally, the feedback in password systems should not provide any meaningful information to anyone who does not have previous knowledge of the password. Others may see the feedback, but unless they already know the secret, the feedback will not help them to uncover the password. The interim feedback should not explicitly reveal "right" or "wrong", but instead provide information that requires interpretation only possible with previous knowledge of the password.

We accomplish this in CCP and PCCP by showing the sequence of images as the user logs in. As each correct image appears, the user receives feedback that the password is entered correctly up to this point. If an unknown image appears, legitimate users should immediately realize that the last click-point entered was incorrect. Users should further recognize the error as they will not have a correct click-point to enter on this incorrect image; however, if they do enter a click-point, the next image will also be incorrect, and, as such, serve as additional notice that an error has been made during password entry. The usual "login failed" mechanism is still in place if users reach the end of the password with incorrectly entered click-points.

The timing of the implicit feedback, as implemented in our schemes, is especially useful for users. Users do not have to wait until they have entered the entire password before receiving feedback. They are also implicitly informed at which stage an error occurred, avoiding situations that occur with traditional text passwords where users repeatedly enter the same incorrect password because they assume that the error was a simple typing mistake when in fact the entire password is incorrect. When the password is correctly entered, users receive continuous positive feedback (in the form of correct images) as they progress through password entry.

We are not aware of any other knowledge-based authentication scheme that utilizes implicit feedback. We believe, however, that with careful design it should be possible to include it into other schemes and that this would be advantageous.

9.1.3 Safe-path-of-least-resistance

> **Design Strategy 3:** *Knowledge-based authentication schemes should encourage and influence users to select more secure passwords by making it easier to make a secure choice than an insecure choice.*

Persuasive Technology [42] uses technology to intentionally guide, motivate, or influence users to behave in a desired manner. We use Persuasive Technology to design the user interface such that the easiest way to accomplish a task is also the path we want the user to take. Two Persuasive Technology principles are especially relevant to our current work. The principle of *reduction* makes "target behaviors easier by reducing a complex activity to a few simple steps" [42]. The principle of *tunneling* uses technology to "guide users through a process or experience" [42]. We combine these two ideas to form our own design strategy for security interfaces: the "safe-path-of-least-resistance" shows users what the secure behaviour entails, and makes it easier for them to perform the secure task than to perform an insecure one.

In security, it is often assumed that the secure path will impose at least some additional burden on users, but that it is worth the additional effort due to the increased security it provides. For example, crafting a long text password that appears random yet is still somehow meaningful and memorable is a difficult task, but it is seen as worthwhile (at least by system designers and administrators), especially for important accounts. We believe that it is possible to influence users towards secure behaviour without additional effort on the part of users. This strategy has also been suggested by Yee [143] as a general approach for usable security, and by Dhamija and Dusseault [31] with respect to increasing adoption of identity management systems, although neither discuss the persuasive aspects of the technique.

With PCCP, we influence users to select more random passwords by making this task easier, less time-consuming, and less tedious than selecting insecure, predictable passwords. In the process, we also hope that users learn that choosing click-points from random locations on the images is a good strategy for password selection. We do not preclude users from selecting insecure passwords; they are free to expend the additional effort necessary to create a weaker password, but this comes at additional cost to the user. This design offers a balance between allowing user choice so that

a memorable password can be selected, and increasing security by suggesting more secure options that may not have been otherwise considered by the user.

In more recent work [45, 46], summarized in Section 9.2, we have pursued creating a text password system that uses the safe-path-of-least-resistance to influence users to create more secure text passwords. We believe that the safe-path-of-least-resistance approach can be a useful design strategy for encouraging memorable and secure password selection.

9.1.4 Matching user expectations

> **Design Strategy 4:** *Knowledge-based authentication systems should perform in a manner that matches user expectations of the systems.*

Users form mental models, or internal representations, of the external world, including the objects with which they interact. The mental models are used to interpret and predict interaction with these objects [83]. There are many diverging theories explaining the exact cognitive processes involved in the formation and usage of mental models [111, 132], but there is general consensus on their importance in the design of user interfaces [83, 89, 107].

When users have incorrect or incomplete mental models of a computer system, they are often ill-equipped to deal with problems that arise [107]. With security systems, we see two types mental model problems that can occur. First, users misunderstand the threats and risks associated with computer security, which may lead them to take actions that are less secure than would be the case with a proper understanding. Secondly, when users misunderstand the security mechanism itself, they are often more likely to misuse the system. Users may also be more likely to mistrust or bypass a security system if it behaves in unexpected ways. The importance of having accurate mental models of security systems has been discussed previously in our work with password managers [20] and by other researchers [39, 134, 144].

In our work with discretization of click-based graphical passwords (see Chapter 6), we found a negative impact from having system behaviour that does not reflect users' expectations of the system: we saw that a high percentage of login attempts would have been falsely rejected by a system that utilized robust discretization because the

tolerance regions around click-points were not positioned as users expected. Such high false reject rates may lead to frequent password resets as users doubt their memory of the password, and may lead users to avoid using the system entirely. Users may not be able to differentiate between user errors on their part and peculiarities of the system. These frustrations further increase the burden imposed on users from having to use security mechanisms; not only must users remember and enter passwords, but they must also deal with unexpected system behaviour. With centered discretization, we show that with careful consideration of the usability implications of system implementation, it is possible to reduce the potential for user frustration.

9.2 Research Contributions

The general research topic addressed in this thesis was whether the memorability of passwords could be increased while maintaining or also increasing security. Our specific research question was "Can click-based graphical passwords simultaneously support both memorability and security, while maintaining usability?". We defined four main objectives, summarized below.

Objective 1: Catalogue existing graphical password schemes, focusing equally on usability and security characteristics, and identify the existing graphical password scheme that appears most promising and that warrants closer evaluation.

Objective 2: With respect to security and usability, empirically evaluate the most promising scheme identified through our cataloguing. (This turned out to be the PassPoints scheme.)

Objective 3: Create and empirically test new designs that address any usability and security problems identified in the scheme identified in Objective 2. (Given that PassPoints was the identified scheme, the resulting goal ended up being to increase security and memorability of click-based graphical passwords while maintaining usability.)

Objective 4: Identify the key underlying design characteristics responsible for success of the newly proposed system(s), and generalize these to develop design

strategies that can be applied to other types of knowledge-based authentication schemes.

We first present how our primary research contributions address the objectives set forth in this thesis. We then highlight some notable minor contributions. These contributions advance knowledge in the field of usable security through novel knowledge-based authentication schemes, empirical studies evaluating usability and security, and examination of how usability and security affect each other.

9.2.1 Main contributions

To meet the first objective, we reviewed existing graphical password schemes by cataloguing them according to several usability and security characteristics. While we uncovered a wide variety of approaches to graphical passwords, we discovered that there was little consistency in how these systems were presented or evaluated. We also found that many were not thoroughly assessed from both usability and security perspectives. To our knowledge, the most recent surveys of graphical passwords in the peer-reviewed literature were published in 2005 [77, 115]; our work provides a more comprehensive summary and includes recent work in the area.

To address the second objective, we conducted usability and security analysis of PassPoints. We initially carried out a user study in the lab, followed by a large field study where students from three classes used PassPoints to access online material for approximately two months. In our initial analysis, we show that image choice impacts the usability of PassPoints, that users are extremely accurate in entering their click-points, and that login times and success rates are generally good. In later analysis of the PassPoints datasets, we show that users select passwords that form simple geometric patterns and that the click-point distributions have significant amounts of clustering. Both of these results indicate that attackers may be able to predict passwords with higher likelihood of being chosen, and then use this information to launch efficient dictionary attacks.

The third objective was met by designing, prototyping, and testing two novel click-based graphical password schemes: Cued Click-Points (CCP) and Persuasive Cued Click-Points (PCCP). These were intended to further increase memorability and

usability, as well as increase security when compared to PassPoints. We conducted a lab study of each scheme, showing a significant improvement in the randomness of user chosen passwords. Indeed, both showed remarkable decrease in occurrence of geometric patterns, and PCCP additionally showed significant decrease in click-point clustering. On the measures we used to evaluate patterns and clustering, the PCCP dataset was similar to the randomly generated datasets. The two schemes have additional security benefits, due to the large number of images that attackers would need to discover, collect, and analyze in order to launch successful guessing attacks. The same characteristics that render CCP and PCCP apparently more secure also make them more usable. The use of implicit feedback helps users recognize when, and at which stage, they made a mistake during login. One-to-one cueing helps with memorability of the passwords, as evidenced by the high login success rates and quick password entry times, even though the passwords were more resistant to the attacks considered than PassPoints.

As part of meeting the third objective, we also created centered discretization, a new method for the discretization of click-based graphical passwords. Centered discretization ensures a uniform tolerance area around a click-point; this is a feature that we believe is a major improvement over robust discretization. In our post-hoc analysis, we compare centered discretization and robust discretization. Our results show that centered discretization eliminates what we define as false positives and false negatives that occur with robust discretization. Our algorithm allows for smaller tolerance areas, which increases the theoretical password space, and better usability because the system behaves in a manner consistent with user expectations.

To meet the fourth objective, we identify what we believe are the precise mechanisms that lead to increased usability and security in CCP and PCCP. We believe that these four design strategies can be generalized and are applicable to other knowledge-based authentication schemes. The concept of implicit feedback addresses an important issue in usable security: the need to provide feedback to users without also helping attackers. Implicit feedback provides feedback that is only meaningful to users who already have knowledge of the correct password; the same feedback reveals nothing to those who are do not know the password. In one-to-one cueing,

the system offers a cue to help users remember each component of their password. Each cue helps to trigger the specific memory associated with that cue. Our third design principle uses concepts from Persuasive Technology to encourage users to select less predictable passwords by making this behaviour the safe-path-of-least-resistance. The last design principle addresses the issue of matching the user's expectations of system behaviour and discusses how a disconnect between system performance and a user's mental model can lead to usability and security problems.

9.2.2 Minor contributions

This research also produced several minor contributions. Although these were not directly mandated by our objectives, they provide advancement in the area of graphical password research.

With our empirical studies of PassPoints, we provide evidence confirming some of the usability results first reported by the original PassPoints authors [135–137]. We also provide evidence contradicting some of the earlier findings. Our results suggest better usability than initially thought with respect to accuracy in targeting click-points; this property could be harnessed to increase the theoretical password space. We also clarify that the prototype system used by Wiedenbeck et al. [135–137] for PassPoints did not implement robust discretization. Their systems instead used a centered tolerance approach to verifying click-points, which means that their results do not take into account the variations in system behaviour that could have impacted usability. In Chapter 6, we show that robust discretization would have reduced the reported usability by significantly increasing the number of falsely rejected login attempts.

Our field study of PassPoints provides the first look at the memorability effects of multiple password interference. We found that users having two passwords (one for each of two accounts) had lower login success rates than those who only had to remember one password. This raised further questions about whether memorability was better for graphical passwords than text passwords when multiple passwords needed to be remembered.

One observation of our cataloguing efforts was that there is lack of consistency in

user studies conducted to evaluate graphical password schemes. As a result, it is very difficult to get an accurate comparison of the usability and security of the different schemes. In our work, we have described in detail and used the same methodology and evaluation criteria for all three schemes that we evaluated, allowing for more precise comparison.

In our analysis of user choice in password selection, we introduced and utilized point pattern analysis from spatial statistics to determine and compare the clustering in point patterns that arise in graphical passwords. This approach is typically used in earth sciences and biology. These methods allow for statistical comparison of click-points from each study, and for comparison with randomly generated click-point data that simulates the click-point distribution found in the theoretical password space.

9.3 Research Directions

This thesis has contributed to usable security literature, but it has also raised further questions. In this section, we describe other projects resulting from this thesis. Members of our research group are currently working on some of these projects, while other projects have yet to be undertaken.

Field study of PCCP. PCCP has proven successful in a lab setting, and the next logical step is to conduct a field study evaluating its performance in the real-world. We suggest that this study could most easily be conducted in a manner similar to the PassPoints field study described in this thesis. Such a study would provide a large dataset of click-points for passwords that were used in practice. It would make it possible to examine whether the memorability of PCCP passwords remains high over time and whether the persuasive elements work equally well when users are selecting real passwords.

Attacker study on centered discretization. Reviewers of centered discretization have worried that if attackers gain access to both the grid identifiers used for the click-points of a user's password and the image set for this particular user, attackers may be able to use this information to help predict likely passwords. In this offline attack, the argument is that since the attackers would know that the click-point is necessarily at the center of one of the grid squares, attackers might be able to overlay

the grid onto the image and pinpoint which click-points are most likely.

It remains unclear whether this type of attack would be any more effective than if robust discretization was used, revealing a centered area inside the grid square. Users are unlikely to select "clickable points" on the image that are only one pixel in size and they may not select the exact center of these larger objects as their click-point. To address the issue, however, an empirical study could be conducted. Participants would act as "attackers" who have access to the images and the grids with either the center pixel (centered discretization) or the center area (robust discretization) highlighted. We would then ask them to select the click-points that they believe are most likely to be part of the user's password. Since we already have passwords collected from real users for PassPoints, CCP, and PCCP, these could be used as realistic targets for the attack. The analysis would compare the number of successfully guessed click-points (or passwords) for centered discretization as opposed to robust discretization.

Multiple password interference. Our field study of PassPoints revealed that users who had multiple PassPoints passwords (on different images) had more difficulty remembering their passwords than those users who had only one to remember. We are aware of no study of password interference for text passwords, so it was difficult to gauge the severity of this problem. We have recently completed a lab study comparing the memorability of multiple graphical passwords to the memorability of multiple text passwords.

In our lab study, currently available as a tech report [18], 36 users created 6 distinct passwords, one for each of 6 fictitious accounts (bank, email, instant messenger, library, online dating, and work). The accounts were identified by coloured banners at the top of the application window that included a unique icon and the account name. Users created either 6 text passwords or 6 PassPoints passwords. Later in the session, users had to recall these passwords and log in to each account, in shuffled order. We found that participants in the graphical password condition coped significantly better than those in the text password condition. In particular, they made fewer errors when recalling their passwords, did not resort to creating passwords directly related to account names, and did not use similar passwords across multiple accounts. We suggest that this is due to memory cues offered by graphical passwords which help

users to recall their passwords without resorting to insecure coping strategies.

Further work includes testing CCP and PCCP's performance under the same conditions as was tested for PassPoints to see whether they offer even further memorability benefits. If results of this second lab study are positive, then the long-term memorability of multiple passwords should also be investigated through a field study.

Varying parameters to enlarge the theoretical password space. Another study currently in the planning stages investigates whether users perform equally well when the system parameters are modified to enlarge the theoretical password space. This work is being conducted primarily by Elizabeth Stobert, an honours student from the psychology department who is a member of our group. Her user study will look at variations such as increasing the number of click-points and increasing the size of the background image for PCCP.

Text Passwords. We are also looking at the applicability of our design strategies to text passwords. This project is joint work with Alain Forget, who is the primary researcher. Persuasive Text Passwords (PTP) [43,45,46] employ persuasive strategies similar to PCCP's viewport to encourage the creation of more secure passwords. After users choose a text password, the system strengthens the password by inserting random alphanumeric characters within the password; users may shuffle for different characters if they are unhappy with the current selection, but the user's password ultimately includes the initial password plus randomly inserted characters. Results show that this is an effective strategy to increase the security of text passwords, but that there appears to be an upper limit in the amount of randomness that users are able and willing to memorize. Future work includes investigating how one-to-one cueing, implicit feedback, and other persuasive strategies can also be incorporated into text passwords to increase security, memorability, and usability.

General Design Principles for Usable Knowledge-based Authentication. Finally, we believe that the work presented in this thesis could be expanded to form a set of general design principles for usable authentication. Different design guidelines and approaches have been proposed, but these have yet to be unified. The earliest design guidelines for usable security were proposed by Whitten and Tygar [134]. We proposed an extension to those guidelines based on our work with

password managers [20]. Yee [143, 145] proposed preliminary guidelines for secure interaction design and guidelines aimed at designing systems that perform according to users' intentions. Recently, Cranor [24] proposed a framework, built on the C-HIP model from warnings science, to systematically identify potential causes of human failure in security systems. Others have proposed models for specific areas of security, such as Dourish and Remiles's approach [36] to helping users build better mental models of system security through visualizations, and Dhamija and Dusseault [31]'s recommendation for identity management systems.

Similarly, there is no existing general set of design principles for usable knowledge-based authentication. This set of design principles would need to address issues such as balancing memorability and password strength. Although not a comprehensive set, the four design strategies identified in this thesis may contribute to a general set of design principles. Both cueing and implicit feedback help with memorability, while the safe-path-of-least-resistance assists in creating stronger passwords. Matching user expectations addresses some common usability problems. Future work should include establishing a set of general design principles for usable authentication.

9.4 Conclusion

Our general goal in this thesis was to increase the memorability and security of knowledge-based authentication schemes. We focused on click-based graphical passwords. We were successful at designing innovative schemes that improved memorability and that were more secure than existing alternatives. From this empirical work, we identified the key features of our designs and derived design strategies that we believe are applicable to other knowledge-based authentication schemes.

The relationship between usability and security is a complex one; too often, improvements in one lead to a reduction in the other. As we have shown, it is possible to increase both simultaneously through careful design that considers usability and security in combination. We emphasize the need for thorough usability and security evaluations because system design can significantly impact user behaviour, sometimes in unanticipated ways, which in turn can significantly impact the security of a system.

Bibliography

[1] A. Adams and M. Sasse. Users are not the enemy. *Communication of the ACM*, 42(12):41–46, 1999.

[2] F. Alsulaiman and A. El Saddik. A novel 3D graphical password schema. In *IEEE International Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems*, July 2006.

[3] American Psychological Association. *Publication Manual of the American Psychological Association.* American Psychological Association (APA), 5th edition edition, 2001.

[4] J. Anderson and G. Bower. Recognition and retrieval processes in free recall. *Psychological Review*, 79(2):97–123, March 1972.

[5] R. Anderson. Why cryptosystems fail. In *1st ACM Conference on Computer and Communications Security*, December 1993.

[6] D. Andrews, B. Nonnecke, and J. Preece. Electronic survey methodology: A case study in reaching hard-to-involve Internet users. *International Journal of Human-Computer Interaction, Lawrence Erlbaum Associates*, 16(2):185–210, 2003.

[7] M. Backes, M. Durmuth, and D. Unruh. Compromising reflections — or — how to read LCD monitors around the corner. In *IEEE Symposium on Security and Privacy*, 2008.

[8] A. Baddeley and R. Turner. R. spatstat: An R package for analyzing spatial point patterns. *Journal of Statistical Software*, 12(6):1–42, 2005.

[9] J. Birget, D. Hong, and N. Memon. Graphical passwords based on robust discretization. *IEEE Transactions on Information Forensics and Security*, 1(3):395–399, 2006.

[10] G. Blonder. Graphical passwords. United States Patent 5,559,961, 1996.

[11] I. Britton. Freefoto website. http://www.freefoto, accessed February 2007.

[12] A. Brodskiy. Personal communication, September 3 2006.

[13] S. Brostoff and M. Sasse. Are Passfaces more usable than passwords? A field trial investigation. In *British Human-Computer Interaction Conference (HCI)*, September 2000.

[14] S. Chakrabarti and M. Singhal. Password-based authentication: Preventing dictionary attacks. *Computer, IEEE Computer Society*, 40(6):68–74, June 2007.

[15] S. Chiasson, R. Biddle, and P. van Oorschot. A second look at the usability of click-based graphical passwords. In *3rd Symposium on Usable Privacy and Security (SOUPS)*, July 2007.

[16] S. Chiasson, A. Forget, R. Biddle, and P. van Oorschot. Influencing users towards better passwords: Persuasive Cued Click-Points. In *Human Computer Interaction (HCI), The British Computer Society*, September 2008.

[17] S. Chiasson, A. Forget, R. Biddle, and P. van Oorschot. User interface design affects security: Patterns in click-based graphical passwords (Manuscript under submission). Technical Report TR-08-14, School of Computer Science, Carleton University, 2008.

[18] S. Chiasson, A. Forget, E. Stobert, P. van Oorschot, and R. Biddle. Multiple password interference in text and click-based graphical passwords. (Manuscript under submission). Technical Report TR-08-20, School of Computer Science, Carleton University, September 2008.

[19] S. Chiasson, J. Srinivasan, R. Biddle, and P. van Oorschot. Centered discretization with application to graphical passwords. In *USENIX Usability, Psychology, and Security (UPSEC)*, April 2008.

[20] S. Chiasson, P. van Oorschot, and R. Biddle. A usability study and critique of two password managers. In *15th USENIX Security Symposium*, August 2006.

[21] S. Chiasson, P. van Oorschot, and R. Biddle. Graphical password authentication using Cued Click Points. In *European Symposium On Research In Computer Security (ESORICS), LNCS 4734*, pages 359–374, September 2007.

[22] L. Coventry. Usable biometrics. In L. Cranor and S. Garfinkel, editors, *Security and Usability: Designing Secure Systems That People Can Use*, chapter 10, pages 175–197. O'Reilly Media, 2005.

[23] F. Craik and J. McDowd. Age differences in recall and recognition. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 13(3):474–479, July 1987.

[24] L. Cranor. A framework for reasonning about the human in the loop. In *USENIX Usability, Psychology, and Security (UPSEC)*, April 2008.

[25] L. Cranor and S. Garfinkel. *Security and Usability: Designing Systems that People Can Use.* O'Reilly Media, edited collection edition, 2005.

[26] D. Davis. Compliance defects in public key cryptography. In *6th USENIX Security Symposium*, July 1996.

[27] D. Davis, F. Monrose, and M. Reiter. On user choice in graphical password schemes. In *13th USENIX Security Symposium*, August 2004.

[28] 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):128–152, 2005.

[29] D. Denning and P. MacDoran. Location-Based Authentication: Grounding cyberspace for better security. *Computer Fraud & Security, Elsevier Science Ltd.*, February 1996.

[30] S. Designer. John the Ripper password cracker. http://www.openwall.com/john/.

[31] R. Dhamija and L. Dusseault. The seven flaws of identity management: Usability and security challenges. *IEEE Security & Privacy*, pages 24–29, March/April 2008.

[32] R. Dhamija and A. Perrig. Déjà Vu: A user study using images for authentication. In *9th USENIX Security Symposium*, 2000.

[33] R. Dhamija, J. Tygar, and M. Hearst. Why phishing works. In *ACM Conference on Human Factors in Computing Systems (CHI)*, April 2006.

[34] P. Diggle. *Statistical Analysis of Spatial Point Patterns.* Academic Press: New York, NY, 1983.

[35] A. Dirik, N. Menon, and J. Birget. Modeling user choice in the Passpoints graphical password scheme. In *3rd ACM Conference on Symposium on Usable Privacy and Security (SOUPS)*, July 2007.

[36] P. Dourish and D. Redmiles. An approach to usable security based on event monitoring and visualization. In *New Security Paradigms Workshop (NSPW)*, September 2002.

[37] P. Dunphy, J. Nicholson, and P. Olivier. Securing Passfaces for description. In *4th Symposium on Usable Privacy and Security (SOUPS)*, July 2008.

[38] P. Dunphy and J. Yan. Do background images improve "Draw a Secret" graphical passwords? In *14th ACM Conference on Computer and Communications Security (CCS)*, October 2007.

[39] J. C. F. Asgharpour, D. Liu. Mental models of security risks. In *Financial Cryptography and Data Security, LNCS, Springer*, 2007.

[40] L. Faulkner. Beyond the five-user assumption: Benefits of increased sample sizes in usability testing. *Behavior Research Methods, Instruments, & Computers*, 35(3):379–383, 2003.

[41] D. Florencio and C. Herley. A large-scale study of WWW password habits. In *16th ACM International World Wide Web Conference (WWW)*, May 2007.

[42] B. Fogg. *Persuasive Technologies: Using Computers to Change What We Think and Do.* Morgan Kaufmann Publishers, San Francisco, CA, 2003.

[43] A. Forget and R. Biddle. Memorability of Persuasive Passwords (poster). In *ACM SIGCHI Student Research Competition*, April 2008.

[44] A. Forget, S. Chiasson, R. Biddle, and P. van Oorschot. Persuasion as education for computer security. In *AACE E-Learn Conference*, October 2007.

[45] A. Forget, S. Chiasson, P. van Oorschot, and R. Biddle. Improving text passwords through persuasion. In *4th Symposium on Usable Privacy and Security (SOUPS)*, July 2008.

[46] A. Forget, S. Chiasson, P. van Oorschot, and R. Biddle. Persuasion for stronger passwords: Motivation and pilot study. In *3rd International Conference on Persuasive Technology*, June 2008.

[47] J. Goldberg, J. Hagman, and V. Sazawal. Doodling our way to better authentication (student poster). In *ACM Conference on Human Factors in Computing Systems (CHI)*, April 2002.

[48] E. Goldstein. *Cognitive Psychology.* Wadsworth Publishing, 2006.

[49] P. Golle and D. Wagner. Cryptanalysis of a cognitive authentication scheme (extended abstract). In *IEEE Symposium on Security and Privacy*, May 2007.

[50] K. Golofit. Click passwords under investigation. In *12th European Symposium On Research In Computer Security (ESORICS), LNCS 4734*, September 2007.

[51] L. Gong, M. Lomas, R. Needham, and J. Saltzer. Protecting poorly chosen secrets from guessing attacks. *IEEE Journal on Selected Areas in Communications*, 11(5):648–656, June 1993.

[52] N. Govindarajulu and S. Madhvanath. Password management using doodles. In *9th International Conference on Multimodal Interfaces (ICMI)*, November 2007.

[53] J. Halderman, B. Waters, and E. Felten. A convenient method for securely managing passwords. In *14th International World Wide Web Conference (WWW)*, 2005.

[54] E. Hayashi, N. Christin, R. Dhamija, and A. Perrig. Use Your Illusion: Secure authentication usable anywhere. In *4th ACM Conference on Symposium on Usable Privacy and Security (SOUPS)*, Pittsburgh, July 2008.

[55] G. Heiman. *Basic Statistics for the Behavioral Sciences.* Houghton Mifflin Company: Boston, MA, 1992.

[56] T. Hewett, R. Baecker, S. Card, T. Carey, J. Gasen, M. Mantei, G. Perlman, G. Strong, and W. Verplank. ACM SIGCHI Curricula for Human-Computer Interaction. http://www.sigchi.org/cdg/index.html, 1996.

[57] A. Hollingworth and J. Henderson. Accurate visual memory for previously attended objects in natural scenes. *Journal of Experimental Psychology: Human Perception and Performance*, 28(1):113–136, 2002.

[58] R. Ihaka and R. Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.

[59] A. Jain, L. Hong, and S. Pankanti. Biometric identification. *Communication of the ACM*, 43(2):91–98, February 2000.

[60] M. Jakobsson and S. Myers, editors. *Phishing and Countermeasures: Understanding the Increasing Problem of Electronic Identity Theft.* Wiley-Interscience, 2006.

[61] I. Jermyn, A. Mayer, F. Monrose, M. Reiter, and A. Rubin. The design and analysis of graphical passwords. In *8th USENIX Security Symposium*, August 1999.

[62] C. Kaufman, R. Perlman, and M. Speciner. *Network Security: PRIVATE Communication in a PUBLIC World.* Prentice Hall, 2nd edition edition, 2002.

[63] 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):17–28, 2007.

[64] 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: New York, 1970.

[65] B. Kirkpatrick. An experimental study of memory. *Psychological Review*, 1:602–609, 1894.

[66] D. Klein. Foiling the cracker: A survey of, and improvements to, password security. In *2nd USENIX Security Workshop*, 1990.

[67] S. Komanduri and D. Hutchings. Order and entropy in Picture Passwords. In *Graphics Interface Conference (GI)*, May 2008.

[68] M. Kumar, T. Garfinkel, D. Boneh, and T. Winograd. Reducing shoulder-surfing by using gaze-based password entry. In *3nd ACM Conference on Symposium on Usable Privacy and Security (SOUPS)*, July 2007.

[69] C. Kuo, S. Romanosky, and L. Cranor. Human selection of Mnemonic Phrase-based Passwords. In *2nd ACM Conference on Symposium on Usable Privacy and Security (SOUPS)*, July 2006.

[70] B. Laxton, K. Wang, and S. Savage. Reconsidering physical key secrecy: Teleduplication via optical decoding. In *15th ACM conference on Computer and communications security*, 2008.

[71] S. MacKenzie and W. Buxton. Extending Fitts' Law to two-dimensional tasks. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 1992.

[72] S. Madigan. Chapter 3: Picture memory. In J. Yuille, editor, *Imagery, Memory, and Cognition: Essays in Honor of Allan Paivio*, chapter 3. Picture Memory, pages 65–89. Lawrence Erlbaum Associates, 1983.

[73] Merriam-Webster. Merrian-Webster website. http://www.merriam-webster.com/help/faq/total_words.htm, October 2008.

[74] K. Mitnick and W. Simon. *The Art of Deception: Controlling the Human Element of Security.* New York: John Wiley & Sons., 2002.

[75] R. Molich and J. Nielsen. Improving a human-computer dialogue. *Communication of the ACM*, 33(3):338–348, March 1990.

[76] W. Moncur and G. Leplatre. Pictures at the ATM: Exploring the usability of multiple graphical passwords. In *ACM Conference on Human Factors in Computing Systems (CHI)*, April 2007.

[77] F. Monrose and M. Reiter. Graphical passwords. In L. Cranor and S. Garfinkel, editors, *Security and Usability: Designing Secure Systems That People Can Use*, chapter Chapter 9, pages 157–174. OReilley, 2005.

[78] D. Nali and J. Thorpe. Analyzing user choice in graphical passwords. Technical report, TR-04-01, School of Computer Science, Carleton University, May 2004.

[79] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, March 2001.

[80] D. Nelson, V. Reed, and J. Walling. Pictorial Superiority Effect. *Journal of Experimental Psychology: Human Learning and Memory*, 2(5):523–528, 1976.

[81] J. Nielsen. *Usability Engineering.* Boston: AP Professional, 1993.

[82] J. Nielsen and R. Mack. *Usability Inspection Methods.* John Wiley & Sons, Inc, 1994.

[83] D. Norman. *The Design of Everyday Things.* Basic Books, 1988.

[84] M. Orozco, B. Malek, M. Eid, and A. El Saddik. Haptic-based sensible graphical password. In *Proceedings of Virtual Concept*, December 2006.

[85] A. Paivio. *Mind and its evolution: a dual coding theoretical approach.* Lawrence Erlbaum: Mahwah, N.J., 2006.

[86] A. Paivio, T. Rogers, and P. Smythe. Why are pictures easier to recall than words? *Psychonomic Science*, 11(4):137–138, 1968.

[87] Passfaces Corporation. The science behind Passfaces. White paper, http://www.passfaces.com/enterprise/resources/white_papers.htm.

[88] Passlogix. Passlogix website. http://www.passlogix.com.

[89] S. Payne. Chapter 6: User's mental models: The very ideas. In J. Carroll, editor, *HCI Models, Theories, and Frameworks*, chapter Users' Mental Models: The Very Ideas, pages 135–156. Morgan Kaufmann Publishers, San Francisco, CA, 2003.

[90] PD Photo. PD Photo website. http://pdphoto.org, accessed February 2007.

[91] C. Perfetti and L. Landesman. Eight is not enough. *User Interface Engineering*, 2001.

[92] M. Peters. Revised Vandenberg & Kuse Mental Rotations Tests: forms MRT-A to MRT-D. Technical report, Department of Psychology, University of Guelph, 1995.

[93] B. Pinkas and T. Sander. Securing passwords against dictionary attacks. In *9th ACM Conference on Computer and Communications Security (CCS)*, November 2002.

[94] N. Provos, P. Mavrommatis, M. Abu Rajab, and F. Monrose. All your iFrames point to us. In *17th USENIX Security Symposium*, 2008.

[95] D. Ramsbrock, R. Berthier, and M. Cukier. Profiling attacker behavior following SSH compromises. In *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2007.

[96] K. Renaud. Evaluating authentication mechanisms. In L. Cranor and S. Garfinkel, editors, *Security and Usability: Designing Secure Systems That People Can Use*, chapter 6, pages 103–128. O'Reilly Media, 2005.

[97] K. Renaud. On user involvement in production of images used in visual authentication,. *Journal of Visual Language and Computing*, 2008.

[98] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J. Mitchell. Stronger password authentication using browser extensions. In *14th USENIX Security Symposium*, Baltimore, August 2005.

[99] S. Ross. *Unix System Security Tools.* McGraw-Hill, 1999.

[100] V. Roth, K. Richter, and R. Freidinger. A PIN-entry method resiliant against shoulder surfing. In *11th ACM conference on Computer and communications security*, 2004.

[101] A. Salehi-Abari, J. Thorpe, and P. van Oorschot. On purely automated attacks and click-based graphical passwords. In *24th Annual Computer Security Applications Conference (ACSAC)*, 2008.

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

[103] M. Sasse, S. Brostoff, and D. Weirich. Transforming the 'weakest link' – a human/computer interaction approach to usable and effective security. *BT Technology Journal*, 19(3):122–131, July 2001.

[104] M. Sasse and I. Flechais. Usable Security: Why do we need it? How do we get it? In L. Cranor and S. Garfinkel, editors, *Security and Usability: Designing Secure Systems That People Can Use*, chapter 2, pages 13–30. O'Reilly Media, 2005.

[105] C. Seifert. Analyzing malicious SSH login attempts. http://www.securityfocus.com/infocus/1876,, accessed November 2008 2006.

[106] SFR Software. visKey for Pocket PC. http://www.sfr-software.de/cms/EN/pocketpc/viskey/.

[107] H. Sharp, Y. Rogers, and J. Preece. *Interaction Design: Beyond human-computer interaction.* John Wiley & Sons, Inc, 2nd edition edition, 2007.

[108] R. Shepard. Recognition memory for words, sentences, and pictures. *Journal of Verbal Learning and Verbal Behavior*, 6:156–163, 1967.

[109] B. Shneiderman. *Designing the User Interface.* Addison Wesley, 3rd edition, 1998.

[110] J. Spool and W. Schroeder. Testing web sites: Five users is nowhere near enough. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2001.

[111] N. Staggers and A. Norcio. Mental Models: Concepts for human-computer interaction research. *International Journal of Man-Machine Studies*, 38:587–605, 1993.

[112] L. Standing, J. Conezio, and R. Haber. Perception and memory for pictures: Single-trial learning of 2500 visual stimuli. *Psychonomic Science*, 19(2):7374, 1970.

[113] A. Stubblefield and D. Simon. Inkblot Authentication, MSR-TR-2004-85. Technical report, Microsoft Research, Microsoft Corporation, 2004.

[114] X. Suo. A design and analysis of graphical password. Master's thesis, College of Arts and Science, Georgia State University, August 2006.

[115] X. Suo, Y. Zhu, and G. Owen. Graphical passwords: A survey. In *Annual Computer Security Applications Conference (ACSAC)*, December 2005.

[116] H. Tao and C. Adams. Pass-Go: A proposal to improve the usability of graphical passwords. *International Journal of Network Security*, 7(2):273–292, 2008.

[117] F. Tari, A. Ozok, and S. Holden. A comparison of perceived and real shoulder-surfing risks between alphanumeric and graphical passwords. In *2nd ACM Conference on Symposium on Usable Privacy and Security (SOUPS)*, July 2006.

[118] J. Thames, R. Abler, and D. Keeling. A distributed active response architecture for preventing SSH dictionary attacks. In *IEEE Southeastcon*, 2008.

[119] J. Thorpe and P. van Oorschot. Human-seeded attacks and exploiting hot-spots in graphical passwords. In *16th USENIX Security Symposium*, August 2007.

[120] E. Tulving and Z. Pearlstone. Availability versus accessibility of information in memory for words. *Journal of Verbal Learning and Verbal Behavior*, 5:381–391, 1966.

[121] E. Tulving and M. Watkins. Continuity between recall and recognition. *American Journal of Psychology*, 86(4):739–748, 1973.

[122] T. Valentine. An evaluation of the Passface personal authentic system. Technical report, Goldsmiths College University of London, 1998.

[123] M. van Lieshout and A. Baddeley. A nonparametric measure of spatial interaction in point patterns. *Statistica Neerlandica*, 50(3):344–361, 1996.

[124] M. van Lieshout and A. Baddeley. Indices of dependence between types in multivariate point patterns. *Scandanavian Journal of Statistics*, 26(4):511–532, 1999.

[125] P. van Oorschot and S. Stubblebine. On countering online dictionary attacks with login histories and humans-in-the-loop. *ACM Transactions on Information and System Security*, 9(3):235–258, 2006.

[126] P. van Oorschot and J. Thorpe. On predicting and exploiting hot-spots in click-based graphical passwords. Technical report, School of Computer Science, Carleton University, November 2008.

[127] P. van Oorschot and J. Thorpe. On predictive models and user-drawn graphical passwords. *ACM Transactions on Information and System Security*, 10(4):1–33, 2008.

[128] C. Varenhorst. Passdoodles: A lightweight authentication method. Massachusetts Institute of Technology Resarch Science Institute, July 2004.

[129] R. Virzi. Refining the test phase of usability evaluation: How many subjects is enough? *Human Factors*, 34:457–468, 1992.

[130] K.-P. L. 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*, 65:744–757, 2007.

[131] D. Weinshall. Cognitive authentication schemes safe against spyware (short paper). In *IEEE Symposium on Security and Privacy*, May 2006.

[132] L. Westbrook. Mental models: A theoretical overview and preliminary study. *Journal of Information Science*, 32(6):563–579, December 2006.

[133] C. Wharton, J. Bradford, R. Jeffries, and M. Franzke. Applying cognitive walkthroughs to more complex user interfaces: Experiences, issues, and recommendations. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 1992.

[134] A. Whitten and J. Tygar. Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In *8th USENIX Security Symposium*, Washington, D.C., August 1999.

[135] S. Wiedenbeck, J. Waters, J. Birget, A. Brodskiy, and N. Memon. Authentication using graphical passwords: Basic results. In *11th International Conference on Human-Computer Interaction (HCI International)*, July 2005.

[136] 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):102–127, 2005.

[137] S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon. Authentication using graphical passwords: Effects of tolerance and image choice. In *1st Symposium on Usable Privacy and Security (SOUPS)*, July 2005.

[138] 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)*, May 2006.

[139] J. Wolf. Visual Attention. In K. De Valois, editor, *Seeing*, pages 335–386. Academic Press, 2000.

[140] M. Workman. Gaining access with social engineering: An empirical study of the threat. *Information Systems Security, Taylor & Francis Group*, 16(6):315–331, November 2007.

[141] J. Yan, A. Blackwell, R. Anderson, and A. Grant. Password memorability and security: Empirical results. *IEEE Security & Privacy Magazine*, 2(5):25–31, 2004.

[142] J. Yan, A. Blackwell, R. Anderson, and A. Grant. The memorability and security of passwords. In L. Cranor and S. Garfinkel, editors, *Security and Usability: Designing Secure Systems That People Can Use*, chapter 7, pages 129–142. O'Reilly Media, 2005.

[143] K.-P. Yee. User interaction design for secure systems. In *4th International Conference on Information and Communications Security (ICICS), LNCS 2513*, December 2002.

[144] K.-P. Yee. Aligning security and usability. *IEEE Security & Privacy*, 2(5):48–55, Sept-Oct 2004.

[145] K.-P. Yee. Guidelines and strategies for secure interaction design. In L. Cranor and S. Garfinkel, editors, *Security and Usability: Designing Secure Systems That People Can Use*, chapter 13, pages 247–273. OReilley, 2005.

[146] M. Zurko and R. T. Simon. User-centered security. In *New Security Paradigms Workshop (NSPW)*, pages 27–33. ACM, 1996.