

Creating Usable Policies for Stronger Passwords with MTurk

Richard Shay

February 2015
CMU-ISR-15-100

Institute for Software Research
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee

Lorrie Faith Cranor, Advisor
Lujo Bauer
Nicolas Christin
Brian LaMacchia, Microsoft Research

*Submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy in
Computation, Organizations, and Society*

©2015 Richard Shay.

This research was supported in part by NSF grants DGE-0903659, CCF-0424422, and CNS-111676; by CyLab at Carnegie Mellon under grants DAAD19-02-1-0389 and W911NF-09-1-0273 from the Army Research Office; and by a gift from Microsoft Research. The views and conclusions contained in this document are those of the author, and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government, or any other entity.

Keywords: password, policy, usability, security

Abstract

People are living increasingly large swaths of their lives through their online accounts. These accounts are brimming with sensitive data, and they are often protected only by a text password. Attackers can break into service providers and steal the hashed password files that store users' passwords. This lets attackers make a large number of guesses to crack users' passwords. The stronger a password is, the more difficult it is for an attacker to guess.

Many service providers have implemented password-composition policies. These policies constrain or restrict passwords in order to prevent users from creating easily guessed passwords. Too lenient a policy may permit easily cracked passwords, and too strict a policy may encumber users. The ideal password-composition policy balances security and usability.

Prior to the work in this thesis, many password-composition policies were based on heuristics and speculation, rather than scientific analysis. Passwords research often examined passwords constructed under a single uniform policy, or constructed under unknown policies. In this thesis, I contrast the strength and usability of passwords created under different policies. I do this through online, crowdsourced human-subjects studies with randomized, controlled password-composition policies. This result is a scientific comparison of how different password-composition policies affect both password strength and usability.

I studied a range of policies, including those similar to policies found in the wild, policies that trade usability for security by requiring longer passwords, and policies in which passwords are system-assigned with known security. One contribution of this thesis is a tested methodology for collecting passwords under different policies. Another contribution is the comparison between password policies. I find that some password-composition policies make more favorable tradeoffs between security and usability, allowing evidence-based recommendations for service providers. I also offer insights for researchers interested in conducting larger-scale online studies, having collected data from tens of thousands of participants.

This work is dedicated to my amazing parents, Katherine Kalliel and Richard Shay. My parents saw through the sometimes-difficult exterior of my youth to the potential beneath. They worked tirelessly to improve me and to put me into a situation where I could thrive. In my most difficult times, they have always been there for me. I would not be the person I am today if my parents had not invested their time, resources, and love in me. I am eternally grateful.

Acknowledgments

When I began looking at doctoral programs, I was set on staying in New England. Elisa Bertino, with whom I had worked at Purdue, urged me to look into working with Lorrie Cranor at Carnegie Mellon University. She even flew me to a SOUPS conference to meet Lorrie and see her lab's work. Once I met Lorrie, I could see that this was the person I wanted to be my mentor. I am very grateful to Elisa Bertino. Lorrie has been an amazing advisor and mentor. I have watched her work with unbounded enthusiasm and energy. She taught both by what she has said and by her example. It has been a genuine honor to have been her disciple.

In addition, I have the honor of having Nicolas Christin, Lujo Bauer, and Brian LaMacchia on my thesis committee. Their patience, insight, and guidance have been invaluable. I have grown tremendously from their combination of giving me room to pursue my ideas, and pushing me to become better.

I would like to thank all of my co-authors in the research presented in this thesis. Thanks to (alphabetical) Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, Adam Durity, Serge Egelman Alain Forget, Phillip Huh, Patrick Gage Kelley, Saranga Komanduri, Julio Lopez, Michelle Mazurek, William Melicher, Sean M. Segreti, Blase Ur, and Tim Vidas.

Further, thanks to those excellent people who have been my lab-mates and colleagues, beyond those listed above: Rebecca Balebaco, Christian Bravo-Lilo, Hanan Hibshi, Pedro Leon, Abby Marsh, Aleecia McDonald, Greg Norcie, Divya Sharma, Manya Sleeper, Michael Stroucken, and Kami Vaniea. Thanks also to Susan Gentile, from whose example I learned very much. Thanks especially to Saranga Komanduri, who collaborated with me on leading much of the early passwords research. Saranga has been an excellent collaborator and an even better friend.

Before my time at Carnegie Mellon, I have been extremely fortunate to benefit from some amazing mentors. I would like to thank Steve Schatvet, Shelley Tyre, William Earle, Richard Rasala, David Pucci, and Elisa Bertino.

Most of all, thanks to my loving and supportive family. Thanks to my parents Katherine Kalliel and Richard Shay, and my sister Emily Shay. And thanks to my Sitto, Effie Kalliel. Without them, I would not and could not be who I am today.

Contents

1	Introduction	1
1.1	Password Policies, Strength, and Usability	1
1.2	Thesis Overview	2
2	Background and Related Work	5
2.1	Guessing Users' Passwords	5
2.1.1	Online and Offline Attacks	5
2.1.2	Assumptions in our Threat Modeling	6
2.1.3	Real-World Password Breaches	7
2.2	Password-Composition Policies	8
2.2.1	The Limits of Password Policies	9
2.3	Measuring Password Strength	9
2.3.1	Entropy	10
2.3.2	Guessability	10
2.4	Guess Numbers and Threat Modeling	11
2.4.1	Password Hashing with bcrypt	11
2.4.2	Guess Numbers for Specific Attackers	12
2.4.3	Attack Models and Acceptable Loses	13
2.5	Human-Subjects Research on Passwords	14
2.5.1	Passphrases and System-Assigned Passwords	14
2.5.2	Password-Related Surveys	15
2.5.3	Longer-Duration Password Studies	16
2.5.4	Laboratory Studies	16
2.5.5	Interview Studies	17
2.5.6	MTurk Passwords Studies	18
3	Experimental Framework	19
3.1	SHELF Implementation	20
3.2	SHELF Features	21
3.2.1	Monitoring and Organizing Large-Scale Studies	21
3.2.2	Preventing Repeat Participants	22
3.2.3	Keeping Participants from Viewing the Wrong Page	22
3.2.4	Conducting Participants Through the Study	23

4	Study Protocol	25
4.1	Data-Collection Protocol	25
4.2	Generating Guess Numbers	26
4.2.1	PCFG Training Data and Word Lists	27
4.3	Study Metrics	28
4.3.1	Password Strength	28
4.3.2	Password Usability	29
4.4	Statistical Testing	31
4.5	Ethical Considerations	31
4.6	Limitations	31
4.6.1	Ecological Validity	31
4.6.2	Remote Data Collection	32
4.6.3	Other Limitations	33
5	How Usable are System-Assigned Passphrases?	35
5.1	Introduction	35
5.2	Methodology	36
5.2.1	Conditions	37
5.2.2	Password conditions	37
5.2.3	Passphrase conditions	38
5.2.4	Statistical Testing	39
5.3	Results	40
5.3.1	Demographics	40
5.3.2	Study Dropouts	41
5.3.3	Password Storage	41
5.3.4	Assignment	42
5.3.5	Part-One Recall	42
5.3.6	Part-Two Recall	44
5.3.7	User Sentiment	45
5.4	Error Analysis Results	46
5.4.1	Length	46
5.4.2	Ignoring Spaces and Capitalization	46
5.4.3	Off-by-One Errors	47
5.4.4	Closest Dictionary Word Correction	47
5.5	Discussion	47
6	How Secure and Usable are Some Common Password Policies?	55
6.1	Introduction	55
6.2	Methodology	56
6.2.1	Conditions	56
6.3	Demographics	57
6.4	Security Results	58
6.4.1	Comparing Policies for Guessability	58
6.4.2	Guessability and Entropy	59
6.4.3	User Perception of Security	61

6.5	Usability Results	61
6.5.1	Password Creation	62
6.5.2	Part One Recall	63
6.5.3	Part Two Recall	63
6.5.4	Password Storage	64
6.6	Discussion	65
6.6.1	Meeting comp8 Requirements	67
6.6.2	Shortcomings of the basic16 Condition	67
7	Can Longer Passwords be Secure and Usable?	69
7.1	Introduction	69
7.2	Study I	70
7.2.1	Conditions	70
7.2.2	Participants	71
7.2.3	Security Results	71
7.2.4	Usability Results	73
7.2.5	Password Patterns	78
7.3	Using Findings from Study I to Create Study II Conditions	79
7.4	Study II Findings	81
7.4.1	Participants	81
7.4.2	Security Results	81
7.4.3	Usability Results	82
7.4.4	Password Patterns	84
7.5	Discussion	88
7.5.1	Results Summary	88
7.5.2	Comparing 2word16 and 2s-list12	88
7.5.3	Create a Substring Blacklist	89
7.5.4	Recommendations for Service Providers	90
8	Can Creation-Time Feedback Help Users Create Passwords?	91
8.1	Introduction	91
8.2	Research Questions	93
8.3	Methodology	93
8.3.1	Measuring Password Strength	94
8.3.2	Conditions	94
8.3.3	Password Creation in One Step	94
8.3.4	Password Creation in Three Steps	95
8.3.5	Statistical Testing	97
8.4	Results	97
8.4.1	Participants	97
8.4.2	Password Strength	98
8.4.3	User Perception of Password Strength	100
8.4.4	Usability	101
8.5	Discussion	103
8.5.1	Q1: Impact of blacklist and pattern requirements	103

8.5.2	Q2: Impact of branded look-and-feel	104
8.5.3	Q3: Impact of password-creation feedback	104
8.5.4	Q4: Impact of guiding and insertion	104
9	A Reflection on our Collected Data	107
9.1	Factors Correlating with User Sentiment	107
9.1.1	Analysis Results	108
9.1.2	Analysis Discussion	108
9.2	Demographics	111
9.3	Mobile Device Usage	111
9.4	Real Email Passwords	112
10	Practical Recommendations for System Administrators	115
10.1	Password Cracking and What You Can Do About It	115
10.1.1	Offline Attacks	116
10.1.2	Password Hashes	116
10.1.3	Other Factors for Password Security	117
10.2	Password-Composition Policies	117
10.2.1	Password Instructions and Feedback	117
10.2.2	System-Assigned Passwords	117
10.2.3	The Traditional “Strong” Policy	118
10.2.4	Combining Character and Length Requirements	119
10.2.5	Using a Substring Blacklist	119
10.2.6	Improving an Existing Policy	120
10.3	Concluding Remarks	120
11	Conclusion	121
11.1	Data-Collection Protocol	121
11.2	Analyzing Password Policies	121
11.3	Studies of Password-Composition Policies	122
11.4	Understanding Participants and their Sentiment	123
11.5	Concrete Advice for Service Providers	123
11.6	Future Work	123
A	How Usable are System-Assigned Passphrases?	133
A.1	Part One Survey	133
A.2	Part Two Survey	135
B	How Secure and Usable are Some Common Password Policies?	137
B.1	Part One Survey	137
B.2	Part Two Survey	139
C	Can Longer Passwords be Secure and Usable?	141
C.1	Part One Survey	141
C.2	Part Two Survey	143

D	Can Creation-Time Feedback Help Users Create Passwords?	145
D.1	Part One Survey	145
D.2	Part Two Survey	146

Chapter 1

Introduction

Users depend on their online accounts for myriad important and sensitive purposes. Users go online and create accounts to send email, find dates, and shop for Magic cards. Email accounts and social-networking accounts often represent a person's face to the outside world. Passwords are also used to protect accounts in business and academic settings. It is very common for universities to employ accounts for scholastic activities such as registering for classes and checking grades. Despite how much personal data is stored within these accounts, a single password is often the sole barrier keeping out a potential attacker.

Because account information is often protected only by a password, these passwords are attractive targets for theft. An attacker making live, online password guesses on the account website is called an *online attack*. Under this threat model, the service provider can notice the incorrect guesses being made and lock out the attacker after a specific number of incorrect guesses.

An attacker has many more chances to crack a victim's password in an *offline* attack. In this model, the attacker steals a password file from a service provider, containing the passwords of many potential victims. In practice, such theft has occurred in large scale in many instances in the past few years [11,32,34,40,78,95]. The threat model considered in this thesis is one in which the attacker has a hashed, possibly salted set of passwords. In order to determine the password for a particular account, the attacker guesses a password and salts and hashes the guess. If the resultant hash matches the one in the stolen password file, the attacker knows the guessed password to be the victim's password. The attacker can make a very large number of guesses against victims' passwords, limited by hash type and hardware available.

1.1 Password Policies, Strength, and Usability

Although stronger passwords better protect user accounts, users will often create easily guessed passwords when they can [5]. In order to increase password strength, many service providers have implemented *password-composition policies*. These constrain the available space of passwords in an effort to preclude users from selecting easily guessed passwords. Password-composition policies affect both security and usability; the passwords created under such a policy are ideally resistant to attackers' guesses while still being memorable to their owners.

While password-composition policies have existed long before this thesis, they were often based on heuristics and guesswork. Research on password-composition policies was not suf-

ficiently conclusive to recommend one policy or another. Some research looked at passwords without regard for the policies under which the passwords were created [5]. Other research had only a limited corpus of passwords from which to draw [51, 97]. Missing from the literature was work that contrasted large corpora of passwords created under controlled, randomly assigned conditions.

In this thesis, I present the results of using Amazon Mechanical Turk to collect security and usability data for participants creating and recalling passwords under assigned, randomized conditions. I advance the state-of-the-art in password-composition policy and password-composition-policy research through both the methodology and the findings. Over the course of four studies, I explore the security and usability tradeoffs of different password-composition policies, including system-assigned and user-generated passwords. The result is a better understanding of how to use policy to lead users toward stronger yet usable passwords, and actionable guidelines for service providers. My contributions also include a novel and potent technique for conducting human-subjects studies using Amazon Mechanical Turk, more akin to laboratory studies than traditional online surveys.

At a high level, I find that password-composition policies can affect password strength and password usability. The relationship between password usability and password strength is not necessarily directly inverse. Some password-composition policies – specifically those requiring longer passwords with some further character requirements – can lead to secure yet usable passwords. Overall, I find that while a tradeoff between usability and security must be made, some policies are more conducive to a favorable tradeoff. The rest of this thesis will explain and expound upon these findings.

Thesis Statement

The objective of this thesis is to create guidelines for password-composition policies that lead to strong yet usable passwords. These guidelines will be based on empirical data gathered through online crowdsourced human-subjects testing using randomized, controlled conditions.

1.2 Thesis Overview

Chapter 2 presents previous research on passwords and password policy. I present a list of recent password breaches to motivate my focus on offline attacks. I discuss approaches to measuring password strength. I then look at human-subjects passwords research, including previous passwords studies conducted on Mechanical Turk.

Chapter 3 presents SHELF, a framework for facilitating online studies on Mechanical Turk. SHELF is a generalized framework that other researchers could use to conduct other online human-subjects studies. I discuss the implementation of SHELF. Then, I discuss how researchers can use SHELF to make creating, monitoring, and analyzing data from their online studies much easier.

Chapter 4 describes the data-collection methodology I used in the four studies I present in this thesis. I present the user-study protocol as well as the security and usability metrics. This includes when each usability metric is important, and under what circumstances it is most relevant.

Chapter 5 presents *How Usable are System-Assigned Passphrases?*, in which participants were assigned passwords or passphrases. Conditions included short passwords, passwords intended to be pronounceable, and passphrases comprised of dictionary words. System-assigned passwords and passphrases had considerable usability difficulties, with a high rate of storage. Passphrases failed to out-perform passwords with equivalent strength in any usability metric, including self-reported sentiment and observed behavior. Passwords took less time to enter and were less error-prone than passphrases. Post-facto error correction could improve the usability of passphrases, but did not make them more usable than passwords on any metric.

Users struggled with system-assigned passwords, and so all of the subsequent studies focus on user-created passwords. Chapter 6 presents *How Secure and Usable are Some Common Password Policies?* This compared several real-world policies and demonstrated that policies can affect both usability and password strength. The two strongest policies it tested were a policy requiring only sixteen characters and a traditional “strong” policy requiring eight characters with multiple classes and a dictionary check. The 16-character policy led to more usable passwords and more security against an attacker able to make a large number of guesses. This study found that users often fulfilled the requirements of the traditional policy in predictable ways, reducing password strength. This study also found that dictionary checks can increase security, and that the choice of dictionary mattered.

Chapter 7 presents *Can Longer Passwords be Secure and Usable?*, a two-part study of policies that require longer passwords. Chapter 6 suggested that policies requiring longer passwords could have usability and security benefits over traditional “strong” password policies, but also enable some very weak passwords. These studies tested several conditions in an effort to retain the security and usability benefits of longer passwords while reducing the number of easily guessed passwords. The first study found that adding certain character-class requirements reduced the number of easily guessed passwords, while still being more usable than a traditional policy. There were patterns common in cracked passwords. For example, passwords containing the string *1234* were three times as likely to be cracked as those without. The second study examined policies that prohibited common substrings, and policies that placed further structural requirements on passwords. It also explored more combinations of length and character-class requirements. Using a substring blacklist increased password strength. While it made password creation more difficult, it had no adverse effect on password recall.

Chapter 8 presents *Can Creation-Time Feedback Help Users Create Passwords?* This study focused on making strict password-composition requirements, such as those studied in previous chapters, more palatable to users through real-time password-creation feedback. This study also examined guiding participants through a multi-step password-creation process. It found that letting participants know whether and how requirements were met during password creation helped them create passwords with fewer errors. On the other hand, guiding participants through password creation reduced password strength. This study found that looking only at requirements in isolation did not paint a complete picture of passwords. Presentation can affect user sentiment, behavior, and security.

The remaining chapters are based on the data from these studies. Chapter 9 looks back at the study metrics and survey questions. I discuss how observed factors correlated with self-reported user sentiment. I also discuss participants’ demographics, their use of mobile devices while taking the studies, and their responses about their own real email passwords.

Chapter 10 offers advice for system administrators. This is intended to provide practical guidance for better user security based on our findings. Finally, in Chapter 11, I conclude the thesis. I summarize the findings, discuss recommendations for researchers, and describe potential future work.

Chapter 2

Background and Related Work

This chapter provides context for many of the choices made in this thesis, including the choice of attacker model and the choice of data-collection protocol.

In Section 2.1, I present one way that an attacker can learn a victim’s password: an offline attack. I discuss how this thesis models offline attacks. I then discuss several recent data breaches to motivate my focus on the offline attack model in Section 2.1.3. Section 2.2 discusses password-composition policies, how they are intended to strengthen passwords, and how they are only effective against certain kinds of attacks. Section 2.3 discusses evaluating password strength, including information entropy and password guessability. Section 2.4 discusses the computational strength of several real-world attackers, and estimates how their computational power would translate into being able to crack passwords. Section 2.5 discusses human-subjects passwords research. I organize the presentation of those findings by methodology to highlight different methodologies that researchers have employed to study password composition.

2.1 Guessing Users’ Passwords

This section discusses the threat model used in this thesis and its assumptions. The goal is to help the reader better understand the security results, and understand the scenarios to which those results are applicable. This section first defines offline attacks, the attack model on which this thesis focuses. Then, this section discusses specific assumptions made about the attacker in this thesis.

2.1.1 Online and Offline Attacks

In the attacks described in this subsection, an attacker is attempting to break into a victim’s account. In an *online* attack against a password-protected account, the attacker tries guessing the victim’s password on a live login page. The service provider can notice this and lock out the attacker, though an attacker can avoid detection by making a smaller number of guesses over time [27].

This thesis focuses on the *offline* attack model, in which an attacker steals a *hashed* password file from a service provider. A *hash* is a one-way function that, given an input, produces output

that cannot easily be used to determine the input. The same input always yields the same output, but two similar inputs should not necessarily produce similar outputs.

Theft of hashed password sets is occurring frequently enough to make the news several times per year [24, 50, 55, 68]. In an offline attack, the attacker guesses a password and applies the same salting and hashing function as was applied to the targeted password in the stolen file. If the hashes match, the attacker knows the victim's password unless there was a collision. Suppose a user creates a password to access a website. The website stores password P with salt S and hash H as $H(P + S)$. When the user subsequently attempts to authenticate, he or she submits password attempt P' . The website then checks that $H(P + S) = H(P' + S)$. In an offline attack, an attacker steals the password file containing the user's salted and hashed password, $H(P + S)$. The attacker can make a larger number of guesses for P . For a guess P_N , if $H(P + S) = H(P_N + S)$, then the attacker has determined the user's password to be P_N .

Both online and offline attacks require that an attacker guess a victim's password to break into that victim's account. As I will discuss in Section 2.2.1, there are other attacks that are not prevented by strong passwords, such as phishing and shoulder surfing.

2.1.2 Assumptions in our Threat Modeling

This section discusses some of the assumptions made in how this thesis models password cracking.

Hashing the Password File

The research in this thesis assumes the attacker has stolen a password file, and that the passwords in that file are hashed. If the attacker were guessing passwords on a live system, then the service provider could detect the guessing and lock out the attacker. If the attacker stole a password file that kept passwords in plaintext, then the attacker has already met the objective of learning users' passwords. Some service providers have encrypted their password files using a single-key symmetric cipher [73]. This is considered a poor security practice, because the attacker acquiring the decryption key would immediately have access to all of the passwords. The attacker has already acquired a copy of the password file, which makes it likely the attacker has compromised the server on which authentication occurs.

Using a Slow Hash

The number of hashed guesses that an attacker can guess in a given period of time is bounded by several factors. The more computing power available to an attacker, the more guesses the attacker can make. However, a service provider does not have control over this. A service provider does, however, have control over the hashing function being used. A slower hashing function leads to each guess the attacker makes taking longer.

A slower hash causes each legitimate authentication attempt to take more time. It can also increase server load for legitimate authentication [73]. However, given that an offline attacker may be making 10^{12} or more guesses, it hinders an attacker far more than legitimate users or server providers. Using a slow hash is a security measure that increases cost slightly for the legitimate user, and by a much larger factor for the attacker.

Salting the Password File

The threat model assumes that passwords are hashed with a slow hash and salted. A *salt* is a string that is added to a password before that password is hashed. Adding a unique salt to each password before it is hashed does two important things. First, it prevents attackers from using *rainbow tables*, which are pre-computed sets of password hashes to look up password hashes. Second, it means that each user's password must be cracked individually. Without a salt, two users with the same password would have the same hash, so determining the password of one user would also determine the password of all users with the same password.

2.1.3 Real-World Password Breaches

In the past few years, it has become common to hear about stolen password sets. In 2010, these breaches were common enough for some to call for an end to passwords altogether [62]. However, researchers have argued that text passwords will likely persist because, despite their shortcomings, they are entrenched and there is no especially promising alternative [8, 35]. This thesis focuses on the offline attack model, which requires that a password file be stolen from a service provider. This section mentions recent large-scale password breaches to help demonstrate that the offline attack model is worth studying.

At the start of this decade, the media reported on large-scale password breaches. In 2010, attackers stole 748,490 usernames and passwords from Gawker. The passwords were salted and hashed, but the hashing was executed poorly. Gawker used DES, truncated passwords to 8 characters, and removed non-ASCII characters [6]. In 2011, attackers stole account information for 24.6 million Sony Online Entertainment users. The compromised data included passwords, phone numbers, logins, and addresses [76]. Further, attackers claimed to have compromised one million unhashed passwords on `sonypictures.com` through an SQL injection attack [78]. Attackers in 2011 also took 90,000 pairs of email and hashed passwords from military contractor Booz Allen Hamilton [11]. Attackers stole 90,000 hashed but unsalted passwords from Swedish blog website Bloggtoppen [34] and 1.3 million login credentials from Sega, including emails and encrypted passwords [40].

Several password breaches stood out in 2012. Attackers stole 6.5 million hashed passwords from the website LinkedIn [67]. A fifth of the hashed passwords could be cracked in a few hours using publicly available tools on a "customer grade laptop." Over one third of the cracked passwords had length eight or less [95]. Attacks used an SQL injection to steal 453,000 login credentials from a Yahoo server, including unhashed passwords [32].

The year 2013 saw a large number of password breaches. Attackers obtained account information, including encrypted passwords, from 2.9 million Adobe users [2]. Adobe claims to have reset the password information automatically for affected users, and recommended changing the password on any website using the same password. Software company Evernote forced its users to reset their passwords after its authentication data was compromised. The company reported that the compromised data included usernames and salted, encrypted passwords [24]. Twitter reported that it found evidence that an attacker had gained access to information for a quarter-million of its users, including usernames, email addresses, and encrypted-and-salted passwords [55]. Video-delivery website Vudu, owned by Walmart, suffered a physical theft of its servers. Included on the stolen machines was user information including names, email addresses, and encrypted pass-

words. The company responded by resetting all of its users' passwords [50]. Internet coupon website Living Social reported that user data for 50 million users had been taken. This included email addresses, user names, and salted, hashed passwords [68].

2.2 Password-Composition Policies

A rational attacker conducting an offline attack guesses the most likely passwords first. This is an effective attack against users who create easily predicted passwords. In practice, as discussed in Section 2.5, many users create passwords from a fairly small pool of weak passwords. A stronger password, therefore, is a password that an attacker is unlikely to guess. In order to prevent weak passwords and increase password strength, many service providers employ *password-composition policies* – sets of requirements for users creating passwords. These policies can help prevent attacks that require the attacker to guess a password. As discussed in Section 2.2.1, there are some attacks, such as phishing and shoulder-surfing, that are not countered by having strong passwords.

As an example of a password-composition policy, consider the one Carnegie Mellon University employs. We refer to this policy as *comp8*; it requires eight characters, one of each of the four character classes, and a dictionary check.¹ This policy is based on the authentication requirements for member universities of the InCommon Federation [38]. Those requirements, in turn, are based on guidelines published by the United States National Institute of Standards and Technology (NIST) [16].

When Carnegie Mellon University transitioned from a lenient policy to its strict *comp8* policy, a number of its community members expressed annoyance [88]. An ideal password-composition policy helps users to make strong passwords, without making it onerous to create and recall passwords. User attention and effort are finite resources, and this can affect password behavior [28]. Moreover, many service providers employ password policies that do not place realistic expectations on users and are not supported by research [27].

Researchers have been advocating password-composition policies for many years. In 1995, researchers used a dictionary-based attack to guess roughly 40% of the roughly 14,000 hashed passwords that administrators had provided to them. This resulted in those researchers advocating “proactive password checking” to prevent easily guessed passwords [5].

Several studies have demonstrated negative consequences of password-composition policies. In work performed before this thesis, colleagues and I used simulation to show that an organization's financial health can be negatively affected by using onerous password-composition policies [83]. A 32-participant diary study, published in 2010, observed that password-composition policies were often burdensome to users and could lead to decreased productivity [39]. Another interview study suggested that users struggle to make strong passwords [91]. These studies indicate that the password-composition policies intended to protect users can be burdensome to users and the service provider.

Password-composition policies are intended to make it difficult for attackers to guess passwords by making passwords less predictable. However, their effectiveness is often limited because users tend to fulfill their requirements in predictable ways. For example, in one study we found that users often select symbols from only a small fraction of the symbols on a keyboard [88]. In

¹<http://www.cmu.edu/iso/governance/guidelines/password-management.html>

another study, researchers found that when users employ digits, symbols, and uppercase characters in their passwords, they often do so predictably [97]. NIST wrote in its electronic authentication guideline that users are expected to create passwords of mostly lowercase letters when not otherwise required, and to fulfill requirements in predictable ways [15].

2.2.1 The Limits of Password Policies

Password-composition policies are designed to induce difficult-to-guess passwords. Therefore, an attack that is not hindered by password strength will not be prevented by a password-composition policy. Such attacks include key-logging attacks, in which the victim's computer is rigged up to provide illicit data to an attacker. An attacker can also observe a user entering a password, leading to a shoulder-surfing attack. Users can be tricked into giving up their passwords, such as with a phishing attack. Password-composition policies won't help protect accounts if the service provider discloses the contents of those accounts. Governments can often force information disclosure, such as with a subpoena. Users can be coerced or threatened into giving up their passwords.

Another way that an attacker can break into a victim's account without guessing the victim's password is by taking advantage of password-reset mechanisms that many service providers offer. For example, attackers compromised several accounts of Wired editor Mat Honan by hijacking password-reset mechanisms. They never needed to crack his passwords [36].

Password-composition policies are not a complete security solution because passwords are not a complete security solution. They can keep out some classes of attackers, but cannot keep out all threats. Passwords are a part of a complete security approach. Some attackers may have the resources to crack passwords, while other attackers may circumvent passwords entirely. The goal is not to make users secure, but to make them *more* secure. Service providers should, for example, monitor their networks for suspicious activity, including online attacks. This can include noting that a single user account has made a large number of failed login attempts. If a service offers a single sign-on system such that increased value is placed behind its passwords, having password security becomes increasingly important.

One way to increase account security is to use two-factor authentication. For example, Google² and Microsoft³ both offer services that allow the user to designate a mobile phone number. If someone tries to authenticate as that user from a new device, a code is sent to the mobile device. That code then needs to be entered in order for the attacker to get into the account. This would help protect against an attacker who might obtain the user's password, such as via key-logging or phishing.

2.3 Measuring Password Strength

This thesis contrasts password-composition policies, including the strength of the passwords created under them. I next discuss two measures of password strength found in the literature: information entropy and password guess numbers. I discuss both metrics, and discuss why this thesis favors using guess numbers as a comparative metric of password strength.

²<https://www.google.com/landing/2step/>

³<http://windows.microsoft.com/en-us/windows/two-step-verification-faq>

2.3.1 Entropy

The *information entropy* of a given text corpus, including a password set, is a measure of the information or uncertainty contained within that text, in bits. Claude Shannon introduced the concept of information entropy in 1949 [80]. Shannon’s algorithm for estimating the entropy of a corpus, which he called *n-gram entropy*, calculated statistics for each substring of length n in that corpus. The entropy of a given character is a function of the conditional probability of that character being present, given the previous $n - 1$ characters. The entropy of each character is estimated and summed; because entropy is additive, this produces the entropy of the entire corpus. A larger value of n leads to a more accurate entropy estimate because more characters are used as context for the entropy estimation, but the calculation also becomes more computationally intensive [81].

Let C be a corpus of text. Shannon’s formula for the n -gram entropy of C is:

$$- \sum \Pr(a + b) \log_2 \Pr(b | a)$$

- a is string of length $n - 1$.
- b is a single character.
- $a + b$ is a string of length n , comprised of a concatenated with b .
- $\Pr(a + b)$ is the probability of the string $a + b$ in C , which can be calculated as the number of instances of $a + b$ in C divided by the total number of n -grams in C .
- $\Pr(b | a)$ is the conditional probability of b given that the previous $n - 1$ characters are a , which can be calculated as the number of instances of $a + b$ divided by the number of $n - \textit{grams}$ whose first $n - 1$ characters are a .

Shannon further stated that when $N = 0$, n -gram entropy is $\log_2 26$ [81].

Calculating entropy based on insufficient text samples leads to an underestimate of the actual entropy [61]. The entropy of a password provides a theoretical lower bound on the number of guesses required to crack that password for an attacker using an optimal guessing strategy [57]. In 2006, the National Institute of Standards and Technology published guidelines for password-composition policies, with different policies having projected per-password entropy [16]. Passwords research has used entropy to quantify password strength [26, 88].

2.3.2 Guessability

While entropy provides a theoretical bound on how quickly an attacker can crack a password [57], a number of researchers have advocated using guessability instead of entropy as a metric of password strength. The *guess number* of a password is the number of guesses that would be required for a given password-guessing algorithm, with a given set of training data, to guess that password.

John Pliam compared entropy unfavorably to what he called “marginal guesswork,” a concept that correlates to what this paper calls guessability. Marginal guesswork is the number of guesses required to determine a secret with a given probability of success. Pliam stated that, when items to be guessed have an even distribution, entropy and guessability converge. This is relevant to system-assigned passwords, because if system-assigned passwords have an even distribution, their entropy and guessability converge. However, Pliam proved that there are also distributions

Estimated Attacker	Based On	Time for 10^{12} bcrypt hashes	Time for 10^{12} MD5 hashes
Single Computer	2012 MacBook	Over 2,000 years	56 days
Hactivist	Estimates of Anonymous	207 days	20 minutes
Criminal Network	Torpig Botnet	27 days	2.5 minutes
Foreign State	Chinese Supercomputer	5 days	30 seconds

Table 2.1: A summary of estimated time for various possible attackers to reach 10^{12} hashes using bcrypt with its default cost factor of ten. Each increase in the cost factor would increase the time taken by a factor of two. For comparison, the table also contains the estimated time for hashing using MD5. The assumptions made for each estimate are in Section 2.4. This omits one’s own government because they can often use other means to obtain data stored in servers under their jurisdiction.

in which entropy and marginal guesswork diverge. One example is a password set primarily composed of weak passwords but also containing a few very complex passwords. Those complex passwords raise the average entropy for the entire set, even though the bulk of the passwords are easily guessed. Whereas, for marginal guesswork, a password is classified as either cracked or not cracked for a given guess threshold, so a few hard-to-guess passwords do not skew the result [69].

Joseph Bonneau also argued that entropy is not an appropriate metric for evaluating password strength. He argued that entropy does not correlate directly to the difficulty faced by an attacker in guessing a password. Instead of considering the average strength of passwords in a password set, Bonneau argued in favor of measuring how much of a password set is easily cracked, which he called a “partial guess metric” [7].

Because of the reasons given above, guessability is the primary metric of password strength in this thesis. As discussed in Section 4.2, this meant generating *guess numbers* for the passwords. A guess number is the number of guesses that a given password-guessing algorithm, with a given set of training data, would take to guess a specific password.

2.4 Guess Numbers and Threat Modeling

The studies in this thesis present the number of passwords per condition cracked after a given number of guesses, often to a threshold around 10^{12} . This subsection discusses how those guess numbers might correlate with different attackers. Section 2.4.1 discusses bcrypt, the hashing function on which my guess-number estimates are based. Section 2.4.2 iterates through a number of attackers and estimates their ability to make password guesses. Table 2.1 summarizes these estimates.

2.4.1 Password Hashing with bcrypt

Many hashing algorithms take a fixed amount of time to execute. The problem with these fixed-time hashing algorithms is that an algorithm that is slow on today’s computer systems may be executed much more quickly on future systems because of increasing computing capacity. Provos and Mazieres, working for the OpenBSD Project, presented a solution to this problem in 1999. They presented the *bcrypt* hashing scheme, which is designed for use with passwords. The algorithm is intentionally relatively slow. Moreover, bcrypt is designed to take as a parameter a *cost*

factor, which can be used to increase exponentially the time the hash takes to execute. Increasing the cost-factor parameter by one doubles the time. This allows hashing with `bcrypt` to be made more resource-intensive as computer power increases [72].

`bcrypt` has become the default password-hashing function in FreeBSD [72]. It has also become the default password-encryption function in Ruby on Rails,⁴ with a free implementation in Ruby.⁵ The Openwall project has made free versions of `bcrypt` available on other platforms.⁶ SHELF, the online survey framework described in Chapter 3, hashes researcher passwords with `bcrypt`. Researchers have found that using specialized hardware can lead to a three-to-fourfold increase in hashing speed with `bcrypt` [56]. The subsequent discussion will be about hardware that has not been specifically designed for `bcrypt`.

2.4.2 Guess Numbers for Specific Attackers

First, consider an attacker using a standard laptop computer to hash passwords. This might represent an attacker unable to assemble more resources to hash passwords, or an attacker conducting an attack of convenience. For benchmarking purposes, I am using my own laptop.⁷ `bcrypt` has a default cost factor of 10, meaning that it iterates 2^{10} times. I performed a benchmarking test by hashing 1,000 passwords of eight random lowercase letters each. This took 73.3 seconds, less than a tenth of a second per password. In order to reach 10^{12} guesses against a password file encrypted with the default cost factor of 10, it would take the attacker over 2,000 years. Increasing the cost factor to 11 roughly doubled this to take 143.5 seconds for 1,000 passwords. Increasing the cost factor to 20 made hashing take about one minute and 15 seconds per password on my laptop. For comparison, using MD5 to hash 10^{12} passwords on the same computer would take 55.9 days.

Next, consider a “hactivist” attacker such as Anonymous. This attacker may be comprised of a loosely organized cluster of individual actors, perhaps acting in order to achieve a political goal. The computing power of such a loose organization is not easy to determine, but I can estimate. Estimates for the number of people online in an Anonymous chatroom can number around 7,000.⁸ In 2011, PayPal chose to give the FBI the 1,000 most active IP address during an attack. In 2012, Anonymous itself claimed to have over 9,000 members.⁹ Those higher numbers are for a less computationally intensive task than password-cracking, however. Based on this, I conservatively estimate that Anonymous might be able to harness the computing capacity of a few thousand computers. There is no indication that Anonymous taps into further resources, or that they have the full resources of those computers. Therefore, as a very conservative estimate, suppose that Anonymous has access to the resources of 4,096 computers. For simplicity, assume that they are all similar to my laptop, described above. Then, using the standard cost factor of 10, 10^{12} hashes would take about 207 days.

Another threat to password security is criminals. Criminals may be extra-legal organizations, similar to hactivists. However, unlike hactivists, they are more likely to hire a botnet, rather

⁴api.rubyonrails.org/classes/ActiveModel/SecurePassword/ClassMethods.html

⁵<https://rubygems.org/gems/bcrypt-ruby>

⁶<http://www.openwall.com/crypt/>

⁷2012 MacBook Pro, running OSX 10.10.1 (Yosemite), Retina display, 8 GB 1600 MHz Memory, with a four-core single-processor 2.3 GHz Intel Core i7, running ruby 1.9.3

⁸<http://www.wired.com/2014/06/anonymous-sabu/>

⁹<https://twitter.com/YourAnonNews/status/160283918526980096>

than depend on donated computing resources. The exact computational resources of a criminal organization are unknown and often exaggerated,¹⁰ so I will make an estimate. Researchers from UC Santa Barbara took control of the *Torpig* botnet in 2009, giving a rare view inside a large-scale botnet. They found that at any given time, the mean and median size of the botnet was just under 50,000 machines [92]. Further, a botnet is often unable to utilize the full computational capacity of an infected machine. Flow-control between nodes would further tax computational capacity. Therefore, I conservatively estimate that a botnet would provide an attacker with the equivalent of around 32,000 full-time, dedicated machines. Getting to 10^{12} guesses on bcrypt with the default cost factor of 10 would take about 27 days.

Consider a foreign nation-state. Finding reliable numbers for the capacity of such an attacker may not be possible. Therefore, assume that the attacker has access to a top-of-the-line supercomputer. The world's top supercomputer as of November 2014, according to `top500.org`, has a speed of 33,862.7 terraflops.¹¹ This machine resides in China. According to benchmarking software Xbench¹², my laptop has a speed of 211.06 gigaflops. Based on those numbers, the Chinese supercomputer has a speed of roughly 160,441 times that of this laptop. With a default cost factor of 10, the super computer could make 10^{12} hashes in about 5.3 days. Increasing the cost factor of bcrypt to 20 would cause the super computer to take about 14 years.

Finally, consider the government of the United States. This can include agencies such as the NSA and FBI. These agencies do not publish estimates of their computing capacity, making it difficult to determine their capacity. More importantly, in many scenarios for a US-based organization, password strength may not be useful to protect data from the NSA. The United States government can issue a subpoena to compel a service provider under its jurisdiction to disclose information to which it has access. Therefore, for most large-scale US online service providers, password strength against the NSA's computing capacity may not be helpful.

Table 2.1 summarizes my estimates for computing power for different attackers. These estimates are presented in rough calculations of how long different attackers would need to hash 10^{12} guesses using bcrypt with its default cost factor of 10. Note that these are estimates for all but the single computer. Further, the nature of bcrypt allows the hashing time to be adjusted; each increase in the cost factor would double the estimated time.

2.4.3 Attack Models and Acceptable Loses

There are some use-cases in which it would be unacceptable for any accounts to become compromised. This might be, for example, a situation in which doctors are using passwords to protect patient information, and a single account being compromised might lead to a lawsuit. Another example might be where there are specific data-protection laws in place for certain classes of data, such as the Health Insurance Portability and Accountability Act (HIPAA).¹³ In these cases, despite their usability difficulties (as discussed in Chapter 5), system-assigned passwords might be a reasonable approach. They would let the administrator determine password strength, and prevent the users from creating weak passwords.

¹⁰<http://www.zdnet.com/article/botnet-size-may-be-exaggerated-says-enisa/>

¹¹<http://www.top500.org/list/2014/11/>

¹²<http://www.xbench.com>

¹³<http://www.hhs.gov/ocr/privacy/>

In some cases, it may not be worthwhile for a service provider to try to prevent all accounts from being compromised. A service provider forming a password-composition policy should be mindful of roughly how many compromised accounts are acceptable. I led development of a simulation tool to help organizations reason about how best to balance password strength and usability for their own context [83, 84].

There are several factors for a service provider to consider. Not all types of accounts are the same. In a work environment, user accounts are used by employees who generate value for the company, so it may be especially problematic for them to be compromised. Even within a work environment, some accounts may have more value than others; a president's account being compromised may cause more harm than that of a typical worker. On the other hand, there are contexts where some users may not be concerned about compromised accounts. For example, users of an email client may be unconcerned about their spam accounts being compromised. Likewise, users may be less concerned about their low-value accounts being compromised, such as accounts they were forced to make to read a newspaper online. Another consideration is whether having some number of compromised accounts might threaten the integrity of the system itself, which would be more common in a work environment than for a webmail client.

2.5 Human-Subjects Research on Passwords

This section presents related human-subjects passwords research. Section 2.5.1 presents human-subjects research on two specific categories of passwords: passphrases and system-assigned passwords. In order to highlight the strengths and weaknesses of different methodological approaches, I organize the remaining sections by methodology. I examine password-related surveys in Section 2.5.2, longer-duration in-situ password studies in Section 2.5.3, and laboratory studies in Section 2.5.4. Section 2.5.5 looks at interview studies. Finally, in Section 2.5.6, I discuss other Mechanical Turk password studies.

2.5.1 Passphrases and System-Assigned Passwords

Passphrases are passwords comprised of a sequence of natural-language words. They have been discussed in the literature for over three decades [70]. The comic xkcd helped to rekindle interest in passphrases by declaring their superiority over similar-strength passwords [64]. An academic institution's blog even referenced the comic [77]. However, human-subjects studies focused on passphrases are relatively uncommon. Amazon has a password mechanism it calls a *payphrase*, which is a user-created password composed of multiple words used to facilitate online ordering. An analysis of 100,000 payphrases, published in 2012, found that they often contained predictable patterns. For example, they contained titles of popular movies and books, as well as digrams of words commonly found together in natural language. This suggests user-created passphrases may contain predictable patterns [9].

In most circumstances, users create their own passwords. However, some research has focused on *system-assigned* passwords. Assigning passwords allows service providers to ensure minimum strength. The idea of assigning a password has existed for many years [52], but has not become wide-spread. Research has shown that users struggle when assigned a password. A study published in 2007 asked 19 student participants to recall a system-assigned password after two weeks.

Across different password-creation algorithms, participants struggled with recall [54]. Another study, published in 2009, focused on login success across three different password policies. In this 12-week field study, each of 52 undergraduate participants was assigned to one of three conditions, and used the passwords with their coursework. These conditions included user-created eight-character passwords with a non-letter; system-assigned passwords; and passphrases with at least 16 characters. Participants assigned passwords had the highest instance of login failure, while participants who created sentence-like passphrases had the least [43].

2.5.2 Password-Related Surveys

Researchers have employed surveys to learn about user behavior. Surveys are often not as in-depth as laboratory studies, and they do not facilitate contrasting different conditions. They tend to provide description of current practices and sentiment, rather than facilitating investigation of prospective password policies. Surveys tend to be easier to administer per participant than laboratory studies, and therefore they tend to have more participants.

A common theme in password-behavior surveys is that participants reused and wrote down passwords. In 1997, Adams et al. reported a password survey of 139 participants. Half of the participants reported writing down their passwords. Half of the participants with more than one password connected most or all of their passwords with a common theme or creation technique [1]. In a survey of 997 Department of Defense employees, published in 1999, Zviran and Haga asked about password creation and characteristics. The most common password length was six characters, and 80% of participants used all-alphabetic passwords. 80% also never changed their passwords. Over a third of participants reported writing down their passwords. The authors found no association between password composition and the importance of the data protected by that password, or between how a password was selected and its importance [101]. Medlin et al. administered a survey on password creation and behavior to 118 healthcare workers, publishing the results in 2008. This survey included a question that asked those healthcare workers to share their home or work passwords. Workers who frequently changed their passwords were more likely to share them. Overall, 73% of workers did share a password [59].

Many surveys have focused on password use within academic communities. In part, this may be because the researchers are already located in a school. In addition, an academic institution provides a number of users sharing a single password policy. A survey of 218 psychology students, published in 2004, found that the average user had 8.2 passwords but only 4.5 unique passwords [12]. There is evidence that the number of accounts user have has increased since then. Experian released results from a 2013 study of 2,000 adults from the United Kingdom that showed an average of 19 online accounts each [25]. Kumar presented results in 2011 from 195 members of a university community in India. 45% of participants never changed their passwords and 70% reported reusing them. Participants employed a number of password-creation techniques, the most common being to base the password on the name of a family member [49]. In 2006, Bryant and Campbell published results from a password study of 884 undergraduate students in Australia. The most common password length was eight characters, with only 7.4% of participants using over 11 characters. Around a third wrote down their passwords [13]. Another survey, published in 2006, looked at the password behavior of 315 students. Most participants, 60%, did not to create more complex passwords for higher-value websites. 53% of participants did not change their passwords regularly unless required [74].

In 2010, we asked 470 Carnegie Mellon students, faculty, and staff about their password-related attitudes and behaviors. This survey occurred just as the University transitioned from a lenient password-composition policy to a much more stringent policy. This allowed for a deeper look at how users respond to encountering strong password-composition policies. While participants were annoyed with the new policy, they also felt that the new policy provided better protection and were ambivalent about reverting to the previous policy [88].

2.5.3 Longer-Duration Password Studies

Several studies have examined how participants use their real passwords over longer stretches of time. This allowed researchers to learn about patterns in user behavior. These studies tend to study users in situ, rather than assigning them to study-specific conditions.

Florencio and Herley conducted a large-scale password study over three months, publishing the results in 2007. 544,960 participants installed a Windows toolbar that monitored their password behavior. The researchers gathered information about how many accounts users had, their password strength, and their password reuse. They found that users had about 25 accounts each. The average user had 6.5 passwords, typing eight passwords per diem. Password reuse was common and most passwords contained only lowercase letters unless required otherwise [26]. In 2010, Inglesant and Sasse reported results from a week-long diary study of 32 office workers. Users were concerned about security, but often struggled to follow password policies. Their password policies often hindered productivity. Only 60% of participants used a password with at least eight characters [39]. In 2011, Grawemeyer and Johnson presented a seven-day diary study with 22 participants. Participants authenticated on average 45 times per diem. Users stored, shared, and reused passwords. The most common password-creation techniques were basing a password on a phrase (28%) or on a single word or name (26%) [33].

2.5.4 Laboratory Studies

Laboratory studies have examined password behavior. Similar to the online studies in this thesis, laboratory studies can have users create passwords under specific sets of requirements. However, they tend to be able to draw participants only from a limited geographic area. Further, because they require researcher interaction, they tend to be limited to a small number of participants.

A 2002 paper from Proctor et al. described the results of two laboratory studies on password creation. 24 Purdue undergraduate students created two passwords each, one with at least five characters and one with five characters plus additional requirements, such as having a digit and an uppercase letter. Passwords with additional requirements took significantly longer to create and had significantly more creation errors. John the Ripper cracked 18 of the 24 passwords without additional requirements and only eight of the passwords with additional requirements. Most of the passwords created under the additional requirements followed a common pattern of starting with an uppercase letter and ending with a digit. The authors repeated the experiment with a minimum length of eight characters rather than five. John the Ripper cracked four of the 24 passwords with only the length requirement, and three of the 24 passwords with the additional requirement. Increasing the length of passwords was an effective way to increase their strength while being more usable than other requirements [71].

In a 2006 study, Gaw and Felten asked 49 undergraduate students for a list of websites on which they have accounts, and were asked to recall their passwords for those websites in a laboratory setting. Password reuse was very common, and participants were more likely to reuse a password exactly rather than with modification. Participants with more accounts tended to reuse passwords more often. Participants were also shown different passwords and asked about their security; participants considered password security from the perspective of a human attacker actively guessing a victim's password [31].

Vu et al. published in 2007 a three-part laboratory study with Purdue undergraduate students. In the first part of the study, 32 participants created either three or five passwords, each with a number of requirements that included having four character classes. Participants with five passwords needed more login attempts than those with three passwords, and made stronger passwords. In the second part, 40 participants created mnemonic passwords, with half being given additional restrictions. Passwords with additional requirements took longer to enter, but were stronger. In a third experiment with 60 participants, passwords derived from a whole sentence were not easier to recall than a mnemonic password based on a sentence [96].

Forget et al. presented work in 2008 that explored persuading users to create more secure passwords by adding random characters in random locations to passwords users had just created. The authors found that inserting two characters increased password security without perceptibly affecting usability. Furthermore, adding more than two characters resulted in decreased usability but no more security, since users would select weaker pre-improvement passwords to compensate for the increased memory load of additional characters [29].

In a paper published in 2013, Egelman et al. examined password meters using a laboratory study and an online study. In the laboratory study, 47 members of a university community created new passwords. Meters improved password strength. In a follow-up study on MTurk, password meters did not improve password strength. The authors concluded that password meters increase password strength, but only for higher-value passwords [23].

2.5.5 Interview Studies

Studies have used interviews to learn more about password behavior. An advantage of an in-person interview over an online study is that it facilitates follow-up questions in response to what participants say. Interviews can also be more in-depth than surveys. On the other hand, interviews tend to be time-consuming for researchers. In practice, interviews have focused on password use under existing policies.

Singh et al. conducted a qualitative study on password-sharing habits in Australia, publishing the results in 2007. They found that password sharing was common. Couples, in particular, tended to share passwords with one another as a sign of trust. Disabled people reported being forced to share their passwords with a caretaker [89].

Notoatmodjo and Thomborson conducted a laboratory study about password behavior and perceptions with 26 student participants in New Zealand, publishing the results in 2009. Participants categorized and described their passwords. Passwords for higher-value accounts tended to be longer, though not significantly so. The authors found evidence that users with more accounts tended to reuse passwords more often. They also found that users were more likely to reuse passwords for lower-value accounts [66].

Stobert and Biddle published a study in 2014 that described their efforts to use grounded theory to paint a picture of the password lifecycle. The authors conducted 27 interviews about how participants used passwords. Many participants wrote down their passwords as a backup, and did not refer to the written passwords regularly. Further, none of the participants used a dedicated password manager, instead primarily using their browser. Users budgeted their cognitive resources in the space of passwords. Users also had a poor understanding of the threats that their passwords faced [91].

2.5.6 MTurk Passwords Studies

The Carnegie Mellon passwords group has used Mechanical Turk to conduct human-subjects passwords research in studies not included in this thesis. These studies have followed a data-collection protocol similar to that described in Section 4. Researchers recruited participants online, asked them to create or memorize passwords under a given policy, and asked them to recall the password after a few minutes and a few days.

Our group conducted an online study on the effects of password-strength meters, publishing the results in 2012. We asked participants to create passwords under different policies while being shown password-strength feedback as a meter. Meters could lead to stronger passwords, though specific visual stimuli did not appear to affect password strength. The meters that were more difficult to fill did lead to stronger passwords, up to a point. Beyond a threshold, users did not respond to further requirements with stronger passwords [94].

Another study, published in 2013, examined the strength of actual passwords used by the students, faculty, and staff of Carnegie Mellon University. Our passwords group indirectly studied metrics from genuine passwords. We also conducted an MTurk study in which workers created study passwords under the same requirements as actual members of the Carnegie Mellon University community. While MTurk passwords were slightly more vulnerable to cracking than genuine university passwords, they were fairly close in strength. Further, the two sets of passwords had many similarities including their general composition. This work further validates the external validity of using MTurk to study and contrast password-composition policies [58].

Some members of our passwords group collaborated with researchers from Microsoft Research to study a form of real-time password-creation feedback. We explored the usability and security implications of an algorithm that displayed predictions for the next characters participants might type as they created their passwords. The objective was to prevent participants from entering characters that were the most likely to be guessed, based on the characters they had already entered. Feedback affected user behavior and password-creation usability. Requiring a number of non-predicted characters increased password strength without diminishing usability [47].

The studies presented in this thesis use a similar data-collection protocol as the above studies to understand passwords. However, they differ in substantial ways. The two above-mentioned studies that compare conditions focus on the visual element of password-strength indicators for user-created passwords [47, 94]. The study presented in Chapter 5 focuses on system-assigned passwords, rather than user-created passwords. The studies in Chapters 6 and 7 contrast password-composition policies without any visual element. The final study presented in this thesis, in Chapter 8, does focus on visual elements during password creation. However, rather than using its visual element to indicate password strength, it indicates whether and how the user's password has met password-composition requirements.

Chapter 3

Experimental Framework

In the studies presented in this thesis, I wanted to draw causal conclusions about how different sets of password requirements affect password strength and usability. This posed a number of data-collection challenges. Some of the password-composition policies I wanted to study were not already deployed in practice. Further, the analyses would require collecting enough data to detect significant differences between conditions. Those considerations meant needing to collect data from a large number of participants who were randomly assigned to conditions.

Chapter 2 discusses previous methodologies used to study password policies. None of them met these requirements. Laboratory studies can study how participants create passwords under randomly assigned sets of requirements. However, they are very time-consuming and labor-intensive, making it infeasible to collect data from thousands of participants. Surveys and in-situ studies can be faster to administer and can accommodate larger numbers of participants. However, they are better suited to studying the current state of password usage, and less able to facilitate assigning participants randomly to conditions.

The solution was to combine the large numbers of participants of online surveys with the randomly assigned conditions of laboratory studies. We could use Amazon's Mechanical Turk crowdsourcing service (MTurk) to recruit and pay participants. Assigning participants to conditions, and keeping their data organized, would be much easier with a dedicated study framework. I wanted a data-collection framework with the following capabilities:

- Use MTurk to screen, recruit, and pay large numbers of participants.
- Facilitate interactions with MTurk like posting tasks, emailing, and paying participants.
- Assign participants to controlled conditions.
- Automate data collection so researchers do not need to be hands-on with each participant.
- Keep data from thousands of participants organized.
- Coordinate data for multiple experiments on MTurk, Survey Gizmo, and local servers.
- Be able to prevent any one person from participating in more than one study.
- Provide researchers with a panoptic, real-time view of data collection to monitor progress.

I met these requirements by developing SHELF, a general framework to facilitate large-scale online human-subjects studies. SHELF has data-collection and study-monitoring features. I was SHELF's primary architect and developer. Saranga Komanduri implemented some features, including much of the interface between SHELF and MTurk. We used SHELF in each study presented in this thesis, and refined and improved SHELF between studies. Section 3.1 describes the implementation of SHELF. Section 3.2 highlights some of SHELF's salient features, and how they helped to meet the design objectives discussed above.

3.1 SHELF Implementation

I implemented SHELF in Ruby on Rails – SHELF uses Ruby v1.9 and Rails 3.1.1. Our studies have used MySQL for the back-end database, but SHELF is intended to be database-agnostic. SHELF itself is a generalized experimental framework – there is an abstract form of SHELF that has the base code common to many use-case scenarios. Each experiment to be conducting using SHELF has its own, separate *instance*, or fork, of the framework. An administrator starting a new SHELF project copies the abstract SHELF template, and then modifies it for the specific experiment. This includes creating the specific subject-facing pages, and creating the study conditions.

SHELF is an Internet-based framework. An instance of SHELF is run on a server, with its own URL. A researcher using SHELF goes to a specific URL and authenticates with a login and password. A participant taking a study on SHELF is given a URL for the study, which can include that participant's MTurk workerid.

SHELF has so far primarily been used for passwords research. Therefore, I have created an abstract form of SHELF intended for passwords studies. Password-specific features in the abstract password SHELF include facilities for password creation and recall. Most of the studies in this thesis were created by copying the abstract password SHELF framework and customizing it for the specific study. Throughout the studies in this thesis, I have constantly improved SHELF. If a new feature was specific to a given study, then I added it to the specific instance of SHELF. On the other hand, if the feature was more generally useful, I added it to the abstract SHELF.

SHELF evolved over the course of the studies in this thesis. For example, in the study presented in Chapter 6, SHELF was monolithic and all studies were contained in the same database. As our passwords group conducted more studies, I realized that having each experiment isolated in its own SHELF instance would keep overall database size down, and allow more experiment-specific customization. Many of the other features described in this chapter were not present in the initial version of SHELF, but were added as the system was used.

Creating a new experiment in SHELF requires copying the abstract SHELF template, and then modifying that template. This requires some knowledge of Ruby on Rails. I designed SHELF to require as little coding as possible; I tried to give SHELF many reasonable defaults, and much of an experiment can be configured through a web interface. A number of participant-facing pages are already present in SHELF, such as a consent form. In addition, SHELF comes with a set of default database tables, which are sufficient for many human-subjects study protocols.

The following are default database tables in SHELF. Each includes a built-in researcher-facing webpage to facilitate monitoring study progress.

- **User:** A researcher.

- **Setting:** A number of variables determine aspects of the study and are stored as variables in a table rather than being hard-coded into the system to facilitate changes. These include how much participants are paid and whether the study should be launched in sandbox mode.
- **Subject:** This is an individual MTurk user. Depending on settings, a subject may be allowed to participate in one or more studies.
- **Run:** A pass of a Subject through a study. Each run belongs to a Subject and to a Condition.
- **Condition:** A Condition has multiple Runs, and each Run belongs to exactly one Condition. In this study, a Condition determines the password-composition policy.
- **Experiment:** An Experiment represents an individual study and has a set of Conditions.
- **Collection:** This is a set of Runs that are analyzed together. For example, all of the pilot participants may constitute a Collection.
- **Visit:** A visit represents a Subject visiting a page in the experiment, as part of a Run. Each Run has many Visits.
- **Input Record:** This represents the input acquired from a participant on a given page. For example, the password-creation field on a page would have an Input Record for keystrokes.

3.2 SHELF Features

SHELF makes conducting large-scale MTurk studies much easier. Rather than needing to create individual tasks manually, SHELF creates correctly configured tasks with the click of a button. Then, as data is collected, it helps researchers monitor data. The only time that a researcher needs to go directly to MTurk is when adding more money into the MTurk account to pay participants. The remainder of this section discusses several of the features built into SHELF to help run experiments more quickly and easily.

3.2.1 Monitoring and Organizing Large-Scale Studies

Conducting a research study requires monitoring data collection. Each researcher has an account with a username and password – passwords are stored salted and hashed using bcrypt. A researcher who logs in sees a screen similar to that in Figure 3.1. The researcher can walk through the study as though a participant in any condition, which is useful for developing and testing. Researchers can view how many participants many have completed each step of a study by condition. They can also monitor how much money remains to pay participants. Most interactions with MTurk can be conducted through SHELF. This can help detect technical problems with data collection, and detect whether a condition is causing an unacceptable dropout rate. Once data collection, researchers can export data for analysis.

The screenshot shows the SHELFL feedback interface for a researcher named Rich. The top navigation bar includes 'SHELFL_feedback' on the left and 'Log Out' on the right. Below this, there are several menu items: 'Home Runs', 'About AMT Interactions', 'Rich Researchers Subjects', 'Control Panel Experiments', and 'Settings Passwords'. Underneath these, there are links for 'Experiments : feedback', 'Conditions', 'Collections', and 'Hits'. The main content area is titled 'feedback' and contains a sub-section 'Conditions'. A table lists ten conditions, each with a name, a status of 'Active', and four action buttons: 'edit', 'delete', 'Test', and 'Test2'. Below the table, there are two links: 'New Condition' and 'Edit'.

Condition Name	Status	edit	delete	Test	Test2
3c12	Active	edit	delete	Test	Test2
3c12-rt	Active	edit	delete	Test	Test2
3c12-plain	Active	edit	delete	Test	Test2
3c12-bl	Active	edit	delete	Test	Test2
3c12-bl-rt	Active	edit	delete	Test	Test2
3cp12-rt-10splice2	Active	edit	delete	Test	Test2
3cp12-rt-10guide2	Active	edit	delete	Test	Test2
3cp12-rt	Active	edit	delete	Test	Test2
3cp12	Active	edit	delete	Test	Test2

[New Condition](#)
[Edit](#)

Figure 3.1: The screen a researcher, in this case Rich, sees upon logging into SHELFL. He can view, delete, and test different conditions. “feedback” is the internal name of the study presented in Chapter 8. There are also links to screens containing the different data in the study, such as the subjects.

3.2.2 Preventing Repeat Participants

SHELFL can be configured to exclude prospective participants based on several criteria. Researchers can prevent participants from taking a study more than once, or taking more than one study. The studies in this thesis were between-subjects studies, and used statistical tests that assumed each data point was independent. Amazon assigns each MTurk account a unique *workerid*. To facilitate preventing participants from taking more than one of a group of studies, SHELFL can import and export sets of workerids. That way, one study using SHELFL can import the list of workerids from a prior study in order to detect and reject repeat participants. Because it may be possible to de-anonymize MTurk workers based on their workerid, these lists can be encrypted. As an additional measure to detect repeat participants, SHELFL can place cookies on participants’ machines.

3.2.3 Keeping Participants from Viewing the Wrong Page

Participant taking a study on SHELFL view a sequence of web pages. By clicking on buttons or other indicators, they move to the subsequent page. Participants do not authenticate to SHELFL, but I did not want anyone to be able to stumble onto the wrong page by playing with URLs. Therefore, I implemented a check-sum system for each participant-facing page. Each Run (a participant’s course through a study) has a unique identifier number. Each user-facing page also has a unique identification number, and each study has a hash value. SHELFL hashes the Run id, page id, and salt together to generate a checksum. Anyone attempting to reach a participant-facing page without the correct checksum goes to an error page.

3.2.4 Conducting Participants Through the Study

SHELF is designed to conduct participants automatically through the course of a study. A prospective participant encounters the study on MTurk, in which HITs advertise paid tasks for workers. Once a participant has accepted the HIT, that participant receives a URL into SHELF. Each MTurk worker has a unique *workerid*, and this URL is custom-made to include that workerid. Sending this workerid lets SHELF record, and pay, the worker. SHELF assigns participants to their conditions round-robin, and can have different subject-facing pages depending on condition. As participants take the study, SHELF records their activity in the study database. After participants completed the study, SHELF automatically approved their HITs, ensuring they were compensated. Our study protocol required emailing participants to return for Part Two of the study. SHELF uses a cron job to contact participants and invite them to return with a customized URL.

The next chapter of this thesis discusses the data-collection protocol for the studies presented in this thesis. This data-collection protocol makes heavy use of SHELF. SHELF itself is designed to be flexible, and the following chapter describes only one possible use of the framework.

Chapter 4

Study Protocol

In this chapter, we describe our data-collection processes and the data we collect. This thesis presents data from five studies, each with a similar protocol. Each study chapter has an individual Methodology section describing its conditions and other specific factors. Section 4.1 describes the route that participants take through our studies. In Section 4.2, we describe the PCFG password-guessing algorithm that we used to calculate password guess numbers. We describe our security and usability metrics in Section 4.3. In Section 4.4, we present the statistical tests that we use in our studies. Section 4.5 discusses ethical considerations of our research. Section 4.6 discusses limitations of our methodology.

4.1 Data-Collection Protocol

We recruited participants on MTurk, as described in Section 3. All of our studies required that participants be at least 18 years old. The study we present in Chapter 6 allowed participants from anywhere, while the rest required participants be located in the United States. Participants could only take a study once, and could be in at most one study. In our first MTurk study, described in Chapter 6, we began by paying participants 25 cents for completing the first part of the study and 50 cents for completing the second part. We soon¹ increased this to 55 and 70 cents to increase participation, and we kept these payment amounts.

Figure 4.1 depicts our protocol. After recruitment on MTurk, we asked participants to complete a consent form. Then, they begin the initial task, which we call *Part One* of our study. We asked participants in all but one condition to imagine creating a new password for their email service provider. We used text matching or very similar to the following in each study:

Imagine that your main email service provider has been attacked, and your account became compromised. You need to use a new password for your email account, since your old password may be known by the attackers...We will ask you to use this password in a few days to log in again so it is important that you remember your new password. Please take the steps you would normally take to remember your email password and protect this password as you normally would protect the password for your email account. Please behave as you would if this were your real password!

¹We paid 984 participants the lower rates of 25 and 50 cents.

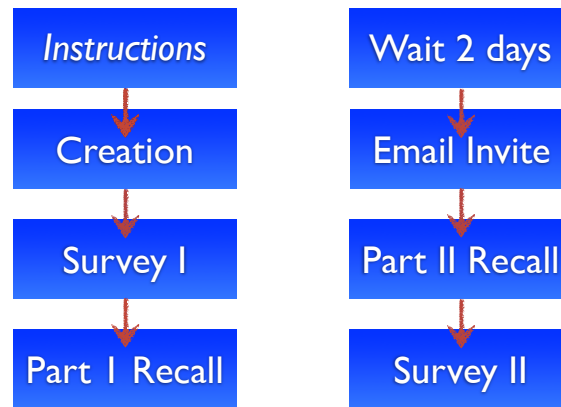


Figure 4.1: Out data-collection protocol. The first column represents Part One, linked to from MTurk. The second column represents part two, to which participants were invited through email two days after completing Part One.

As a function of their study condition, participants either created a password under a specific password-composition policy, or were assigned a password. After creating or being assigned a password, participants took a quick survey, as shown in the Appendices. We did this to learn about the participant’s sentiment regarding the password-creation process and as a distractor task. We then asked participants to enter their passwords, which we call Part One recall. If unsuccessful in five attempts, the participant saw the password on-screen. Successful recall ended Part One of the study, and we compensated the participant.

Two days after completing Part One of the study, we invited the participant to return for *Part Two* of the study. We used MTurk to email workers through their workerid, and we did not need to learn their email addresses. We asked returning participants to recall their passwords, which we call Part Two recall. On the fifth failed attempt, participants saw the password on the screen. Participants could click on a “Forgot My Password” link to be emailed a link to their password.

Participants then took a second brief survey. This asked about the password-recall process, sentiment related to password recall, and whether and how participants wrote down or otherwise stored their password. To set a cutoff for our own data collection, and to consider only recall from participants who had seen their password within five days, participants who finished Part Two more than three days after Part One were compensated but excluded from our analyses.

4.2 Generating Guess Numbers

As described in Section 2.3.2, we used *guess numbers* to compare the strength of passwords. To generate these guess numbers, we used a Probabilistic Context-Free Grammar (PCFG) password-guessing algorithm. Weir et al. first devised the concept [98], and our passwords research group refined the algorithm in subsequent research [45].

The PCFG algorithm constructs a probabilistic context-free grammar from training data, such as wordlists or other password corpora. Probabilities are learned for overall password structures (sequences of character classes) and learned separately for substrings of a single character class. These structural and substring probabilities are combined into probabilities for potential password

guesses, and the algorithm guesses passwords in probabilistic order. Computational limits generally require a cutoff before the entire space has been guessed [98].

Our lab continued to improve the PCFG algorithm over the course of the studies in this thesis. Rather than using the PCFG algorithm to generate a full list of guesses and then matching study passwords against that list, in [45] our lab introduced the concept of a *guess number calculator*. Given a password, the guess number calculator determines or estimates the number of guesses the PCFG algorithm would need to guess that password.

Algorithm 1 describes the creation of a lookup table for the guessnumber calculator. We adopt Weir et al.’s terminology of *structure* for a string representing the character classes of a given password. For example, the structure of password *Password1!* is *UlllllllIDS*. A *terminal* is an instantiation of a structure, and a *probability group* is a set of terminals with the same probability [45].

To find the number of guesses required for PCFG to guess a given password, we use the lookup table created by Algorithm 1 to determine how many guesses would be made before that password’s probability group would begin to be guessed. Because all passwords in a given probability group are guessed in deterministic order, it is straightforward to calculate the guess number of the password itself. Creating this lookup table is time-intensive and resource-intensive, so we set a cutoff for each study – around 10^{12} to 10^{14} guesses.

Algorithm 1 This is the training algorithm for the PCFG algorithm to create a lookup table. Given a password, we looked up the number of guesses required to get to that password’s probability group. We then calculated how many additional guesses were required to guess the password itself. An *l.c.s.* is a *longest common substring*, the longest substrings in a probability group made from characters of the same character class. For example, for *UULL9UUU*, the *l.c.s.*’s would be *UU*, *LL*, *9*, and *UUU*.

```

 $\mathcal{T}$  = New Lookup Table
for all structures  $s$  do
  for all probability_group  $pg \in s$  do
    for all  $l.c.s. \in pg$  do
       $c_i$  = Number of terminals of  $l.c.s.$ 
       $p_i$  = Probability of  $l.c.s.$  in training data
    end for
     $probability = \prod p_i$ 
     $\mathcal{T}.add: probability, pg, \prod c_i$ 
  end for
end for
Sort( $\mathcal{T}$ ) by probability
Add to each value in ( $\mathcal{T}$ ) the sum of prior values

```

4.2.1 PCFG Training Data and Word Lists

The PCFG algorithm requires training data, generally corpora of known passwords. The training data for password structures can differ from that used for password substrings. This helps to tune the PCFG algorithm for guessing specific sets of password requirements. For example, if a policy

requires eight characters and three character classes, we can include only passwords matching these requirements in the structural training data.

Our security results simulated an attacker who was aware of the password-composition requirements for each study condition. We trained the PCFG algorithm separately either for each condition, or for each set of passwords sharing structural requirements.

We trained the PCFG algorithm using both publicly available data sets and study data. We used study data for training the PCFG algorithm by using two folds. We randomly split the passwords being cracked together into two halves, and used each half as training data to crack the other half, along with public data. We then recombined the halves, yielding guess-number results for each password.

The PCFG algorithm itself evolved during the course of the studies presented in this thesis, and so comparing guessing results across studies may be of limited value. In some of our studies, in order to facilitate cracking longer passwords, we used a word list from Google [10]. Some of our later studies also used passwords from prior studies as PCFG training data. However, within a given study, we used the same version of the PCFG algorithm with analogous training data for each condition.

We used six publicly available word lists as training data for PCFG algorithm, as well as to construct dictionary checks for some of our conditions. The *RockYou* password set [19] includes more than 30 million passwords, and the *MySpace* password set [75] contains about 45,000 passwords. Both sets resulted from password breaches. The *inflection list*² contains words in varied grammatical forms such as plurals and past tense. The *simple dictionary* contains about 200,000 words and is a standard English dictionary available on most Unix systems. We also used two dedicated cracking dictionaries from the Openwall Project³ containing standard and mangled versions of dictionary words and common passwords. The *free Openwall list* contains about 4 million words, and the *paid Openwall list* contains more than 40 million words. These data sources are publicly available, and we expect attackers would train with readily available word lists.

4.3 Study Metrics

This section discusses the security and usability metrics we collected and analyzed.

4.3.1 Password Strength

Password strength is important for both system administrators and passwords researchers. Section 2.3 discussed prior work on password strength, including the arguments for using guess numbers to measure password strength. Section 4.2 described how we calculated guess numbers with the PCFG algorithm. Guess numbers let us compare how passwords in each condition perform against an increasing number of guesses. We plot the percentage of guessed passwords in each condition against the number of guesses that have been made, in log scale. In addition, we provide entropy estimates for passwords in some of our studies (as described in Section 2.3.1), as well as descriptive statistics such as usage of different character classes.

²<http://wordlist.sourceforge.net>

³<http://www.openwall.com/wordlists/>

Our security model is an offline attack in which the attacker has access to passwords in hashed (salted or unsalted) form. We simulated a larger number of guesses – often over 10^{10} . The actual time needed to reach that number of guesses depends on the attacker’s hardware and the hashing function being used. To provide a rough idea of how guess numbers translate to time, we performed benchmarking tests in our lab. We used oclHashcat-1.21 on a top-of-the-line graphics card⁴ to perform the deliberately slow Blowfish⁵ hash (configured with a cost factor 05), on which the password-hashing scheme bcrypt⁶ is based. We performed 6,874 hashes per second. This means that an attacker using similar hardware would reach 10^8 guesses after about 4.0 hours. The attacker would reach 10^{12} guesses after about 4.6 years, but this could be reduced by dividing the work among multiple cards. A more in-depth discussion of translating guess numbers into time is in Section 2.1.

4.3.2 Password Usability

We examine usability for password creation, and recalling passwords after a few minutes and a few days. We ask participants questions in order to understand their perceived sentiment. We also record observed metrics, including attempts needed and time taken. We consider taking more attempts and more time to be indicative of increased difficulty. We describe these metrics in more detail below.

Password Creation or Assignment

Password-creation usability is especially important for real-world systems in which a frustrated or annoyed user can easily abandon the service. It may be less important for universities or offices where users are less likely to leave because creating a password is unpleasant.

- **Creation attempts:** This is the number of attempts participants needed to create a password complying with the requirements. For system-assigned passwords, it is the number of attempts needed for the participant to enter the password correctly. We consider a participant requiring more attempts to create a password to indicate password creation being more difficult or taking more effort.
- **Password-creation sentiment (difficult, annoying, fun):** We asked participants whether they agreed with creating or learning their password being difficult, annoying, and fun as three separate Likert questions. This measures self-reported user sentiment for password creation.

Part One Recall

After creating their passwords, we gave participants a short survey. Then, we asked them to recall their password. We measured how accurately and quickly they were able to do so. Short-term password-recall usability is relevant in a number of real-world scenarios, such as websites that ask users to create a new password and then to sign in with that password. Difficulty in this step might

⁴AMD Radeon R9 290x with stock clock frequencies

⁵<https://www.schneier.com/blowfish.html>

⁶<http://bcrypt.sourceforge.net>

deter usage. Further, short-term recall usability helps to align our findings with those of past and future human-subjects laboratory studies of password policy. As discussed in Section 2.5.4, these laboratory studies are often limited to studying password recall soon after creation.

- **Recall attempts:** This is the number of attempts needed to recall the password shortly after creation. Conditions that took more attempts to recall are considered less memorable in the short term.
- **Recall time:** This is the average time taken to enter the password, typically measured in median seconds. We consider shorter recall times to signify the passwords being easier to recall and type.

Part Two Recall

Two days after participants finished Part One of our study, we invited them to return for Part Two. Upon returning, we asked them to recall their password. Part Two recall represents recalling the password after between two and five days have passed since creating it. This is relevant to service providers concerned about retaining users, as well as to those who need to pay help-desk costs for helping users who forgot their password. For researchers, it provides insight into recall over a larger time period than a typical laboratory study.

- **Recall attempts:** This is the number of attempts participants took to recall their passwords. After five failed attempts, we showed their password on the screen. More attempts may indicate more recall difficulty, or recall taking more effort.
- **Recall time:** This is how long participants took to recall their passwords, in median seconds. In most studies, we looked only at participants who recalled their password on the first attempt. Less time is considered more usable.
- **Password-recall sentiment (difficulty):** We asked participants whether they agreed with password recall being difficult, as a Likert question. This measures self-reported password-recall difficulty.
- **Password reminder usage:** Participants could use a password reminder to be emailed a link to view their passwords on the screen. Use of this reminder might indicate participants finding it difficult or frustrating to recall their passwords. It might also indicate participants perceiving that recalling their passwords would take more effort.

Further Usability Metrics

- **Part One dropout:** We consider a participant who has completed the consent form, but who did not finish Part One of the study, to have dropped out. We consider a higher dropout rate to indicate usability difficulty, because dropping out may signify that the participant despaired of recalling the password, or was unable to make a satisfactory password that met the condition. This metric is relevant to both researchers and service providers, as the former want more participants and the latter want more users.

- **Password storage:** We consider a participant who has stored his or her password to have had a more difficult time with it, or to have anticipated having a more difficult time recalling it. We only consider password storage for participants who completed Part Two of the study. We consider a participant to have stored his or her password if we detected that participant pasting or autofilling the password. We also consider a participant to have stored his or her password if that participant did not indicate otherwise in the Part Two survey. This metric is relevant to researchers because it puts other recall metrics into context. For example, password storage might explain why participants reported less difficulty with recalling a difficult password.

4.4 Statistical Testing

Unless otherwise noted, our statistical tests use a significance level of $\alpha = .05$. We compare metrics between study conditions. Some studies compare each condition to each other condition; other studies compare only certain conditions to each other. When we compare metrics for conditions, we distinguish between quantitative and categorical variables. When conducting a quantitative comparison across conditions, we first conduct an omnibus comparison using Kruskal-Wallis (KW). This is an analogue of ANOVA that does not assume normality. If this test is significant, we perform pairwise Mann-Whitney U (MW) tests with Holm-Bonferroni correction (HC). In order to test for differences in categorical variables across conditions, we first perform an omnibus χ^2 test. If that shows significance, then for each pair of conditions we are testing, we perform Fisher's Exact Test (FET) with HC correction.

4.5 Ethical Considerations

The studies in this thesis were approved by the Carnegie Mellon University Institutional Review Board. Our participants consented to take part in our research and received a researcher's contact information. They all affirmed that they were at least 18 years old. In order to train our password-cracking algorithm, we used publicly available stolen password sets. These were widely known and circulated before we used them, and many had already been used in other research [21,97,98]. We contend that because the data are already widely available, our use of them constitutes no further harm to the victims. We did not collect passwords alongside usernames or other login data. Further, because attackers likely were already using these passwords for their own purposes, our use of them to study password-composition policies is especially important if we want to help security administrators keep pace with attackers.

4.6 Limitations

This section discusses limitations of our study protocol.

4.6.1 Ecological Validity

The ecological validity of our research is important to the generalizability of our results. The MTurk population is younger and better-educated than the general population, and is more diverse

than is typical for laboratory studies [14, 41]. Previous research on using MTurk for research has found that it can produce high-quality data [4, 22, 37, 46, 93]. Using MTurk allowed us to collect a very large sample size under controlled conditions. Traditional in-person data collection would not have been a feasible way to accomplish this. Mechanical Turk enabled much larger sample sizes for our studies, allowing our statistical tests to have more power. We were also able to collect data across a much larger geographic area than would have been feasible using in-person laboratory studies.

The passwords in our studies did not protect high-value accounts. To help participants think more as they would in reality, we asked them to imagine that they were creating passwords for their primary email account. In the study described in Chapter 6, we had one condition in which we asked participants instead simply to create a password for a study. Participants in that condition created significantly weaker passwords. This is evidence that by asking participants to imagine creating a real password, they were more likely to buy into our study.

Within each of our studies, we took care to differentiate the experiences of our participants only in ways related to their differing conditions. This means that the differences observed between participants in each condition can be attributed to the differences between those conditions. In most cases, this is only a difference in their password-composition policies. In this way, even if the study being conducted on MTurk made participants behave differently than they would in real life, we believe that the comparisons between the conditions would still be valid because all conditions were run on MTurk.

In Section 2.5.6, we describe a study conducted by our passwords research group. In this study, we collected metrics about genuine passwords used by the students, faculty, and staff of Carnegie Mellon University. We also asked MTurk workers to create passwords under the same requirements, including a replication of the password-creation screens seen by genuine users. This enabled us to contrast the passwords created by MTurk workers with genuine passwords made under the same requirements and branding. We found that while the MTurk passwords were slightly weaker than the genuine passwords, they were very similar and shared many characteristics. This demonstrated that MTurk workers create passwords that are in many ways similar to those of genuine users, supporting the ecological validity of MTurk passwords research [58].

4.6.2 Remote Data Collection

Remote data collection has both advantages and disadvantages compared to traditional in-person human-subjects data collection. The research in this thesis would not have been feasible without using online crowdsourcing. MTurk allowed us to recruit far more participants than would have been possible using traditional methods. We recruited participants across a broad geographic space, instead of using only local participants. Further, while traditional live human-subjects data collection requires a researcher be present with one or a few participants, our data collection allowed participants to take the study independently.

While this remote, automated data collection enables many more participants than traditional methodologies, it has some downsides. When a live researcher sees a participant struggling with something in a study, this can be noted and reported. A live researcher can also ask a participant to speak aloud to collect more information, and ask spontaneous questions that come up during the study. These are not feasible through our methodology. This shortcoming can be at least somewhat mitigated through live in-person study piloting.

4.6.3 Other Limitations

Advances continue in the domain of password-cracking algorithms. While a study of these algorithms is not a focus of this thesis, we do use them to contrast password strength between conditions in a study. There may exist yet-undiscovered password-cracking algorithms that would produce different password-strength results. Further, advances in hardware may facilitate attackers making more guesses than our research considers.

Our research focuses on specific security and usability metrics, as outlined in Section 4.3. Other metrics that may be useful are not recorded. For example, while we examine password recall after a few minutes and a few days, many password use-case scenarios involve password recall after a longer period of time.

Our security model considers how well passwords resist an offline attack. There are other threats to passwords, such as shoulder-surfing and phishing, in which password strength is not a factor.

Chapter 5

How Usable are System-Assigned Passphrases?

5.1 Introduction

Allowing users free rein to create their own passwords often leads to weak, easily guessed passwords [5, 71, 100], resulting in security breaches and loss of privacy for victims [18]. Many organizations attempt to address this problem using password-composition policies, which limit the password-creation space in an effort to prevent users from choosing passwords that are too easily guessed [16]. Unfortunately, strict password-composition policies sometimes lead to user frustration without substantial security benefit [1, 39]. Also, even under a strict policy, users may fulfill policy requirements in predictable ways [90], such as basing their passwords on older passwords, names, or words [88, 100], or reusing passwords across domains [26].

The other studies in this thesis examine password-composition policies in which users create passwords under certain requirements. These requirements, such as requiring that user-created passwords have at least sixteen characters, are intended to prevent users from creating easily guessed passwords. This study takes a different approach to ensuring the guessability of passwords. We assign system-created passwords to participants, removing their choice about passwords. This lets us ensure that each system-assigned password has a level of security that would be difficult to ensure with user-created passwords.

We assign participants both short passwords and longer passphrases. A *passphrase* is a password composed of a sequence of words. Passphrases are typically much longer than ordinary passwords, and proponents argue that they are more secure and easier to remember. One NIST publication states that “any long password that can be remembered must necessarily be a ‘pass-phrase’ composed of dictionary words” [16]. The use of passphrases has recently garnered appreciable attention [64, 77], and some institutions have adopted passphrases as a password policy (e.g., [99]). Despite this recent interest in passphrases, however, there is little empirical evidence to support claims of superior usability over passwords.

This chapter describes the results of a 1,476-participant study on the usability of system-assigned passphrases and system-assigned passwords. The passphrases we study are sequences

This chapter is largely a reproduction of [86], co-authored with Patrick Gage Kelley, Saranga Komanduri, Michelle Mazurek, Blase Ur, Timothy Vidas, Lujó Bauer, Nicolas Christin, and Lorrie Faith Cranor.

of three or four English words drawn at random from a set dictionary and separated with spaces. Our passwords are also system-assigned and are five to eight characters in length. While the other studies in this thesis focus on user-created passwords, this study is limited to system-assigned passwords. We control for guessability and contrast usability for different password-assignment policies.

Our findings suggest that system-assigned passphrases are far from a panacea for user authentication. Rather than committing them to memory, users tend to write down or otherwise store both passwords and passphrases when they are system-assigned. When compared to our password conditions, no passphrase condition significantly outperformed passwords in any of our usability metrics, indicating that the system-assigned passphrase types we tested fail to offer substantial usability benefits over system-assigned passwords of equivalent strength. We even find that system-assigned passphrases might actually be less usable than system-assigned passwords. For instance, users were able to enter their passwords more quickly and with fewer errors than passphrases of similar strength.

While our results in general do not strongly favor system-assigned passwords over system-assigned passphrases or vice versa, we identify several areas for further investigation. For example, larger dictionary sizes do not appear to have a substantial impact on usability for passphrases. This could be leveraged to make stronger passphrases without much usability cost. We also find that lowercase, pronounceable passwords are an unexpectedly promising strategy for generating system-assigned passwords.

Researchers have proposed error-corrected passphrase systems [3, 42, 60]. Our results suggest that sophisticated error correction, such as mapping the word a user enters to the closest word in the passphrase dictionary, is necessary to make passphrases comparably usable to passwords. Without error correction, many passphrase conditions perform significantly worse than our password conditions.

We next discuss our methodology in Section 5.2. We present results on usability, accuracy, and sentiment in Section 5.3, and describe our error analysis in Section 5.4. We consider our findings in Section 5.5.

5.2 Methodology

Our data-collection protocol is described in Section 4.1. We used an instance of the SHELF framework, which is discussed in Chapter 3. We limited our data collection to participants from the United States because our passphrases are constructed using words from American English. Because we used only system-assigned passwords in this study, we did not generate guess curves. Instead, the security of each condition against guessing attacks by someone who knows the password-generation algorithm can be calculated a priori.

In this study, we use the term “secret” to mean either a password or a passphrase. Instead of being asked to create a password, participants were assigned a secret based on which of the 11 conditions they were assigned. After being shown the secret, participants were required to check a box on the screen to hide the secret and then enter that secret twice, once as confirmation. They could uncheck the box to see their assigned secret again, but could not type while their secret was visible.

Condition name	Entropy (bits)	Length	Dictionary size	Example
<i>Password Conditions</i>				
pw-length5	30	5 characters	64 characters	@J#8x
pw-pronounce	30.2	8 characters	190 syllables	tufritvi
pw-length6	36	6 characters	64 characters	R6wy\$_
<i>Passphrase Conditions</i>				
pp-small	29.4	4 words	181 words	one between high tell
pp-med-unorder	29.3	4 words (any order OK)	401 words	help any country our
pp-large-3word	29.4	3 words	1,024 words	own decide some
pp-nouns	30	4 nouns	181 nouns	sense child reason paper
pp-nouns-instr	30	4 nouns (w/ instructions)	181 nouns	phone star record right
pp-sentence	30	4 words (N-V-Adj-N)	181 words each	end determines red drug
pp-medium	33.9	4 words	401 words	also that research must
pp-large	39.2	4 words	1,024 words	because strategy cover us

Table 5.1: A summary of experimental conditions, with data about their characteristics and example secrets assigned to participants.

5.2.1 Conditions

We assigned participants round-robin to one of 11 conditions, which are summarized in Table 5.1. The conditions varied in the type of secret assigned to the participant. Participants were unable to modify their assigned secret or to obtain a replacement. We focused on system-assigned secrets so that we could precisely control their entropy (and their guessability), and focus on their usability.

Three conditions were variants of passwords, and eight were variants of passphrases. Our password conditions did not use spaces. In the passphrase conditions, we required that participants enter words separated by spaces and in the same order they were assigned, unless otherwise specified. Secrets in all 11 conditions were case-sensitive.

We designed two of our three password conditions and six of our eight passphrase conditions to have approximately 30 bits of entropy so that we could compare system-assigned passwords to equally strong system-assigned passphrases. This entropy value was chosen because guidelines frequently used in practice [16, 38] recommend password policies that provide an estimated 30 bits of entropy. While research has suggested that entropy may not be the best indicator of resilience to attack [45, 97], when all elements from a set occur with equal probability (as is the case with our system-assigned secrets), entropy maps directly to the probability that an attacker with knowledge of the password-generation algorithm will successfully guess a password.

5.2.2 Password conditions

Three of our conditions focused on passwords.

- **pw-length5**: Participants were assigned a five-character password, where each character is chosen randomly from a dictionary of 64 characters, including lowercase letters, uppercase

letters, digits, and symbols. We removed characters that could easily be confused with other characters, e.g., both the letter “O” and the digit “0.” This password space has 30 bits of entropy.

- **pw-pronounce:** Participants were assigned an eight-character password likely to be pronounceable by an English speaker. To generate these passwords, we used an implementation¹ of an algorithm originally proposed by Gasser [30] and later adopted as a NIST standard [65]. Prior work has identified a flaw in this scheme: certain passwords are chosen with high probability since the probability of a syllable occurring in a password mirrors its relative frequency in English [53]. To overcome this, we generated the full list of eight-character pronounceable passwords without duplicates (≈ 1.2 billion) and assigned each password on this list with equal probability, resulting in 30.2 bits of entropy.
- **pw-length6:** Participants were assigned a six-character password, where characters are chosen as in the pw-length5 condition. The extra character makes passwords in this condition have 36 bits of entropy. This condition helps determine how the length of randomly generated passwords affects usability.

5.2.3 Passphrase conditions

We tested eight variations on passphrases. We generated dictionaries for all passphrase conditions using word-frequency data² with part-of-speech (e.g., noun, verb) tags from the Corpus of Contemporary American English (COCA) [20]. This list ranks the most common words in the 425-million-word COCA based on the number of times they appear and their diffusion throughout different sources. So that our dictionaries would contain only well-known words, we chose the N most common words matching particular criteria for each dictionary. For instance, a dictionary of 181 nouns would contain the 181 most common nouns from COCA.

We selected word lists of particular sizes so that different conditions would each have 30 bits of entropy. However, we later discovered that word lists not restricted to a particular part of speech contained duplicate words. For instance, “to” was present on the list as both an infinitive marker and as a preposition. Thus, the actual passphrase entropies in the next three conditions, intended to be 30 bits, were as low as 29.3 bits.³

- **pp-small:** Participants were assigned four words, randomly selected with replacement from a 181-word dictionary.
- **pp-med-unorder:** Participants were assigned four words, randomly selected with replacement from a 401-word dictionary. Unlike all other conditions, participants could enter the words in their passphrase in any order.
- **pp-large-3word:** Participants were assigned three words, randomly selected with replacement from a 1,024-word dictionary.

¹<http://www.adel.nursat.kz/apg/> (visited 5/2012)

²<http://www.wordfrequency.info/top5000.asp> (visited 5/2012)

³All entropies were calculated using Shannon’s formula on the frequency distribution of unique words [80].

The next two conditions are similar to the pp-small condition, except they use larger dictionaries of the most common words in order to test whether the size of the dictionary has a measurable impact on usability:

- **pp-medium:** Participants were assigned four words, randomly selected with replacement from the 401-word dictionary used in the pp-med-unorder condition. These passphrases present 33.9 bits of entropy.
- **pp-large:** Participants were assigned four words, randomly selected with replacement from the 1,024-word dictionary used in the pp-large-3word condition. These passphrases present 39.2 bits of entropy.

We also tested whether passphrases that followed certain part-of-speech patterns aid memorability. The next three conditions use passphrases with 30 bits of entropy.

- **pp-sentence:** Participants were assigned passphrases of the form “noun verb adjective noun,” where nouns, verbs, and adjectives are chosen from separate 181-word dictionaries. So that it would make sense for the verb to be followed by a noun, the verb dictionary contained only verbs whose entry in The Free Dictionary⁴ listed at least one transitive definition. Since all nouns but one were singular, we manually conjugated all verbs to agree with a singular subject. Although these passphrases were unlikely to make semantic sense due to the random selection of words, they might resemble English sentences.
- **pp-nouns:** Participants were assigned four nouns, randomly sampled with replacement from a dictionary containing the 181 most common nouns.
- **pp-nouns-instr:** The condition is identical to pp-nouns, except that we gave the participant specific instructions for memorizing the passphrase. The instructions asked participants to “try to imagine a scene that includes all of the words in your password phrase. This will help you to remember it more easily. Research has found that the more bizarre, unusual, and exaggerated you make your scene, the easier it will be to remember. So, take a moment to construct your scene, and think about it whenever you need to enter your password.” This specific instruction mimics the example from the xkcd comic [64].

5.2.4 Statistical Testing

In our pairwise tests, we compared a subset of all possible pairs of conditions. All eight of our 30-bit conditions are compared to each other. We also compare pp-medium with pp-med-unorder, because they both use a 401-word dictionary; and pp-large with pp-large-3word since both use 1,024-word dictionaries. We compare pw-length5 with pw-length6 as the latter uses longer passwords, but they are otherwise identical. Finally, we compare pp-medium with pw-length6 to compare a password and a passphrase condition with higher entropy.

In addition to looking at conditions independently, we sometimes combine a subset of our password conditions and a subset of our passphrase conditions to compare larger sample sizes of passwords and equivalent-entropy passphrases. The *combined passwords participants* comprise

⁴<http://www.thefreedictionary.com/> (visited 5/2012)

Condition	Storage (%)	Mean assignment attempts	Agree creation attempts	Agree creation difficult (%)	Agree creation annoying (%)	Agree creation fun (%)	Part One recall attempts	Part One recall time (med s)	Part Two recall attempts	Part Two recall time (med s)	Used reminder
pw-length5	75.9	1.3	22.0	42.4	9.4	1.1	3.4	1.3	4.0	18.3	
pw-pronounce	72.2	1.1	20.3	38.0	19.3	1.1	3.1	1.3	3.3	24.6	
pw-length6	80.8	1.5	44.2	61.5	4.8	1.2	4.2	1.4	5.5	17.3	
pp-small	76.6	1.2	14.4	37.8	9.0	1.5	5.3	1.6	5.3	20.7	
pp-nouns	69.7	1.5	25.0	46.8	14.4	1.3	7.4	1.5	8.4	24.5	
pp-nouns-instr	63.4	1.5	20.4	38.7	25.7	1.3	7.4	1.5	8.6	26.7	
pp-sentence	71.3	1.3	19.1	43.6	18.1	1.3	7.7	1.5	9.0	24.5	
pp-med-unorder	76.4	1.3	15.1	38.7	17.0	1.6	6.1	1.4	6.5	29.2	
pp-large-3word	72.8	1.3	1.9	32.0	8.7	1.5	4.7	1.3	5.1	21.4	
pp-medium	66.3	1.4	16.8	33.7	20.8	1.4	6.5	1.4	6.6	34.7	
pp-large	75.0	1.3	18.0	48.0	12.0	1.4	7.4	1.5	7.8	24.0	

Table 5.2: This table contains data for the analyses in this chapter.

our two 30-bit password conditions, pw-length5 and pw-pronounce. The *combined passphrases participants* comprise our 30-bit passphrase conditions that use 181-word dictionaries: pp-small, pp-sentence, pp-nouns, and pp-nouns-instr.

5.3 Results

In this section, we present the results of our study. We begin by discussing participant demographics in Section 5.3.1. We then look at drop-out rates per condition in Section 5.3.2, as higher drop-out rates may indicate participants are struggling more in those conditions. In Section 5.3.3, we discuss participants storing their assigned secrets. We examine how well participants were able to enter their secrets immediately upon assignment in Section 5.3.4. Section 5.3.5 discusses part-one recall rates, and Section 5.3.6 part-two recall rates. We further investigate usability by examining user sentiment in Section 5.3.7. A summary of results metrics is in Table 5.2.

5.3.1 Demographics

2,689 participants began our study in February and early March 2012. 2,294 completed the first part of our study and 1,562 returned for the second part within three days of being sent an email invitation two days after completing the first part. An additional 88 participants returned for the second part between three and 42 days after completing the first part; we do not include them in our analysis. Of the participants who returned within three days, 1,476 completed the second part of our study. With the exception of our discussion of drop-out rates in Section 5.3.2, we focus on these 1,476 participants throughout our analysis.

Of the 1,468 participants who reported gender, 51.9% reported being female and 47.6% reported being male. The mean age was 31 years, while the median was 28. The standard deviation was 11.2, and our oldest participant reported being 74 years old. Of the 1,464 participants who reported their highest academic degree, 653 reported having at least a bachelor’s degree. Participants were asked whether they had degrees or jobs in “computer science, computer engineering, information technology, or a related field.” Of the 1,460 who answered, 263 answered in the affirmative. We found no statistically significant differences between our conditions in reported gender, age, or background and education.

Because using a keyboard on a mobile phone could impact a participant’s ability to enter his or her secret, we examined participants’ user-agent strings. For example, if a user-agent string contains “iPhone,” that is strong evidence that the participant is taking the study from an iPhone. Only 25 participants show evidence of this,⁵ and there were no more than four per condition.

5.3.2 Study Dropouts

Of the 2,689 participants who started our study, 2,294 finished the first part (85.3%); 1,562 participants returned within three days of receiving our email invitation to complete the second part of the study, and 1,476 of these completed the second part.

The proportion of participants who completed the first part of the study varied by condition ($\chi^2_{10}=28.288, p=.002$), with participants in the two 30-bit password conditions, surprisingly, most likely to finish. Completion rates for the first part ranged from 78.9% for pp-sentence to 90.4% for pw-pronounce. Significantly more participants finished the first day in pw-pronounce than in pp-nouns (HC FET, $p=.020$) and pp-sentence (HC FET, $p=.011$). More also completed the first day in pw-length5 than in pp-sentence (HC FET, $p=.035$). Combined password participants were more likely to finish than combined passphrase participants ($\chi^2_1=14.768, p<.001$).

The proportion of participants who returned for the second part of the study within five days after completing the first did not vary significantly by condition ($\chi^2_{10}=6.759, p=.748$), and neither did the proportion of these who finished the second part ($\chi^2_{10}=15.956, p=.101$). We also saw no significant difference between combined password participants and combined passphrase participants for returning within five days ($\chi^2_1=3.423, p=.064$) or finishing the second day ($\chi^2_1=0.015, p=.901$).

5.3.3 Password Storage

During the second part of the study, we asked participants if they wrote down their secrets, on paper or electronically. We consider a participant not to have stored his or her secret if the participant affirmed not writing it down, and we do not detect pasting or autofilling the secret. We labelled these participants, 410 of our 1,476 total (27.8%), as *no-storage*; other participants we call *storage* participants. The proportion of no-storage participants is low across conditions; it does not vary significantly by condition ($\chi^2_{10}=17.351, p=.067$), nor is it significantly different between the combined password participants and the combined passphrase participants ($\chi^2_1=2.444, p=.118$).

The no-storage participants are most relevant when evaluating the memorability of secrets. However, as users can and do store secrets for their real accounts, the behavior of storage partic-

⁵We searched the user-agent strings for: Android, iPhone, iPod, iPad, mobile, RIM Tablet, BlackBerry, Opera Mini, Windows Phone, SymbianOS, Opera Mobi, nook, Windows CE, smartphone, webOS, BREW.

ipants also provides useful insights. For some of our analyses, we look specifically at no-storage participants. In some cases, differences between conditions that were statistically significant when looking at all participants are no longer significant when looking only at no-storage participants. However, this may be due in part to the smaller number of no-storage participants.

Among the 72.2% of participants who stored their secret, 48.3% indicated writing it down on paper and 43.6% reported storing it electronically; 23% pasted their secret. A single participant may have done more than one of the above. We asked participants “If you wrote down or stored your password for this study, how is it protected (choose all that apply)?” Of our 1,066 storage participants, 21.9% did nothing to protect their passwords. 26.7% said they stored it on a computer or device used only by themselves, the most popular response. 24.5% stored the password in a room or office used only by that participant.

We also asked our participants about their real email passwords. 308 indicated referring to a written-down or stored password when logging in with their real email password, and 1,168 did not. We also asked if they had ever stored their real email password. 768 participants indicated never writing down their real email password, while 373 did so on paper and 430 electronically. This 52.6% of participants who did not store their real passwords is a significantly larger proportion than the 32.8% who indicated not storing their study secret ($\chi^2_1=114.287, p<.001$).

5.3.4 Assignment

After receiving instructions, our participants were assigned a secret and immediately asked to enter it. They could toggle between being able to view the secret and being able to enter it. This was intended to ensure that participants observed and were able to type their secret. We measured the number of attempts needed to enter the secret successfully, and the fraction of participants who correctly entered their secret on the first try. Significant results of statistical tests are shown in Table 5.3.

Overall, participants needed an average of 1.3 attempts to enter their secret and 78.7% entered it successfully on the first try. For both metrics, there was a significant difference across conditions. pw-pronounce secrets needed fewer attempts and were more likely to be entered on the first try than pp-nouns-instr and pp-nouns, and also needed fewer attempts than pp-sentence. In aggregate, combined password participants needed fewer attempts and were more successful in entering their secret on the first attempt than combined passphrase participants. Similar relationships hold for no-storage participants: pw-pronounce needed fewer attempts and was more successful on the first attempt than pp-nouns-instr. No-storage participants also show difference based on password length, with pw-length5 requiring fewer attempts than pw-length6.

5.3.5 Part-One Recall

We asked participants to recall their secret after completing a brief survey. Participants who could not recall their secret after five attempts were shown the secret on the screen. The vast majority of participants in each condition succeeded in entering their secret within five attempts, ranging from 92.5% in pp-med-unorder to 99.5% in pw-length5 and pw-pronounce. The significant results of our statistical tests are in Table 5.4.

The proportion of participants who correctly entered their secret on the first try varied significantly across conditions. A larger proportion in pw-length5 and pw-pronounce entered their

Success on first entry				<i>Omnibus</i> $\chi^2_{10}=28.026, p=.002$	
cond 1	%	cond 2	%	p-value	
pw-pronounce	90.4%	pp-nouns	73.4%	HC FET, $p=.001$	
		pp-nouns-instr	71.2%	HC FET, $p<.001$	
combined pw	84.9%	combined pp	74.5%	$\chi^2_1=14.233, p<.001$	
Success on first entry (no-storage)				<i>Omnibus</i> $\chi^2_{10}=27.021, p=.003$	
pw-pronounce	92.3%	pp-nouns-instr	67.1%	HC FET, $p=.027$	
Attempts needed				<i>Omnibus</i> KW $\chi^2_{10}=28.573, p=.001$	
cond 1	mean	cond 2	mean	p-value	
pw-pronounce	1.14	pp-nouns	1.45	HC MW, $U=14526, p<.001$	
		pp-nouns-instr	1.50	HC MW, $U=14367.5, p<.001$	
		pp-sentence	1.33	HC MW, $U=7473, p=.025$	
combined pw	1.23	combined pp	1.41	KW $\chi^2_1=14.979, p<.001$	
Attempts needed (no-storage)				<i>Omnibus</i> KW $\chi^2_{10}=28.888, p=.001$	
pw-length5	1.24	pw-length6	2.55	HC MW, $U=274.5, p=.037$	
pw-pronounce	1.08	pp-nouns-instr	1.53	HC MW, $U=1350, p=.025$	

Table 5.3: Statistically significant results for secret entry immediately upon assignment.

Success on first entry				<i>Omnibus</i> $\chi^2_{10}=34.936, p<.001$	
cond 1	%	cond 2	%	p-value	
pw-length5	94.8%	pp-small	81.1%	HC FET, $p=.008$	
		pp-med-unorder	80.2%	HC FET, $p=.007$	
pw-pronounce	94.7%	pp-small	81.1%	HC FET, $p=.009$	
		pp-med-unorder	80.2%	HC FET, $p=.007$	
combined pw	94.7%	combined pp	87.2%	$\chi^2_1=13.867, p<.001$	
success on first entry (no-storage)				<i>Omnibus</i> $\chi^2_{10}=26.558, p=.003$	
pw-pronounce	94.2%	pp-large-3word	64.3%	HC FET, $p=.033$	

Table 5.4: Statistically significant differences in success on the first try for Part One recall.

password correctly on the first attempt than in pp-small or pp-med-unorder. Combined password participants outperformed combined passphrase participants.

Among no-storage participants, significantly more in pw-pronounce entered their secret correctly on the first try than in pp-large-3word. Looking at participants who entered their secret within five attempts, we see omnibus significance between conditions, but pairwise tests reveal no significant differences.

We measured the time between the first and last keystroke on the first correct entry for participants whom we did not detect pasting or autofilling their secrets, and who entered the secret within five attempts. Results are in Table 5.5. Combined passphrase participants had a median time of 7 seconds, significantly more than combined password participants, with a median of 3 seconds. pw-length5 and pw-pronounce each performed significantly better than all of the 30-bit passphrase conditions, and pw-length6 performed better than pp-medium. pp-small and pp-

large-3word performed significantly better than pp-nouns, pp-nouns-instr, and pp-sentence; and pp-large-3word also outperformed pp-large.

5.3.6 Part-Two Recall

Forty-eight hours after finishing the first part of our study, we invited participants to return for the second part. Our analysis includes the participants who returned within 72 hours of being invited and completed both parts of the study. We find that a majority in each condition wrote down their secrets, and nearly half that did not store their secret clicked on the “Forgot Password” link. Upon returning, a participant was asked to recall his or her secret. Five incorrect entries resulted in the secret being shown on screen. How participants fared in entering their secrets is shown in Table 5.6.

Returning participants could click a link to be emailed a link to their secret. 48.8% of no-storage participants used this feature. The proportion does not vary significantly by condition ($\chi^2_{10}=11.992$, $p=.286$), nor between combined passphrase participants and combined password participants ($\chi^2_1=1.764$, $p=.184$). 210 of our 1,476 participants did not use the email reminder or store their secrets. Across all conditions, out of the 354 participants who used the email reminder, 197 (55.6%) made no attempt to recall their secrets and 87 (24.6%) made only one attempt before using the reminder.

We consider a participant to have succeeded in recalling his or her secret if he or she entered it within five attempts without using the reminder. Overall, 74.7% of our participants were successful, including 48.5% of no-storage participants and 84.7% of storage participants. There were no significant differences between conditions ($\chi^2_1=2.413$, $p=.120$), nor between combined password and combined passphrase participants ($\chi^2_1=2.618$, $p=.106$). There were also not significant differences for no-storage participants between conditions ($\chi^2_{10}=11.786$, $p=.3$). There was significant difference in how many participants succeed on the first attempt between conditions ($\chi^2_{10}=4.774$, $p=.906$).

For each participant on his or her first attempt at secret entry, we calculated the edit distance between what was entered and the assigned secret. We use the Damerau-Levenshtein edit-distance metric, which is the minimum number of insertions, deletions, substitutions, and adjacent transpositions required to transform one string into another [3]. This mean edit distance is shown in Table 5.8. It was less than one for each of the password conditions, and for passphrases it ranged between 1.12 for pp-large-3word and 2.96 for pp-nouns-instr. The median for each condition was zero. Edit distance did not vary significantly between conditions (KW $\chi^2_{10}=12.579$, $p=.248$), or for just no-storage participants (KW $\chi^2_{10}=10.407$, $p=.406$).

Looking at successful no-storage participants in the pp-med-unorder condition, six of nine entered the password in the same order as it was assigned. Overall, 68 out of 74 participants in pp-med-unorder entered the passphrase in the same order as it was assigned.

Another metric for usability was the use of *deletes* during secret entry. A delete may indicate a participant changing his or her mind about a secret while entering it. We counted each instance of one or more characters being removed from the secret-entry field as a single delete and recorded the number per secret-entry attempt for each participant. Deletions per condition are shown in Table 5.8. Looking only at participants who succeeded in entering their secret on the first try in part two, the mean for each password condition was less than one, while for passphrase conditions it ranged from 1.76 for pp-medium to 3.78 for pp-sentence. pw-length5 had significantly fewer dele-

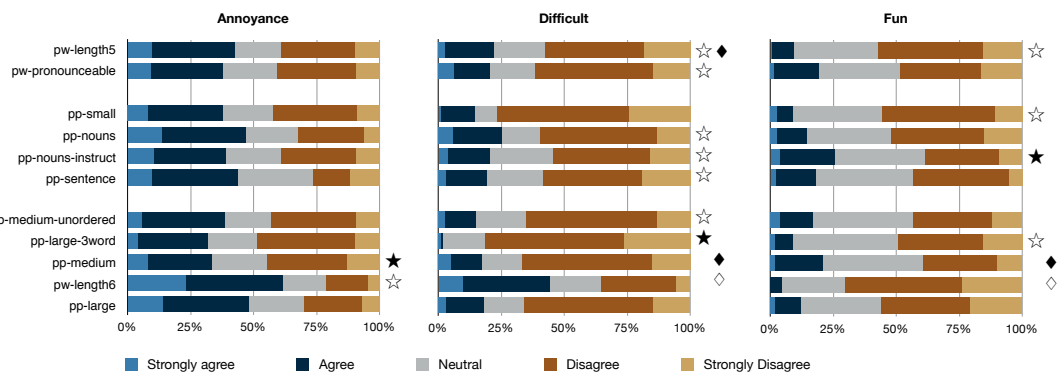


Figure 5.1: Likert response data on annoyance, difficulty, and fun. Data significance is shown in symbol groups; a condition marked with a solid symbol performs significantly better than conditions marked with the same symbol drawn as an outline.

tions than any 30-bit passphrase condition except `pp-small`, and `pw-pronounce` had significantly fewer deletions than `pp-nouns-instr` (HC MW, $p < .026$).

We also looked at *login* time, the total time a participant took to enter his or her secret, measured from the participant’s first arrival at the secret-entry screen until the end of the participant’s last visit to that screen. Login times by condition are shown in Table 5.8. The only significant pairwise difference for login time was `pw-pronounce` (median 25 seconds) performing significantly better than `pp-nouns` (median 35 seconds) (HC MW, $U=13716.5$, $p=.018$). Login time does not vary significantly by condition for no-storage participants (KW $\chi_{10}^2=15.79$, $p=.106$).

We next examined *entry time* the time between the first and the last keystroke on the initial correct entry for participants who neither pasted nor autofilled their secrets in the second part of our study, and entered their secret within five attempts. `pw-length5` and `pw-pronounce` each performed significantly better than most of the passphrase conditions, and `pp-small` and `pp-large-3word` outperformed most other passphrase conditions; these differences are shown in Table 5.7.

5.3.7 User Sentiment

In Part One, we asked participants to indicate their agreement, from “strongly disagree” to “strongly agree,” with the statements “learning my password was [fun/difficult/annoying].” We classify participants as either agreeing (“agree” or “strongly agree”) or not agreeing with each statement. An overview of results is shown in Figure 5.1, with detailed statistical results in Table 5.9.

We see a significant difference in annoyance, fun, and difficulty memorizing across conditions. Pairwise tests show that `pw-length6` was substantially more annoying, difficult to learn, and less fun than `pp-medium`, and was also more difficult than `pw-length5`. `pp-large`, and all 30-bit conditions, were more difficult to memorize than `pp-large-3word`. Finally, `pp-nouns-instr` was more fun by a wide margin than `pp-small`, `pp-large-3word`, or `pw-length5`. Comparing our combined password and combined passphrase participants, we see no significant difference in agreement that memorizing the secrets was annoying ($\chi_1^2=0.219$, $p=.639$), difficult ($\chi_1^2=0.022$, $p=.882$), or fun ($\chi_1^2=1.65$, $p=.199$).

5.4 Error Analysis Results

Examining factors that lead to user error can help us understand why passphrases were less successful than we anticipated, and can inform research on improving their performance. In addition, passphrases (and to a lesser extent pronounceable passwords) offer several opportunities for automatic error detection and correction, which may be able to improve usability without loss of entropy.

The overall results of our error analysis are shown in Table 5.10. This table displays the percentage of subjects who correctly entered their secret, both with no correction and adjusted for the use of different error-correcting mechanisms, as discussed below.

5.4.1 Length

We hypothesized that longer secrets (in characters) lead to more typing errors. Table 5.8 shows the mean length of secrets, per condition. For passphrase conditions, in which secrets are generally longer than in our password conditions, we find that longer passphrases reduce the likelihood of authentication success at assignment, but not thereafter. For all passphrase participants, we used logistic regression on passphrase length with an outcome of first-attempt success at assignment, and found a significant relationship ($p=.003$). The shortest passphrase condition, pp-small, had a mean length of 18.3 characters and a first-try success rate of 81% at assignment. By contrast, the longest condition, pp-sentence, had a mean of 25.5 characters and a first-try success rate of only 76%.

The same analysis for first-try accuracy for Part One recall (excluding participants who pasted or auto-filled their passphrase) and part two recall (excluding storage participants) found that length was not a significant factor in either case ($p>.375$). We also found no relationship between length and overall rate of part two recall success (within five attempts, without using the reminder, $p=.406$).

5.4.2 Ignoring Spaces and Capitalization

We required participants to enter their secret exactly as we showed it to them, including spaces and capitalization. In general, however, passphrase dictionaries can be designed to be case-insensitive and unambiguous even when spaces are removed. In addition, our pw-pronounce condition did not include uppercase letters. In such cases, removing spaces and ignoring case when checking input passwords can potentially improve usability with no cost to security. We examine how our passphrase and pw-pronounce participants would have performed had we ignored spaces and capitalization. We find that while error correction provides a small benefit, it does not cause passphrase performance to improve relative to passwords.

As shown in Table 5.10, ignoring case and spaces improves first-attempt accuracy for every passphrase condition as well as pw-pronounce, but has minimal impact on overall success within five attempts, on either Part One or part two. These improvements are small enough that they do not cause changes in the significance relationships among conditions.

Looking only at no-storage participants, however, we do see another difference. As reported in Section 5.3.5, during part-one recall significantly more participants in pw-pronounce entered their secret correctly on the first try than in pp-large-3word. This is still true with the correction

and, in addition, pw-pronounce also performed better than pp-small (HC FET, $p=.018$). Part-two recall continues to have no significant difference ($\chi^2_{10}=7.252$, $p=0.701$).

5.4.3 Off-by-One Errors

It is possible to construct a passphrase dictionary in which no word is within one edit of another. With such a dictionary, users who enter a word that is within one edit of the correct word in their passphrase can be authenticated successfully, with no loss of security. We did not attempt to create such a dictionary, but we did measure how many of our passphrase participants submitted entries with each word within one edit of the correct entry, as shown in Table 5.10.

Applying this correction narrows the gap between passwords and passphrases. For first-attempt success during day-one recall, we still see an omnibus significant difference among conditions ($\chi^2_{10}=18.463$, $p=.048$), but the pairwise differences showing greater accuracy for passwords than passphrases (see Section 5.3.5) disappear. Looking at only no-storage participants, however, pw-pronounce remains more successful than pp-large-3word (HC FET, $p=.033$). Recall attempts on the second day continue to show no significant variation among conditions, including when we examine only no-storage participants.

5.4.4 Closest Dictionary Word Correction

Our security analysis assumes the attacker knows the dictionaries used to generate passphrases, and would therefore never guess a non-dictionary word. As a result, if a passphrase participant enters a word not included in the dictionary for his or her condition, we can replace the entered word with the closest (by edit distance) dictionary word, with no loss of security. Ties are arbitrarily but consistently broken using word order within the dictionary. This correction can be applied only in our passphrase conditions and is case-insensitive; we did not implement it for our participants, but we examine how it would have affected their passphrase entries.

Results of our analysis are shown in Table 5.10. We find that this mechanism, like off-by-one correction, helps passphrase users somewhat but does not outperform uncorrected passwords. As with off-by-one correction, the only change we see in statistical relationships among conditions is for Part One recall; there remains an omnibus difference among conditions ($\chi^2_{10}=23.808$, $p=.008$), but there are no longer any pairwise differences. For no-storage participants, likewise, we see omnibus significance in Part One recall ($\chi^2_{10}=21.517$, $p=.018$), but no pairwise significance.

5.5 Discussion

We compared the usability of eight types of system-assigned passphrases and three types of system-assigned passwords using a number of metrics. This included memorability, time to authenticate, rate of user errors, rate of user storage, and user sentiment. In this section, we summarize our high-level results about passphrases and some surprising findings about pronounceable passwords. Several factors may affect the ecological validity and generalizability of our results. First, passphrases are unfamiliar to most users, whose behaviors and reactions might change given more experience. We would expect different behavior from users with self-selected passphrases, and users keeping track of multiple passphrases. Our memorability results are limited because so many participants stored their secrets.

System-assigned secrets. The use of system-assigned secrets eliminates the problem of users selecting low-entropy secrets, as well as the problem of users selecting a secret that they use for another account. We found that, in general, our system-assigned passwords and passphrases were not well-liked by users, and that the vast majority of users opted to store them. These results are consistent with the results of a previous study which found that 60% of participants stored their system-assigned 4-digit PINs [44]. In contrast, a study with similar methodology found that user-selected passwords were stored between 17% and 50% of the time, depending on condition [48]. Despite their unpopularity with users, system-assigned secrets may serve a role in situations where high entropy is a priority, and secure password storage poses minimal security risk and user inconvenience.

Dictionary choice. We used passphrases composed of words drawn from a variety of dictionaries. All of the dictionaries we used were generated from the most frequently used words in COCA. We found that whether we used a dictionary of the top 181 words, top 401 words, or top 1,024 words made little difference for the metrics we studied. Using the top 181 nouns, or a sentence-like combination of the top 181 nouns, verbs, and adjectives also made little difference. This suggests that we may be able to create high-entropy passphrases using even larger dictionaries without losing usability. Further, some dictionaries are better-suited to implementing error correction, such as a dictionary of words that are all at least three edits apart.

Passphrase length. We found few differences between 3-word and 4-word passphrases. 3-word passphrases were shorter than 4-word passphrases drawn from the same dictionary, and therefore faster to type and resulted in fewer typing errors. But although 3-word passphrases were perceived as significantly less difficult to learn than several other conditions, the number of attempts needed to authenticate did not vary significantly. In addition, 3-word and 4-word passphrases with equivalent entropy are approximately the same length and result in similar typing speeds and error rates. For the conditions we studied, the number of characters in a passphrase appears to affect usability more than does the number of words.

Memory aids. We hypothesized that passphrases would be easier to remember if they were sentence-like, and that passphrases composed of nouns would be easier to visualize than passphrases composed of random words. However, we found that pp-sentence and pp-nouns resulted in slightly longer passphrases than pp-small (which contained short words such as: the, be, and, a, to), but otherwise performed similarly. We had also predicted that pp-nouns-instr would perform better than pp-nouns because the instructions would help people visualize and remember their passphrases. However, we found only small, statistically insignificant differences between these two conditions. Different instructions, such as guiding users to visualize their passphrase or construct a scene or story using words from their passphrase, could prove more effective.

Word order. Requiring users to enter the words in a passphrase in a prescribed order increases the entropy of the passphrase. We explored whether this entropy increase came at the expense of usability. The pp-med-unorder condition was the same as the pp-medium condition, except it did not impose order requirements. Contrary to expectations, we did not find any significant differences between these conditions, nor between the pp-med-unorder and pp-small conditions, which used different dictionaries to maintain equivalent entropy. However, we found that participants did reorder their passphrases. 8.1% of participants in the pp-med-unorder condition took advantage of the ability to reorder their passphrases when entering them in the second part of the study (33.3% if we consider only no-storage participants). In passphrase conditions that did not permit reorder-

ing, we found that 9.3% of passphrase entry errors were due to entering words in the wrong order. Thus, it appears that relaxing the order requirement may provide small usability gains, but these gains were not significant in our study.

Error correction. Our analysis of passphrase-entry errors suggests that usability could be improved by selecting dictionaries that allow automatic correction of entry errors while maintaining a desired entropy. Even with the dictionaries we used, capitalization errors could be corrected without loss of entropy, because no two words differed only in capitalization. For the ordered passphrase conditions, missing spaces could also be corrected without loss of entropy. And, if every word in a dictionary had an edit distance of at least three from every other word in the dictionary, then it would be possible to correct many common typos as well as some errors where users misremember a word in their passphrase as another word that sounds similar.

Pronounceable passwords. We compared system-assigned passphrases and passwords. Based on the negative sentiment and high storage rate associated with system-assigned passwords in a previous study [44], we were initially concerned that random-character system-assigned passwords might not provide a fair comparison. We looked for algorithms to generate relatively short but high-entropy system-assigned passwords that had characteristics that might make them more memorable. We found repeated mention in the literature of Gasser’s algorithm for generating pronounceable random passwords [30]. We discovered that pw-pronounce performed very well in accuracy and entry-speed during part-one recall. One advantage of the pw-pronounce condition seems to be that the passwords in this condition include character combinations that, even if marginally pronounceable, are all lowercase and relatively easy to type.

Part One Secret entry time				Omnibus KW $\chi_{10}^2=329.817, p<.001$
cond 1	median seconds	cond 2	median seconds	p-value
pw-pronounce	3.1	pp-small	5.3	HC MW, U=3296, $p<.001$
		pp-nouns	7.4	HC MW, U=3187.5, $p<.001$
		pp-nouns-instr	7.4	HC MW, U=2833, $p<.001$
		pp-sentence	7.7	HC MW, U=1310.5, $p<.001$
		pp-large-3word	4.7	HC MW, U=3626, $p<.001$
		pp-med-unorder	6.1	HC MW, U=2548.5, $p<.001$
		pw-length5	3.4	pp-small
pp-nouns	7.4			HC MW, U=4126, $p<.001$
pp-nouns-instr	7.4			HC MW, U=3652.5, $p<.001$
pp-sentence	7.7			HC MW, U=1792, $p<.001$
pp-large-3word	4.7			HC MW, U=4223.5, $p<.001$
pp-med-unorder	6.1			HC MW, U=3141.5, $p<.001$
pw-length6	4.2			HC MW, U=4857.5, $p=.039$
pw-length6	4.2	pp-medium	6.5	HC MW, U=1933, $p<.001$
pp-small	5.3	pp-nouns	7.4	HC MW, U=4717.5, $p=.009$
		pw-pronounce	3.1	HC MW, U=10608, $p<.001$
		pp-nouns-instr	7.4	HC MW, U=4328.5, $p<.001$
		pp-sentence	7.7	HC MW, U=2212.5, $p=.002$
		pw-length5	3.4	HC MW, U=9766, $p<.001$
pp-large-3word	4.7	pp-nouns	7.4	HC MW, U=4106.5, $p=.001$
		pp-nouns-instr	7.4	HC MW, U=3809, $p<.001$
		pp-sentence	7.7	HC MW, U=1914, $p<.001$
		pp-large	7.4	HC MW, U=2101.5, $p=.001$
pp-large	7.4	pp-large-3word	4.7	HC MW, U=4455.5, $p=.001$
pp-medium	6.5	pw-length6	4.2	HC MW, U=4790, $p<.001$
pp-nouns	7.4	pp-large-3word	4.7	HC MW, U=8011.5, $p<.001$
pp-nouns-instr	7.4	pp-large-3word	4.7	HC MW, U=8807, $p<.001$
pp-med-unorder	6.1	pp-nouns-instr	7.4	HC MW, U=4886, $p=.015$
pp-nouns-instr	7.4	pp-med-unorder	6.1	HC MW, U=8186, $p=.015$
pp-sentence	7.7	pp-large-3word	4.7	HC MW, U=4477, $p<.001$
combined pw	3.1	combined pp	7.0	KW $\chi_1^2=249.884, p<.001$

Table 5.5: Statistically significant differences in password-recall time for Part One. This was the time between first and last keystroke on the first correct entry for participants whom we did not detect pasting or autofilling their secrets, and who entered the secret within five attempts.

	No-storage			Storage		
	Partici- pants	Login in five tries	Login on first try	Partici- pants	Login in five tries	Login on first try
pw-length5	46	65%	57%	145	86%	81%
pw-pronounce	52	52%	48%	135	83%	77%
combined password	98	58%	52%	280	85%	79%
pp-small	26	42%	38%	85	86%	76%
pp-nouns	57	47%	42%	131	84%	76%
pp-nouns-instr	70	51%	37%	121	83%	78%
pp-sentence	27	41%	37%	67	87%	76%
combined passphrase	180	47%	39%	404	85%	77%
pp-med-unorder	25	36%	36%	81	80%	70%
pp-large-3word	28	57%	50%	75	85%	75%
pp-medium	34	35%	32%	67	81%	73%
pw-length6	20	45%	40%	84	92%	82%
pp-large	25	44%	40%	75	85%	73%
total	410	49%	42%	1066	85%	77%

Table 5.6: Successful logins in part two, for no-storage and storage participants. Participants are considered successful if they entered their secret in five tries without having their secret emailed to them.

Part two secret-entry time		Omnibus (KW $\chi_{10}^2=204.592, p<.001$)		
cond 1	median seconds	cond 2	median seconds	p-value
pw-length5	4.0	pw-length6	5.5	HC MW, U=3931.5, $p=.021$
		pp-small	5.3	HC MW, U=4063.5, $p=.017$
		pp-nouns	8.4	HC MW, U=4256, $p<.001$
		pp-nouns-instr	8.6	HC MW, U=4736, $p<.001$
		pp-sentence	9.0	HC MW, U=2109.5, $p<.001$
		pp-large-3word	5.1	HC MW, U=4152.5, $p=.016$
		pp-med-unorder	6.5	HC MW, U=3193, $p<.001$
		pp-small	5.3	HC MW, U=3765, $p=.001$
pw-pronounce	3.3	pp-nouns	8.4	HC MW, U=3894.5, $p<.001$
		pp-nouns-instr	8.6	HC MW, U=4363.5, $p<.001$
		pp-sentence	9.0	HC MW, U=1899.5, $p<.001$
		pp-large-3word	5.1	HC MW, U=3809, $p<.001$
		pp-med-unorder	6.5	HC MW, U=3002.5, $p<.001$
pp-small	5.3	pp-nouns	8.4	HC MW, U=3675, $p<.001$
		pp-nouns-instr	8.6	HC MW, U=4078, $p=.002$
		pp-sentence	9.0	HC MW, U=1842, $p=.003$
pp-large-3word	5.1	pp-nouns	8.4	HC MW, U=4045, $p=.002$
		pp-nouns-instr	8.6	HC MW, U=4438.5, $p=.013$
		pp-sentence	9.0	HC MW, U=1982, $p=.009$
		pp-large	7.8	HC MW, U=2274, $p=.013$
comb. pw	3.6	comb. pp	7.9	KW $\chi_1^2=148.919, p<.001$

Table 5.7: Differences in password-entry times for participants who entered their secret in five attempts without pasting or autofilling.

	Length	All participants				No-storage			
		Entry time	Login time	Deletions	Edit dist.	Entry time	Login time	Deletions	Edit dist.
pw-length5	5.0	4.0	27.5	0.2	0.9	3.7	32.5	0.2	1.5
pw-pronounce	8.0	3.3	25.0	0.7	0.9	2.7	26.0	0.8	1.1
combined password	6.5	3.6	26.0	0.4	0.9	3.1	27.5	0.5	1.3
pp-small	18.3	5.3	26.0	2.0	1.7	4.2	31.0	0.9	4.4
pp-nouns	24.2	8.4	35.0	3.0	2.5	7.6	35.0	3.9	3.2
pp-nouns-instr	24.7	8.6	34.0	2.7	3.0	7.2	31.0	2.3	4.0
pp-sentence	25.5	9.0	34.0	3.8	2.4	6.0	31.0	5.6	2.1
combined passphrase	23.4	7.9	33.0	2.8	2.5	6.8	32.0	3.2	3.5
pp-med-u	21.3	6.5	36.0	2.7	2.5	5.1	28.5	2.6	1.9
pp-large-3word	18.4	5.1	27.0	2.7	1.1	4.1	23.5	1.7	2.5
pp-medium	21.2	6.6	35.0	1.8	2.3	6.6	44.5	3.3	5.4
pw-length6	6.0	5.5	24.0	0.4	0.8	5.1	44.5	0.2	1.6
pp-large	24.1	7.8	34.5	3.5	2.5	7.0	35.0	3.8	3.7
total	17.2	6.0	31.0	2.0	1.9	5.6	32.0	2.3	2.9

Table 5.8: Length, entry time, login time, number of deletions, and edit distance for each condition. Length is mean characters. Entry time is median part two secret-entry time in seconds, first to last keystroke, for participants who did not paste or autofill their secrets, and who entered them within five attempts. Login time is the median time between a participant first being shown the second-part recall screen and leaving that page for the final time. Mean deletions are shown for participants who entered their secret on the first try during part two recall. Edit distance is the mean distance between the actual secret and what was entered on the first attempt during part two recall.

Annoying				<i>Omnibus</i> $\chi^2_{10}=30.116, p=.001$
cond 1	%	cond 2	%	p-value
pw-length6	61.5	pp-medium	33.7	HC FET, $p=.003$
Difficult				<i>Omnibus</i> $\chi^2_{10}=66.583, p<.001$
pp-large-3word	1.9	pp-large	18.0	HC FET, $p=.002$
		all other 30-bit	14.4 - 25.0	(HC FET, $p<.023$)
pw-length6	44.2	pp-medium	16.8	HC FET, $p=.001$
		pw-length5	22.0	HC FET, $p=.003$
Fun				<i>Omnibus</i> $\chi^2_{10}=43.433, p<.001$
pp-nouns-instr	25.7	pp-small	9.0	HC FET, $p=.014$
		pp-large-3word	8.7	HC FET, $p=.012$
		pw-length5	9.4	HC FET, $p=.001$
pp-medium	20.8	pw-length6	4.8	HC FET, $p=.018$

Table 5.9: Statistically significant differences in agreement with “learning my password was [fun/difficult/annoying].”

Condition	All participants				No-storage				Condition	All participants				No-storage			
	PART ONE		PART TWO		PART ONE		PART TWO			PART ONE		PART TWO		PART ONE		PART TWO	
	On first try	In five tries	On first try	In five tries	On first try	In five tries	On first try	In five tries		On first try	In five tries	On first try	In five tries	On first try	In five tries	On first try	In five tries
pw-length5	95	99	83	99	93	98	72	98	pw-pronounce	95	99	83	99	94	98	75	98
pw-length6	90	99	82	99	90	95	60	100	- Space+case	96	-	84	-	96	-	-	-
pp-small	81	95	79	95	65	85	62	92	pp-nouns-instr	88	97	74	98	80	93	61	97
- Space+case	83	-	85	96	-	-	65	-	- Space+case	89	-	81	99	-	-	73	99
- Near dict.	85	-	82	96	69	-	65	-	- Near dict.	91	-	77	-	83	-	66	-
- Edit dist.	88	-	83	96	73	-	65	-	- Edit dist.	91	-	77	-	83	-	67	-
pp-medium	85	96	77	100	71	97	62	100	pp-sentence	89	98	78	98	85	100	70	93
- Space+case	88	-	80	-	76	-	-	-	- Space+case	93	-	83	99	93	-	78	96
- Near dict.	87	-	80	-	74	-	-	-	- Near dict.	90	99	82	99	-	-	74	96
- Edit dist.	87	-	81	-	74	-	-	-	- Edit dist.	91	99	82	99	-	-	74	96
pp-large	81	96	75	99	72	96	68	100	pp-large-3word	84	93	83	99	64	82	79	96
- Space+case	87	-	81	-	80	-	-	-	- Space+case	87	94	89	-	68	86	82	-
- Near dict.	86	-	78	-	80	-	-	-	- Near dict.	85	95	89	-	68	89	82	-
- Edit dist.	91	97	79	-	88	-	-	-	- Edit dist.	-	94	89	-	-	86	82	-
pp-nouns	89	97	79	97	84	95	70	98	pp-med-unorder	80	92	78	98	72	80	80	96
- Space+case	90	-	82	-	-	-	75	-	- Space+case	81	-	81	99	-	-	-	-
- Near dict.	91	99	84	98	88	96	75	-	- Near dict.	83	-	83	99	76	-	88	-
- Edit dist.	91	98	84	98	88	96	75	-	- Edit dist.	85	93	85	99	80	84	88	-

Table 5.10: The percentage of participants in each condition who successfully recalled their secret in one and in five attempts in the first and second parts of the study. The first row for each condition shows uncorrected data. Subsequent rows show the impact of correction for cases where correction would have allowed more users to log in; in many cases, correction did not help. – *Near dict.* indicates moving words to the closest dictionary word. – *Space+case* indicates ignoring whitespace and capitalization.

Chapter 6

How Secure and Usable are Some Common Password Policies?

6.1 Introduction

Chapter 5 examined the usability of system-assigned passwords and passphrases. Our findings suggested that users struggled with learning and recalling them. In this and subsequent chapters, we shift our focus to user-created passwords. This chapter presents findings using data from our first two papers utilizing MTurk to examine passwords and their users, [48] and [45]. Specifically, this chapter presents data from those papers on how password-composition policies affect the usability and security of user-created passwords. This chapter focuses on common password-composition policies for user-created passwords, including two policies recommended by the National Institute of Standards and Technology (NIST) [16]. The next chapter, Chapter 7, further explores user-created passwords. The password-composition policies that chapter examines are based on findings from this chapter. Prior to this work, it was generally believed that password-composition policies made passwords harder to guess and more secure. However, research had struggled to quantify the level of resistance to guessing provided by different password-composition policies or the individual requirements they comprise.

With the research presented in this chapter, we took a substantial step forward in understanding the effects of password-composition policies on the guessability of passwords. We compiled a dataset of 8,000 plaintext passwords collected from different participants under seven different password-composition policies. We analyzed and compared usability metrics for each condition. We also calculated how many guesses it would take for an attacker to guess each of the passwords. This allowed us to evaluate and contrast the impact on security of each password-composition policy using empirical data.

This chapter proceeds as follows. The general form of our data collection was described in Chapter 4. In Section 6.2, we describe methodology factors specific to this study, including study conditions. We present our demographic findings in Section 6.3 and password-strength findings

This chapter is a partial reproduction of two papers: [48] co-authored with Saranga Komanduri, Patrick Gage Kelley, Michelle Mazurek, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Serge Egelman; and [45] co-authored with Patrick Gage Kelley, Saranga Komanduri, Michelle Mazurek, Richard Shay, Timothy Vidas, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Julio Lopez. This chapter includes new material.

in Section 6.4. Usability findings are in Section 6.5. We discuss the implications of the findings in Section 6.6.

6.2 Methodology

Section 4.1 outlines our data-collection protocol and Section 4.3.2 discusses usability metrics. The data-collection for the research in this chapter used a prototype of the data-collection system that would evolve into the SHELF framework presented in Chapter 3. Section 4.2 describes how we calculate guess numbers for the collected passwords. We also estimate entropy for the passwords in each condition, as discussed in Section 2.3.1. We calculate for each password in a condition the entropy contributed by the number, content, and type of each character. We then sum the individual entropy contributions to estimate the total entropy of the passwords in that condition.

6.2.1 Conditions

We had eight total conditions, with seven sets of password-composition requirements and two password-creation scenarios. We used two scenarios in order to measure the extent to which giving participants different instructions affects password strength.

The *survey scenario* was designed to simulate a scenario in which users create low-value passwords, while the *email scenario* was designed to elicit higher-value passwords. All but one condition used the email scenario. In the *survey scenario*, participants were told, “To link your survey responses, we will use a password that you create below; therefore it is important that you remember your password.” In the *email scenario*, participants were told:

Imagine that your main email service provider has been attacked, and your account became compromised. You need to create a new password for your email account, since your old password may be known by the attackers. Because of the attack, your email service provider is also changing its password rules. Please follow the instructions below to create a new password for your email account. We will ask you to use this password in a few days to log in again, so it is important that you remember your new password. Please take the steps you would normally take to remember your email password and protect this password as you normally would protect the password for your email account. Please behave as you would if this were your real password!

The eight conditions are detailed below. We include NIST’s entropy estimate for passwords with those requirements [16].

- ***basic8survey***: Participants were given the survey scenario and the password-composition policy “Password must have at least 8 characters.” This is the only condition using the survey scenario; all others use the email scenario.
- ***basic8***: Participants were given the password-composition policy “Password must have at least 8 characters.” Only the scenario differentiates this from *basic8survey*.
- ***basic16***: Participants were given the password-composition policy “Password must have at least 16 characters.”

- ***dict8***: Participants were given the password-composition policy “Password must have at least 8 characters. It may not contain a dictionary word.” We performed a dictionary check by removing non-alphabetic characters and checking the remainder against a dictionary, ignoring case. This method is used in practice, including at Carnegie Mellon University. We used the free Openwall list as the dictionary.
- ***comp8***: Participants were given the password-composition policy “Password must have at least 8 characters including an uppercase and lowercase letter, a symbol, and a digit. It may not contain a dictionary word.” We performed the same dictionary check as in *dict8*. This condition reproduced NIST’s comprehensive password-composition requirements [16].
- ***blacklistEasy***: Participants were given the password-composition policy “Password must have at least 8 characters. It may not contain a dictionary word.” We checked the password against the simple Unix dictionary, ignoring case. Unlike the *dict8* and *comp8* conditions, the password was not stripped of non-alphabetic characters before the check.
- ***blacklistMedium***: This condition was the same as the *blacklistEasy* condition, except we used the paid Openwall list.
- ***blacklistHard***: This condition was the same as the *blacklistEasy* condition, except we used a five-billion-word dictionary we created by making five billion guesses with a PCFG password-guessing algorithm trained on the MySpace, RockYou, and inflection lists. Both the training and testing were case-insensitive.

6.3 Demographics

We collected data between August 2010 and January 2011. We collected 5,000 participants across a subset of our conditions for [48]. For [45], we collected data until we had 1,000 participants for each condition.

Among participants who completed Part One of the study, 54.9% returned within 3 days and completed part two. This did not vary significantly by condition ($\chi^2_7=11.132$, $p=0.133$). We detected no statistically significant difference in the guessability of passwords between participants who participated in just the first part of the study and those who participated in both parts.

Among the 8,000 participants, 52.0% percent reported being male and 45.9% female, with a mean reported age of 29.6 years. This makes our sample more male and slightly younger than Mechanical Turk participants in general [14]. 33.3% of participants reported studying or working in computer science or a related field.

Participants in *blacklistMedium* (56.2%) were more likely to be male than those in *comp8* (48.1%) (HC FET, $p=.01$). Participants in *basic16* (mean 30.3) were significantly older than those in *blacklistHard* (28.6) (HC FET, $p=.012$). And *blacklistEasy* participants (37.6%) were significantly more likely to be technical than *blacklistHard* participants (30.5%) (HC FET, $p=.027$). No other pairwise demographics comparisons were significant. Because we looked only at participants who finished Part One of the study, these differences may be due to dropout induced by the conditions themselves.

To look at study dropout rates between conditions, we consider all participants who began the study during the time period in which our 1,000 participants per condition were collected.

Part One Completion				Omnibus $\chi^2=124.632, p<.001$	
cond 1	%	cond 2	%	p-value	
<i>blacklistEasy</i>	88.8%	<i>basic8survey</i>	83.5%	HC FET, $p=.004$	
		<i>dict8</i>	82.3%	HC FET, $p<.001$	
		<i>comp8</i>	75.3%	HC FET, $p<.001$	
<i>blacklistMedium</i>	86.3%	<i>comp8</i>	75.3%	HC FET, $p<.001$	
<i>blacklistHard</i>	86.2%	<i>comp8</i>	75.3%	HC FET, $p<.001$	
<i>basic8</i>	85.6%	<i>comp8</i>	75.3%	HC FET, $p<.001$	
<i>basic16</i>	84.9%	<i>comp8</i>	75.3%	HC FET, $p<.001$	
<i>basic8survey</i>	83.5%	<i>comp8</i>	75.3%	HC FET, $p<.001$	
<i>dict8</i>	82.3%	<i>comp8</i>	75.3%	HC FET, $p<.001$	

Table 6.1: Significant differences in completing Part One of the study.

Participants in *comp8* were significantly less likely to finish Part One of the study than in any other condition. Significant differences are shown in Table 6.1.

6.4 Security Results

In this section, we discuss password-strength results. In [45], we included a more detailed account of using different password-cracking techniques. In this thesis, we focus on contrasting the strength of our conditions.

Among conditions we tested, *basic16* provided the greatest security against an attacker capable of a large number of guesses, outperforming the complicated *comp8* condition. This advantage does not appear, however, until the attacker has had a chance to make a larger number of guesses. We note that this remains true even when we tune the cracking algorithm to target the *basic16* and *comp8* conditions.

While a limited relationship between information entropy and guessability can be observed, especially when considering attacks on the order of a trillion guesses or more, entropy provided no more than a very rough approximation of overall password strength.

6.4.1 Comparing Policies for Guessability

Figure 6.1 shows password strength results when the attacker uses publicly available data as well as experimental data. The y-axis represents the proportion of passwords guessed and the x-axis represents the number of guesses made in log scale. Each condition is divided into two halves, and each half is combined with public data and used for training when generating guess numbers for the other half.

As suggested by Figure 6.1, which password-composition policy is best at resisting guessing attacks depends on how many attempts an attacker is able to make. All of the following pairwise tests are for (HC FET, $p<.05$).

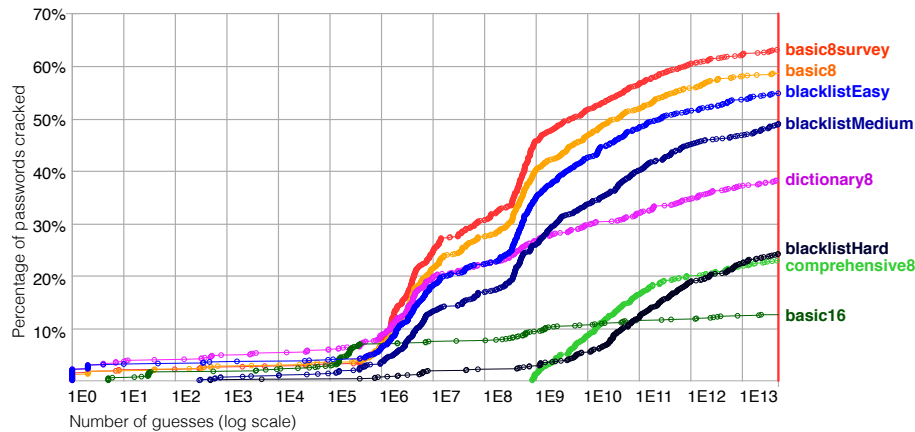


Figure 6.1: The number of passwords cracked compared to the number of guesses per condition. The PCFG guess calculator was trained on both publicly available data and our collected data with two folds per condition.

At one million and one billion guesses, significantly fewer *blacklistHard* and *comp8* passwords were guessed than in any other condition. At one billion guesses, 9.5%, 1.4%, and 2.9% of passwords were cracked in *basic16*, *comp8*, and *blacklistHard* respectively; 40.3% of *basic8* passwords were cracked.

As the number of guesses increases, *basic16* begins to outperform the other conditions. At one trillion guesses, significantly fewer *basic16* passwords were cracked than *comp8* passwords, which were in turn cracked significantly less than any other condition. After exhausting the PCFG-algorithm guessing space, *basic16* remains significantly hardest to crack. The next best at resisting cracking were *comp8* and *blacklistHard*, performing significantly better than any of the other conditions. 14.6, 26.4, and 31.0% of passwords were cracked in *basic16*, *comp8*, and *blacklistHard* respectively; in contrast, 63.0% *basic8* passwords were cracked.

6.4.2 Guessability and Entropy

Historically, Shannon entropy (as discussed in Section 2.3.1) has provided a convenient single statistic to summarize password strength. While information entropy does provide a theoretical lower bound on the guessability of a set of passwords [57], in practice a system administrator may be more concerned about how many passwords can be cracked in a given number of guesses than about the average guessability across the population. Although there is no mathematical relationship between entropy and this definition of guess resistance, we examine the possibility that the two are correlated in practice. To do this, we consider two independent measures of entropy: an empirically calculated estimate and a theoretical NIST estimate. For both measures, we find that entropy estimates roughly indicate which composition policies provide more guess resistance than others, but provide no useful information about the magnitude of these differences.

We ranked our password conditions based on the proportion of passwords cracked in Figure 6.1 at one trillion guesses, and compared this to the rank of conditions based on empirically

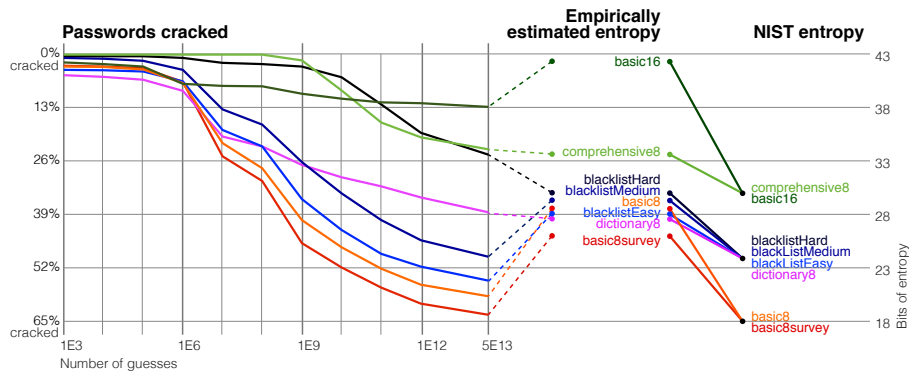


Figure 6.2: Relationship among the resistance of our collected password sets to cracking; empirical entropy estimates we calculated from those sets; and 2006 NIST entropy estimates for our password conditions.

estimated entropy. We found these rankings, shown in Figure 6.2, to be significantly correlated (Kendall’s $\tau = 0.71$, Holm-corrected $p = 0.042$). However, looking at the proportion of passwords cracked at a million or a billion guesses, the correlation in rankings is no longer significant (Holm-corrected $p = 0.275, 0.062$). This suggests that entropy might be useful when considering an adversary who can make a large number of guesses, but is not useful when considering a smaller number of guesses.

Further, empirically estimated entropy was unable to predict correctly the ranking of *dictionary8*, even when considering a large number of guesses. This condition displayed greater resistance to guessability than *basic8*, yet its empirically estimated entropy was lower. This might indicate a flaw in how entropy was estimated, a flaw in the guessing algorithm, or an innate shortcoming of the use of entropy to predict guessability. Since entropy can only lower-bound the guessability of passwords, it is possible for the frequency distribution of *dict8* to have low entropy but high guess resistance.

Examining the 2006 NIST entropy of our password conditions produces three equivalence classes, as shown in Figure 6.2. These arise because NIST entropy is not granular enough to capture all differences between our conditions. First, NIST entropy does not take into account the size of a dictionary or its implementation. All five of our dictionary and blacklist conditions meet the NIST requirement of a dictionary with at least 50,000 words [16]. Implementation details, such as case-insensitive blacklist checking or the removal of non-alphabetic characters before a dictionary check, are not considered in the entropy score. Our results show that these details lead to password policies with very different levels of password strength and should be considered in a future heuristic. Further, the NIST entropy scores for *basic16* and *comp8* are the same, even though *basic16* appears to be much more resistant to powerful guessing attacks.

Perhaps surprisingly, the equivalence classes given by NIST entropy are ordered correctly based on our results for guessability after 50 trillion guesses. Though its lack of granularity fails to capture differences between similar password conditions, NIST entropy seems to succeed at its stated purpose of providing a “rough rule of thumb” [16].

We stress that although both measures of entropy provide a rough ordering among policies, they do not always correctly classify guessability (see for example *dictionary8*), and they do not

Agreement with study password more secure				Omnibus $\chi^2=233.41, p<.001$
cond 1	%	cond 2	%	p-value
<i>comp8</i>	67.2%	<i>basic16</i>	56.8%	HC FET, $p<.001$
		<i>blacklistMedium</i>	50.6%	HC FET, $p<.001$
		<i>dict8</i>	49.3%	HC FET, $p<.001$
		<i>blacklistHard</i>	49.0%	HC FET, $p<.001$
		<i>blacklistEasy</i>	43.5%	HC FET, $p<.001$
		<i>basic8</i>	44.5%	HC FET, $p<.001$
		<i>basic8survey</i>	38.4%	HC FET, $p<.001$
<i>basic16</i>	56.8%	<i>dict8</i>	49.3%	HC FET, $p=.013$
		<i>blacklistHard</i>	49.0%	HC FET, $p=.008$
		<i>blacklistEasy</i>	43.5%	HC FET, $p<.001$
		<i>basic8</i>	44.5%	HC FET, $p<.001$
		<i>basic8survey</i>	38.4%	HC FET, $p<.001$
<i>blacklistMedium</i>	50.6%	<i>blacklistEasy</i>	43.5%	HC FET, $p=.022$
		<i>basic8</i>	44.5%	HC FET, $p=.023$
		<i>basic8survey</i>	38.4%	HC FET, $p<.001$
<i>dict8</i>	49.3%	<i>basic8survey</i>	38.4%	HC FET, $p<.001$
<i>blacklistHard</i>	49.0%	<i>basic8survey</i>	38.4%	HC FET, $p<.001$

Table 6.2: Significant differences in participant agreement that study passwords were more secure than real email passwords.

effectively measure how much additional guess resistance one policy provides as compared to another. These results suggest that a “rough rule of thumb” may be the limit of entropy’s usefulness as a metric.

6.4.3 User Perception of Security

In order to measure how participants perceived the strength of their study passwords, we asked them whether they agreed with, “If my main email account required me to change my password using the same requirements as used in this study, it would make my email account more secure.” Significant differences in responses are shown in Table 6.2. It is interesting that the stronger two conditions both performed best in this question, though in opposite order. Participants in *comp8* were more likely to agree than any other condition, with participants in *basic16* more likely to agree than any other condition except *comp8*.

6.5 Usability Results

This section presents usability results for the 1,000 participants in each condition. We find, overall, that the *basic16* and *comp8* passwords were generally less usable than those in the more lenient conditions. We find, further, that *basic16* is generally more usable than *comp8*, despite the former being more secure than the latter after a number of guesses.

Password creation attempts			Omnibus KW $\chi^2=2024.694, p<.001$	
cond 1	counts	cond 2	counts	p-value
<i>comp8</i>	3.4	<i>blacklistHard</i>	2.1	HC MW, U=706766.5, $p<.001$
		<i>dict8</i>	1.9	HC MW, U=736313, $p<.001$
		<i>basic16</i>	1.7	HC MW, U=767023, $p<.001$
		<i>blacklistMedium</i>	1.4	HC MW, U=821712, $p<.001$
		<i>blacklistEasy</i>	1.2	HC MW, U=860905, $p<.001$
		<i>basic8survey</i>	1.2	HC MW, U=873751, $p<.001$
		<i>basic8</i>	1.1	HC MW, U=881944, $p<.001$
<i>blacklistHard</i>	2.1	<i>dict8</i>	1.9	HC MW, U=531588.5, $p=.026$
		<i>basic16</i>	1.7	HC MW, U=542148, $p=.002$
		<i>blacklistMedium</i>	1.4	HC MW, U=624107.5, $p<.001$
		<i>blacklistEasy</i>	1.2	HC MW, U=670948.5, $p<.001$
		<i>basic8survey</i>	1.2	HC MW, U=687127, $p<.001$
		<i>basic8</i>	1.1	HC MW, U=702157, $p<.001$
<i>dict8</i>	1.9	<i>blacklistMedium</i>	1.4	HC MW, U=592563, $p<.001$
		<i>blacklistEasy</i>	1.2	HC MW, U=639836.5, $p<.001$
		<i>basic8survey</i>	1.2	HC MW, U=656290.5, $p<.001$
		<i>basic8</i>	1.1	HC MW, U=672197.5, $p<.001$
<i>basic16</i>	1.7	<i>blacklistMedium</i>	1.4	HC MW, U=595772, $p<.001$
		<i>blacklistEasy</i>	1.2	HC MW, U=650158, $p<.001$
		<i>basic8survey</i>	1.2	HC MW, U=668797, $p<.001$
		<i>basic8</i>	1.1	HC MW, U=686031.5, $p<.001$
<i>blacklistMedium</i>	1.4	<i>blacklistEasy</i>	1.2	HC MW, U=549397, $p<.001$
		<i>basic8survey</i>	1.2	HC MW, U=566935.5, $p<.001$
		<i>basic8</i>	1.1	HC MW, U=585787.5, $p<.001$
<i>blacklistEasy</i>	1.2	<i>basic8</i>	1.1	HC MW, U=538323, $p<.001$
<i>basic8survey</i>	1.2	<i>basic8</i>	1.1	HC MW, U=520948.5, $p=.026$

Table 6.3: Statistically significant differences in creation attempts across conditions.

6.5.1 Password Creation

Participants were asked to create a password matching the requirements of their condition. Submitting a password that did not meet the requirements resulted in the participant being told what was wrong with the submitted password and being asked to create another. This repeated until the participant entered a satisfactory password.

Significant differences in the number of attempts to create a password successfully are in Table 6.3. We see that participants in *comp8* took significantly more attempts to create a satisfactory password than in any other condition. Participants in *blacklistHard* struggled the most after that, taking more attempts than participants in any condition other than *comp8*. Participants in *basic16* took more attempts than those in half of the conditions yet still took fewer than two attempts on average.

We also asked participants whether they agreed with the creation process being annoying, difficult, and fun. Significant differences in these are in Tables 6.4, 6.5, and 6.6 respectively.

We observe that *comp8* was the most annoying and most difficult to create, more so than

Agreement with Creation Annoying				Omnibus $\chi^2=681.406, p<.001$
cond 1	%	cond 2	%	p-value
<i>comp8</i>	65.4%	<i>basic16</i>	56.7%	HC FET, $p<.001$
		<i>blacklistHard</i>	42.3%	HC FET, $p<.001$
		<i>dict8</i>	37.3%	HC FET, $p<.001$
		<i>blacklistMedium</i>	35.0%	HC FET, $p<.001$
		<i>blacklistEasy</i>	27.2%	HC FET, $p<.001$
		<i>basic8survey</i>	26.6%	HC FET, $p<.001$
		<i>basic8</i>	22.0%	HC FET, $p<.001$
<i>basic16</i>	56.7%	<i>blacklistHard</i>	42.3%	HC FET, $p<.001$
		<i>dict8</i>	37.3%	HC FET, $p<.001$
		<i>blacklistMedium</i>	35.0%	HC FET, $p<.001$
		<i>blacklistEasy</i>	27.2%	HC FET, $p<.001$
		<i>basic8survey</i>	26.6%	HC FET, $p<.001$
		<i>basic8</i>	22.0%	HC FET, $p<.001$
<i>blacklistHard</i>	42.3%	<i>blacklistMedium</i>	35.0%	HC FET, $p=.006$
		<i>blacklistEasy</i>	27.2%	HC FET, $p<.001$
		<i>basic8survey</i>	26.6%	HC FET, $p<.001$
		<i>basic8</i>	22.0%	HC FET, $p<.001$
<i>dict8</i>	37.3%	<i>blacklistEasy</i>	27.2%	HC FET, $p<.001$
		<i>basic8survey</i>	26.6%	HC FET, $p<.001$
		<i>basic8</i>	22.0%	HC FET, $p<.001$
<i>blacklistMedium</i>	35.0%	<i>blacklistEasy</i>	27.2%	HC FET, $p=.001$
		<i>basic8survey</i>	26.6%	HC FET, $p<.001$
		<i>basic8</i>	22.0%	HC FET, $p<.001$
<i>blacklistEasy</i>	27.2%	<i>basic8</i>	22.0%	HC FET, $p=.04$

Table 6.4: Significant differences in how annoying passwords were to create.

any other condition. *basic16* followed, being more annoying and difficult to create than anything besides itself and *comp8*. Differences in annoyance are illustrated in Figure 6.3.

6.5.2 Part One Recall

Table 6.7 shows statistical differences in the number of attempts to recall the password in Part One recall. *basic8* took fewer attempts than *comp8*, *basic16*, and *dict8*. However, the effect size is small, with those taking on average 1.2 instead of 1.1 attempts.

6.5.3 Part Two Recall

When they returned for part two of the study, participants had the option to click a reminder link to be emailed a link to their password. Usage of this reminder did not vary significantly between conditions ($\chi^2=5.008, p=0.659$). Nor did we see a significant difference in the number of attempts needed to recall the password by condition ($\chi^2=10.6, p=0.157$).

Agreement with creation difficult				Omnibus $\chi^2=602.315, p<.001$
cond 1	%	cond 2	%	p-value
<i>comp8</i>	38.8%	<i>basic16</i>	30.7%	HC FET, $p=.001$
		<i>blacklistHard</i>	21.9%	HC FET, $p<.001$
		<i>dict8</i>	19.5%	HC FET, $p<.001$
		<i>blacklistMedium</i>	16.1%	HC FET, $p<.001$
		<i>blacklistEasy</i>	9.6%	HC FET, $p<.001$
		<i>basic8</i>	7.8%	HC FET, $p<.001$
		<i>basic8survey</i>	6.3%	HC FET, $p<.001$
<i>basic16</i>	30.7%	<i>blacklistHard</i>	21.9%	HC FET, $p<.001$
		<i>dict8</i>	19.5%	HC FET, $p<.001$
		<i>blacklistMedium</i>	16.1%	HC FET, $p<.001$
		<i>blacklistEasy</i>	9.6%	HC FET, $p<.001$
		<i>basic8</i>	7.8%	HC FET, $p<.001$
<i>blacklistHard</i>	21.9%	<i>blacklistMedium</i>	16.1%	HC FET, $p=.007$
		<i>blacklistEasy</i>	9.6%	HC FET, $p<.001$
		<i>basic8</i>	7.8%	HC FET, $p<.001$
		<i>basic8survey</i>	6.3%	HC FET, $p<.001$
		<i>dict8</i>	19.5%	HC FET, $p<.001$
<i>dict8</i>	19.5%	<i>blacklistEasy</i>	9.6%	HC FET, $p<.001$
		<i>basic8</i>	7.8%	HC FET, $p<.001$
		<i>basic8survey</i>	6.3%	HC FET, $p<.001$
<i>blacklistMedium</i>	16.1%	<i>blacklistEasy</i>	9.6%	HC FET, $p<.001$
		<i>basic8</i>	7.8%	HC FET, $p<.001$
		<i>basic8survey</i>	6.3%	HC FET, $p<.001$
<i>blacklistEasy</i>	9.6%	<i>basic8survey</i>	6.3%	HC FET, $p=.04$

Table 6.5: Significant differences in how difficult passwords were to create.

Agreement with recall being difficult is depicted in Figure 6.3, and statistical differences between conditions are shown in Table 6.8. Here again, we see the most difficulty being with *comp8*, followed by *basic16*.

6.5.4 Password Storage

We asked participants who returned for part two whether they wrote down or otherwise stored their password. We also detected pasting and auto-filling of passwords during both password recalls. Participants are considered to have stored their password if we ever detected them pasting or auto-filling their password. Further, participants are considered to have stored their password unless explicitly answering “No” to “Did you write down or store the password you created for this study?” Differences in storage between part-two participants are shown in Table 6.9. We see that participants in *comp8* were the most likely to have stored their password, followed by participants in *basic16*.

Agreement with creation fun				Omnibus $\chi^2=28.484, p<.001$	
cond 1	%	cond 2	%	p-value	
<i>basic8</i>	24.5%	<i>basic8survey</i>	16.3%	HC FET, $p<.001$	
<i>blacklistMedium</i>	24.1%	<i>basic8survey</i>	16.3%	HC FET, $p<.001$	
<i>blacklistEasy</i>	23.5%	<i>basic8survey</i>	16.3%	HC FET, $p=.002$	
<i>dict8</i>	23.0%	<i>basic8survey</i>	16.3%	HC FET, $p=.005$	

Table 6.6: Significant differences in how fun passwords were to create.

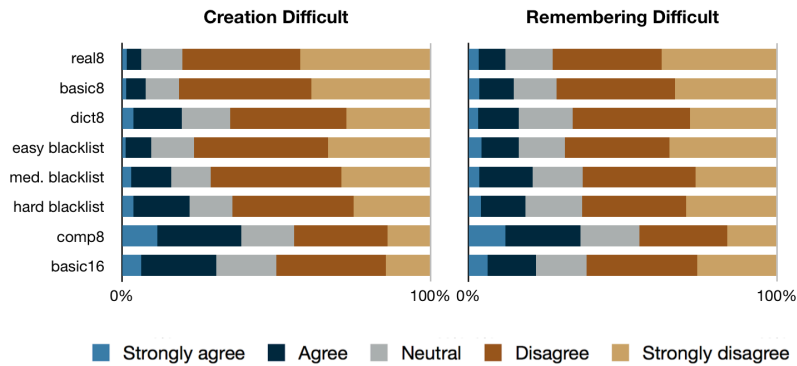


Figure 6.3: Participant agreement with “Creating a password that meets the requirements given in this study was difficult” and “Remembering the password I used for this study was difficult.”

6.6 Discussion

Table 6.10 contains all of the metrics from the comparison tables in this chapter.

Although the number and complexity of password-composition requirements are steadily increasing, the actual value added by those requirements is poorly understood. In this work, we take a substantial step forward in understanding those requirements, both their strength and their usability.

We found several notable results regarding the comparative strength of different composition policies. Although NIST estimates *basic16* and *comp8* to be equivalent in strength, we found that *basic16* is superior for large numbers of guesses. We also observed that *basic16* is easier on users,

Part One recall attempts				Omnibus KW $\chi^2=26.71, p<.001$	
cond 1	count	cond 2	count	p-value	
<i>comp8</i>	1.2	<i>basic8</i>	1.1	HC MW, $U=523639.5, p<.001$	
<i>dict8</i>	1.2	<i>basic8</i>	1.1	HC MW, $U=522456.5, p<.001$	
<i>basic16</i>	1.2	<i>basic8</i>	1.1	HC MW, $U=518018.5, p=.013$	

Table 6.7: Statistically significant differences in Part One recall attempts across conditions.

Agreement with recall difficult				<i>Omnibus</i> $\chi^2=131.042, p<.001$	
cond 1	%	cond 2	%	p-value	
<i>comp8</i>	38.9%	<i>blacklistEasy</i>	21.5%	HC FET, $p<.001$	
		<i>basic16</i>	26.2%	HC FET, $p<.001$	
		<i>basic8</i>	17.5%	HC FET, $p<.001$	
		<i>dict8</i>	19.9%	HC FET, $p<.001$	
		<i>blacklistMedium</i>	22.7%	HC FET, $p<.001$	
		<i>basic8survey</i>	15.6%	HC FET, $p<.001$	
		<i>blacklistHard</i>	21.5%	HC FET, $p<.001$	
<i>basic16</i>	26.2%	<i>basic8</i>	17.5%	HC FET, $p=.006$	
		<i>basic8survey</i>	15.6%	HC FET, $p<.001$	
<i>blacklistMedium</i>	22.7%	<i>basic8survey</i>	15.6%	HC FET, $p=.006$	

Table 6.8: Significant differences in how difficult passwords were to recall.

Storage				<i>Omnibus</i> $\chi^2=143.318, p<.001$	
cond 1	%	cond 2	%	p-value	
<i>comp8</i>	61.4%	<i>basic16</i>	44.3%	HC FET, $p<.001$	
		<i>dict8</i>	40.8%	HC FET, $p<.001$	
		<i>blacklistHard</i>	39.0%	HC FET, $p<.001$	
		<i>blacklistMedium</i>	38.4%	HC FET, $p<.001$	
		<i>blacklistEasy</i>	36.5%	HC FET, $p<.001$	
		<i>basic8</i>	35.7%	HC FET, $p<.001$	
		<i>basic8survey</i>	29.2%	HC FET, $p<.001$	
<i>basic16</i>	44.3%	<i>basic8survey</i>	29.2%	HC FET, $p<.001$	
<i>dict8</i>	40.8%	<i>basic8survey</i>	29.2%	HC FET, $p=.001$	
<i>blacklistHard</i>	39.0%	<i>basic8survey</i>	29.2%	HC FET, $p=.012$	
<i>blacklistMedium</i>	38.4%	<i>basic8survey</i>	29.2%	HC FET, $p=.028$	

Table 6.9: Significant differences in password storage for participants who finished part two.

suggesting it as the better choice for stronger passwords.

We also found that the effectiveness of a dictionary check depends heavily on the choice of dictionary; in particular, a large blacklist created using state-of-the-art password-guessing techniques is much more effective than a standard dictionary at preventing users from choosing easily guessed passwords.

Finally, we report that Shannon entropy, though a convenient single-statistic metric of password strength, provides only a rough correlation with guess resistance and is unable to predict quantitative differences in guessability among password sets.

Condition	Cracked@10 ¹² (%)	Part One completion (%)	Password storage (%)	Mean creation attempts	Agree creation difficult (%)	Agree creation annoying (%)	Part One recall attempts	Part Two recall attempts	Agree recall difficult (%)	Agree study secure (%)	
<i>basic8survey</i>	60.5	83.5	29.2	1.2	6.3	26.6	16.3	1.1	1.3	15.6	38.4
<i>basic8</i>	56.0	85.6	35.7	1.1	7.8	22.0	24.5	1.1	1.3	17.5	44.5
<i>dict8</i>	34.7	82.3	40.8	1.9	19.5	37.3	23.0	1.2	1.4	19.9	49.3
<i>blacklistEasy</i>	51.5	88.8	36.5	1.2	9.6	27.2	23.5	1.1	1.5	21.5	43.5
<i>blacklistMedium</i>	45.1	86.3	38.4	1.4	16.1	35.0	24.1	1.1	1.4	22.7	50.6
<i>blacklistHard</i>	19.0	86.2	39.0	2.1	21.9	42.3	21.1	1.2	1.4	21.5	49.0
<i>comp8</i>	20.1	75.3	61.4	3.4	38.8	65.4	21.1	1.2	1.5	38.9	67.2
<i>basic16</i>	11.8	84.9	44.3	1.7	30.7	56.7	21.1	1.2	1.5	26.2	56.8

Table 6.10: This table contains all of the metrics presented in comparison tables in this chapter. *Agree study secure* refers to the percentage of participants who agree that the study requirements would make their real email password more secure.

6.6.1 Meeting comp8 Requirements

It is noteworthy that participants in *comp8* often struggled with their passwords, and believed their security to be highest. However, the condition did not provide as much protection from a resourceful attacker as *basic16*. In order to gain a better understanding of this, this section looks at how participants met the requirements of *comp8*. Participants tended to meet the character-class requirements in predictable ways, in turn making those requirements much less effective in increasing password variety.

82.5% of *comp8* passwords exceeded the minimum length of eight characters; the median length was ten characters. Using more digits than required was fairly popular (63.5%). However, participants were less keen on using more than the single required uppercase letter (26.2%) or symbol (11.2%).

In fact, the majority of *comp8* passwords (60.4%) began with a capital letter and used no other capital letter. Half of *comp8* participants both used only one symbol, and placed this symbol as the last or second-to-last character. 20.6% of participants used only “!” for their symbol, and another 221 used only “@” for the symbol.

6.6.2 Shortcomings of the basic16 Condition

Against an attacker capable of a large number of guesses, *basic16* was highly effective. However, observing Figure 6.1 reveals that *basic16* still led to some easily guessed passwords. Against a more limited attacker, in fact, *comp8* performed better. This suggests that, even with the advantages of *basic16*, it could be improved such that it precludes the “low-hanging fruit” in the condition. This shortcoming will be discussed and addressed in subsequent studies in this thesis.

Chapter 7

Can Longer Passwords be Secure and Usable?

7.1 Introduction

In Chapter 6, the two most secure password-composition policies we examined were *comp8* and *basic16*. *comp8* is similar to the password-composition policy in use at Carnegie Mellon University, and requires eight characters, four character classes, and a dictionary check.¹ *basic16* requires 16 characters with no other requirements. Chapter 6 found that password creation and recall were easier and less error-prone under *basic16* than *comp8*. On average, *basic16* passwords were also significantly more difficult to guess. However, a number of participants created very simple, easily guessed passwords under *basic16*, such as *passwordpassword*. Despite its usability advantages over *comp8*, the large proportion of easily guessed *basic16* passwords make it poorly suited for real-world deployment.

In this chapter, we present the findings of two studies intended to find policies that we can recommend to service providers. We examined 16 password-composition policies, testing several permutations of length and character-based requirements. We compared password strength, as well as user sentiment, timing, and attempts required for password creation and recall. The first study examined 13,751 passwords created by participants under one of eight policies. These policies included *comp8* as well as policies requiring 12, 16, and 20 characters without further requirements. We also tested adding a three-character-class requirement to the length-12 and length-16 requirements. To evaluate the effectiveness of passphrases, we tested requiring two “words” – strings of letters separated by non-letters. This study showed that certain combinations of character requirements and longer-length requirements led to fewer easily guessed passwords than *basic16*, while still being more usable and more secure than *comp8*.

Many weak passwords in the first study shared common *substrings*, sequences of characters within the password. For example, passwords containing the string *1234* were three times as likely to be guessed as those that did not contain that string. In policies that required multiple character

This chapter includes a partial reproduction of [87], co-authored with Saranga Komanduri, Adam Durity, Phillip (Seyoung) Huh, Michelle Mazurek, Sean M. Segreti, Blase Ur, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. This chapter includes new material.

¹<http://www.cmu.edu/iso/governance/guidelines/password-management.html>

classes, passwords beginning and ending with lowercase letters were much stronger than those beginning or ending with digits, symbols, or uppercase letters. Further, in the first study, some conditions that combined length requirements of 12 or 16 characters with character or structural requirements had performed especially well. These observations encouraged us to conduct the second study presented in this chapter. In the second study, we tested password policies that had substring blacklists and policies that required passwords to begin and end with lowercase letters. We also tested additional combinations of length and character-class requirements based on findings from the first study. We collected data from 8,740 participants, each assigned to one of eight password-composition policies.

In the second study, we found further evidence that a password policy requiring 12 characters and two or three character classes is more usable and more secure than the traditional *comp8* policy. We also found that adding a substring blacklist check to this policy significantly improved password strength without having a negative impact on password recall. These findings let us make practical policy recommendations to service providers who want their users to make stronger passwords.

We present findings from our first study in Section 7.2. In Section 7.3, we discuss how the findings from our first study led to the conditions in our second study. We then present the findings of the second study in Section 7.4. Finally, we discuss the implications of both of these studies together in Section 7.5.

7.2 Study I

This section presents the first of the two studies in this chapter. We describe the study conditions in Section 7.2.1. We discuss the study participants and their demographics in Section 7.2.2. Security results are in Section 7.2.3 and usability results in Section 7.2.4. We discuss common patterns found in weaker passwords in Section 7.2.5.

7.2.1 Conditions

We assigned participants to one of eight conditions, each with its own password policy.

comp8— We asked participants to include at least eight characters and all four character classes. We also performed a dictionary check on the letters in the password. We used the free Openwall password-cracking dictionary² for this dictionary check. We removed digits and symbols from the prospective password and checked it against this dictionary, case-insensitive.

basic12, *basic16*, *basic20*— We asked participants to include at least 12, 16, or 20 characters.

2word12, *2word16*— We asked participants to include at least 12 or 16 characters, and to have “at least two words (letter sequences separated by a non-letter sequence).”

3class12, *3class16*— We asked participants to include at least 12 or 16 characters, and at least three of the four character classes. These conditions combined longer length requirements with some of the character-class requirements of *comp8*.

²<http://www.openwall.com/wordlists/>

Condition	Partic- ipants	Length (median)	Upper (median)	Lower (median)	Digit (median)	Sym. (median)	Fail (%)	Length (%)	Class (%)	Dict. (%)	2word (%)
<i>comp8</i>	1996	10	1	5	2	1	58.0	6.5	26.3	39.0	–
<i>basic12</i>	1693	13	0	10	3	0	40.6	38.2	–	18.5*	–
<i>basic16</i>	1757	17	0	14	3	0	52.6	50.4	–	6.3*	–
<i>basic20</i>	1715	21	0	18	3	0	59.9	57.3	–	4.3*	–
<i>3class12</i>	1653	13	1	8	3	1	44.5	38.2	9.5	23.4*	–
<i>3class16</i>	1625	17	1	11	3	1	52.2	47.2	10.0	9.7*	–
<i>2word12</i>	1659	14	0	11	2	0	54.5	30.4	9.9	6.5*	45.4
<i>2word16</i>	1653	18	0	14	2	1	59.8	44.8	9.6	2.6*	45.1

Table 7.1: Summary of password attributes and creation failure on the first attempt. A password can fail multiple ways. We omit failure from blank fields and confirmation mismatch. *Dict* shows the percent of *comp8* participants who failed the dictionary check on their first attempt. It also shows the percentage of final passwords in other conditions that would have failed the dictionary check.

7.2.2 Participants

We recruited participants between April and June 2013. Table 7.1 shows the number of participants per condition. Of the 15,108 participants who began our study, 13,751 finished Part One. Except the discussion of dropout rates, our analysis focuses on these participants or a subset of these participants. 8,565 returned for Part Two within three days of receiving our invitation; 8,143 finished Part Two. When discussing metrics from Part Two, we focus on these participants.

51.5% of participants reported being male and 47.4% reported being female. Participants' mean age was 29.3 years (median 26). These did not vary significantly between conditions. Looking at user-agent strings, only 1.5% of participants appeared to be using mobile devices.

7.2.3 Security Results

Table 7.1 has descriptive statistics for passwords. Table 7.3 contains a summary of the results for Study I. Participants typically avoided uppercase letters or symbols in their passwords, but often included digits. Figure 7.1 shows the percentage of passwords cracked in each condition as additional guesses are made. Table 7.3 shows the percentages of passwords guessed in each condition after 10^8 and 10^{12} guesses, with pairwise significant differences in Table 7.2.

Overall, passwords in the basic conditions performed poorly against an attacker limited to 10^8 guesses. Passwords in *comp8* performed poorly against an attacker capable of making 10^{12} guesses. A number of conditions that combined longer length and character-class requirements, such as *3class12* and *2word16*, performed well across a range of guess numbers.

Condition *3class16* is the strongest after both 10^8 and 10^{12} guesses. Consistent with the findings of Chapter 6, the basic conditions perform relatively well after 10^{12} guesses, but also contain a number of passwords that are cracked after a small number of guesses. The *2word* and *3class* conditions are stronger than their basic counterparts.

The *comp8* condition is relatively strong against a resource-limited attacker capable only of 10^8 guesses. However, *comp8* performs poorly as the number of guesses increases. After 10^{12} guesses, *comp8* passwords are significantly more likely to be cracked than any other condition

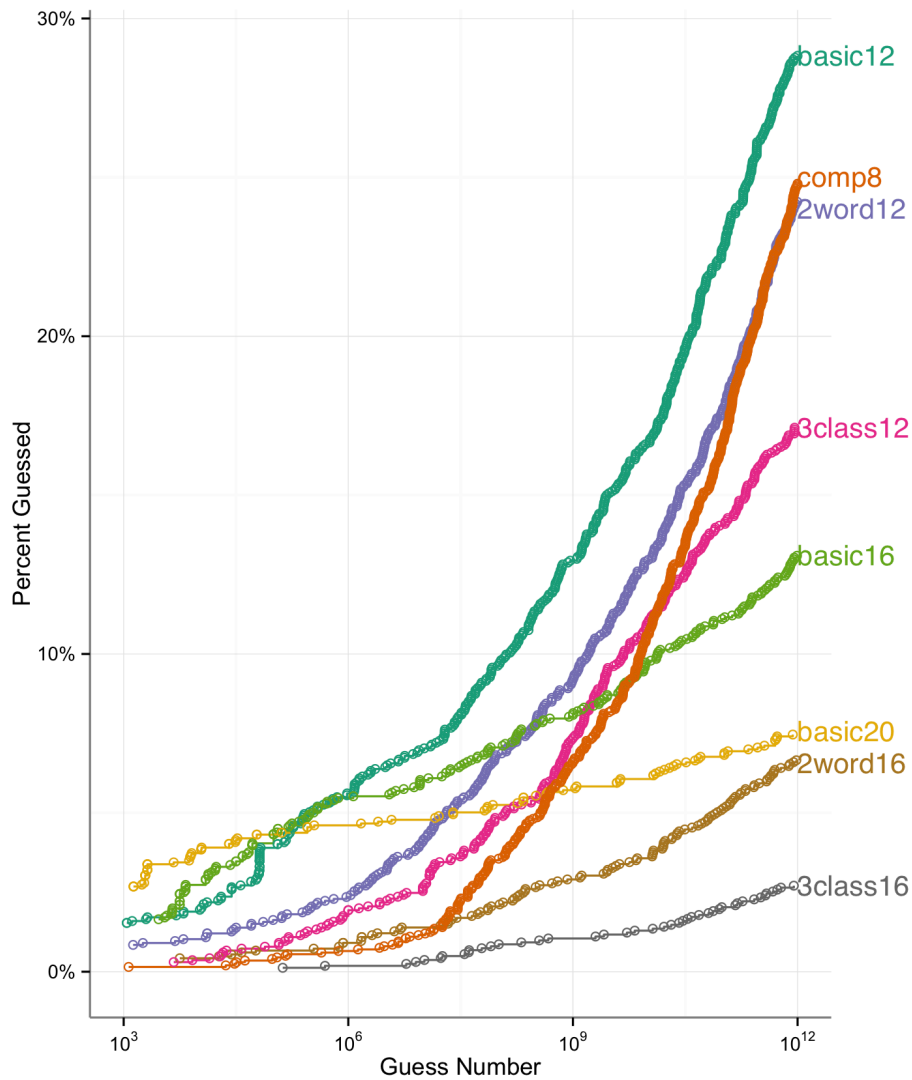


Figure 7.1: The percentage of passwords cracked in each condition by the number of guesses made, in log scale. Our cutoff for guess numbers was 10^{12} .

except *basic12* and *2word12*. *3class12* is similar in strength to *comp8* until around 10^{10} guesses, and remains stronger after.

It appears that adding the 2word requirement improves *basic16* more than *basic12*. Manually examining passwords from participants who finished Part Two shows that *2word16* passwords contained three words 31.8% of the time, almost twice as often as *2word12* passwords. The 2word approach seems more effective when combined with a length-16 requirement, perhaps because this leads more participants to create passphrases.

To understand how participants perceived the strength of their study passwords, we asked whether they agreed with the statement, “If my main email provider had the same password requirements as used in this study, my email account would be more secure.” Agreement ranged

Cracked after 10^8 guesses					Cracked after 10^{12} guesses							
Omnibus $\chi^2=205.59, p<.001$					Omnibus $\chi^2=838.465, p<.001$							
cond 1	%	cond 2	%	p-value	cond 1	%	cond 2	%	p-value			
<i>basic12</i>	9.6%	<i>2word12</i>	6.6%	.021	<i>basic12</i>	28.8%	<i>comp8</i>	24.8%	.019			
		<i>3class12</i>	4.8%	<.001			<i>2word12</i>	24.2%	.012			
		<i>basic20</i>	4.5%	<.001			<i>3class12</i>	17.1%	<.001			
		<i>comp8</i>	3.5%	<.001			<i>basic16</i>	13.1%	<.001			
		<i>2word16</i>	2.0%	<.001			<i>basic20</i>	7.5%	<.001			
		<i>3class16</i>	0.8%	<.001			<i>2word16</i>	6.7%	<.001			
<i>basic16</i>	7.0%	<i>basic20</i>	4.5%	.02	<i>3class16</i>	2.7%	<.001	<i>comp8</i>	24.8%	<i>3class12</i>	17.1%	<.001
		<i>comp8</i>	3.5%	<.001	<i>basic16</i>	13.1%	<.001					
		<i>2word16</i>	2.0%	<.001	<i>basic20</i>	7.5%	<.001					
		<i>3class16</i>	0.8%	<.001	<i>2word16</i>	6.7%	<.001					
<i>2word12</i>	6.6%	<i>comp8</i>	3.5%	<.001	<i>3class16</i>	2.7%	<.001	<i>2word12</i>	24.2%	<i>3class12</i>	17.1%	<.001
		<i>2word16</i>	2.0%	<.001	<i>basic16</i>	13.1%	<.001					
		<i>3class16</i>	0.8%	<.001	<i>basic20</i>	7.5%	<.001					
<i>3class12</i>	4.8%	<i>2word16</i>	2.0%	<.001	<i>2word16</i>	6.7%	<.001	<i>3class12</i>	17.1%	<i>basic16</i>	13.1%	<.001
		<i>3class16</i>	0.8%	<.001	<i>3class16</i>	2.7%	<.001			<i>basic20</i>	7.5%	<.001
<i>basic20</i>	4.5%	<i>2word16</i>	2.0%	<.001	<i>2word16</i>	6.7%	<.001	<i>basic20</i>	7.5%	<i>3class16</i>	2.7%	<.001
		<i>3class16</i>	0.8%	<.001	<i>3class16</i>	2.7%	<.001					
<i>comp8</i>	3.5%	<i>3class16</i>	0.8%	<.001	<i>3class16</i>	2.7%	<.001	<i>3class16</i>	2.7%	<.001		
<i>2word16</i>	2.0%	<i>3class16</i>	0.8%	.043	<i>basic16</i>	13.1%	<i>basic20</i>	7.5%	<.001			
					<i>2word16</i>	6.7%	<.001	<i>2word16</i>	6.7%	<.001		
					<i>3class16</i>	2.7%	<.001	<i>3class16</i>	2.7%	<.001		
					<i>basic20</i>	7.5%	<.001	<i>3class16</i>	2.7%	<.001		
					<i>2word16</i>	6.7%	<.001	<i>3class16</i>	2.7%	<.001		

Table 7.2: Significant differences in the probability of passwords cracked after 10^8 and 10^{12} guesses, representing more and less resource-constrained attackers. Figure 7.1 illustrates these guess numbers along a curve. In both tables, the more secure condition is in the cond 2 column.

from 59.8% for *3class16* and 59.7% for *comp8* to 35.2% for *basic12*. Participants in *comp8* were more likely to agree than in any other condition except *3class16*. This suggests that user perception of password strength does not align with our strength analysis. We suspect that participants expected the *comp8* requirements to lead to strong passwords because it is similar to a traditional “strong” policy. Future work might investigate the security and usability impact of helping users better understand how password-composition requirements can lead to stronger passwords.

7.2.4 Usability Results

In this section, we examine dropout rates, as well as password storage, creation, and recall. Overall, we found that most conditions are significantly more usable than *comp8* on a number of metrics, with only *basic20* and *3class16* being significantly less usable on any metric.

Condition	Part One completion (%)	Password storage (%)	Mean creation attempts	Agree creation difficult (%)	Part One recall attempts	Part One entry time (s)	Part Two recall attempts	Part Two entry time (s)	Agree recall difficult (%)	Cracked@10 ⁸ (%)	Cracked@10 ¹² (%)
	Study I										
<i>comp8</i>	83.0	56.9	2.4	32.8	1.1	7.1	1.7	13.2	39.3	3.5	24.8
<i>basic12</i>	94.5	45.4	1.5	15.2	1.1	6.2	1.6	11.6	27.4	9.6	28.8
<i>basic16</i>	93.9	49.9	1.8	28.5	1.1	7.4	1.6	13.7	30.1	7.0	13.1
<i>basic20</i>	93.9	50.0	1.9	35.2	1.2	9.0	1.6	15.3	32.9	4.5	7.5
<i>2word12</i>	92.0	51.4	1.9	21.9	1.1	6.8	1.6	13.1	31.0	6.6	24.2
<i>2word16</i>	92.1	51.3	2.1	34.7	1.1	8.4	1.7	14.6	36.8	2.0	6.7
<i>3class12</i>	92.0	54.9	1.5	26.0	1.1	7.4	1.7	14.8	35.3	4.8	17.1
<i>3class16</i>	90.5	60.2	1.9	40.3	1.1	8.8	1.7	16.2	42.9	0.8	2.7

Table 7.3: A summary of findings of study I results are shown first, with each condition compared to *comp8*. Lighter blue indicates being significantly better than the control, and darker red indicates being worse. No shading indicates no significant difference. This table is repeated, along with data for Study II, in Table 7.10.

Study Dropout

We consider higher dropout rates to indicate increased participant frustration. Among 15,108 participants who began the study, 91.0% finished Part One. Part One completion varied significantly by condition ($\chi^2_7=246.60$, $p<.001$), ranging from 83.0% for *comp8* to 94.5% for *basic12*. Participants in *comp8* were significantly less likely to finish Part One than those in any other condition (HC χ^2 , $p<.001$). This suggests participants assigned to *comp8* experienced more negative sentiment than in other conditions. They may have found the study more confusing, boring, or frustrating. Participants in *3class16* (90.5%) were significantly less likely to finish Part One than those in *basic12* (94.5%) or *basic16* (93.9%) (HC χ^2 , $p<.004$). Of those participants who finished Part One, 62.3% returned within three days of being invited back; this did not vary significantly by condition ($\chi^2_7=7.69$, $p=0.361$). Of those who returned for Part Two, 95.1% completed Part Two within three days of being invited back; this also did not vary significantly by condition ($\chi^2_7=4.15$, $p=0.762$).

Password Storage

Our analysis of password storage looked at participants who finished Part Two, because we asked participants about their storage behavior only in the Part Two survey. 52.6% of participants were *storage* participants. This ranged from 45.4% for *basic12* to 60.2% for *3class16*; Table 7.5 shows significant pairwise differences. *3class16* had a significantly higher storage rate than every other condition except *comp8* and *3class12*. Password storage rates were highest in conditions that required three or four character classes, and lowest in the basic conditions.

Password creation attempts					Agree password creation difficult				
Omnibus KW $\chi^2=394.337, p<.001$					Omnibus $\chi^2=239.44, p<.001$				
cond 1	mean	cond 2	mean	p-value	cond 1	%	cond 2	%	p-value
<i>comp8</i>	2.4	<i>basic20</i>	1.9	U=1809027, .011	<i>3class16</i>	40.3%	<i>basic20</i>	35.2%	.019
		<i>3class16</i>	1.9	U=1797307.5, <.001			<i>2word16</i>	34.7%	.007
	<i>2word12</i>	1.9	U=1824868, <.001	<i>comp8</i>		32.8%	<.001		
	<i>basic16</i>	1.8	U=1993010.5, <.001	<i>basic16</i>		28.5%	<.001		
	<i>3class12</i>	1.7	U=1992868, <.001	<i>3class12</i>		26.0%	<.001		
	<i>basic12</i>	1.5	U=2137340, <.001	<i>2word12</i>		21.9%	<.001		
<i>2word16</i>	2.1	<i>3class16</i>	1.9	U=1458583.5, <.001	<i>basic12</i>	15.2%	<.001		
		<i>2word12</i>	1.9	U=1481683, <.001	<i>basic20</i>	35.2%	<i>basic16</i>	28.5%	<.001
	<i>basic16</i>	1.8	U=1623889, <.001	<i>3class12</i>	26.0%	<.001			
	<i>3class12</i>	1.7	U=1635127.5, <.001	<i>2word12</i>	21.9%	<.001			
<i>basic20</i>	1.9	<i>basic12</i>	1.5	U=1764158, <.001	<i>basic12</i>	15.2%	<.001		
		<i>3class16</i>	1.9	U=1476928.5, .011	<i>2word16</i>	34.7%	<i>basic16</i>	28.5%	<.001
	<i>2word12</i>	1.9	U=1499017.5, .023	<i>3class12</i>	26.0%	<.001			
	<i>basic16</i>	1.8	U=1645027.5, <.001	<i>2word12</i>	21.9%	<.001			
<i>3class16</i>	1.9	<i>3class12</i>	1.7	U=1666130, <.001	<i>basic12</i>	15.2%	<.001		
		<i>basic12</i>	1.5	U=1803101.5, <.001	<i>comp8</i>	32.8%	<i>basic16</i>	28.5%	.027
	<i>3class12</i>	1.7	U=1487448, <.001	<i>3class12</i>	26.0%	<.001			
<i>2word12</i>	1.9	<i>basic12</i>	1.5	U=1607634, <.001	<i>2word12</i>	21.9%	<.001		
		<i>3class12</i>	1.7	U=1534292.5, <.001	<i>basic12</i>	15.2%	<.001		
<i>basic16</i>	1.8	<i>basic12</i>	1.5	U=1660203.5, <.001	<i>basic16</i>	28.5%	<i>2word12</i>	21.9%	<.001
		<i>3class12</i>	1.7	U=1577844.5, <.001	<i>basic12</i>	15.2%	<.001		
<i>3class12</i>	1.7	<i>basic12</i>	1.5	U=1712210.5, <.001	<i>3class12</i>	26.0%	<i>2word12</i>	21.9%	.032
		<i>basic12</i>	1.5	U=1483504.5, .006	<i>basic12</i>	15.2%	<.001		
					<i>2word12</i>	21.9%	<i>basic12</i>	15.2%	<.001

Table 7.4: Significant differences in study I password-creation usability, with the significantly more usable condition in the cond 2 column. The left-hand side compares password-creation attempts. The right-hand side compares agreement with password creation being difficult.

Password Creation

We interpret taking more password-creation attempts as being less usable. Participants took an average of 1.9 attempts to create a password. Table 7.4 shows significant pairwise differences. *comp8* took the most attempts, (mean=2.4), and *basic12* took the fewest (mean=1.5). To compare their sentiment, we asked participants whether they agreed with the statement, “Creating a password that meets the requirements given in this study was difficult”. Figure 7.2 depicts responses and Table 7.4 shows significant differences. Agreement ranged from 15.2% for *basic12* to 40.3% for *3class16*.

Creation Failure

For a better understanding of password-creation failures, we looked at participants’ first failed attempt. Table 7.1 shows these failures. Participants often failed to meet length or character-class

Password entry time (s)					Agree remembering password difficult					
Omnibus KW $\chi^2=71.58, p<.001$					Omnibus $\chi^2=83.89, p<.001$					
cond 1	med	cond 2	med	p-value	cond 1	%	cond 2	%	p-value	
<i>3class16</i>	16.2	<i>comp8</i>	13.2	U=65587.5, .001	<i>3class16</i>	42.9	<i>3class12</i>	35.3	.012	
			13.1	U=60727, <.001					32.9	<.001
			11.6	U=75507, <.001					31.0	<.001
<i>basic20</i>	15.3	<i>comp8</i>	13.2	U=86932, <.001	<i>basic16</i>		<i>basic12</i>	30.1	<.001	
			13.1	U=80633.5, <.001					27.4	<.001
			11.6	U=100385, <.001		<i>comp8</i>		39.3	<i>basic20</i>	32.9
<i>3class12</i>	14.8	<i>basic12</i>	11.6	U=81122.5, .001			<i>2word12</i>	31.0	.001	
					<i>basic16</i>		30.1	<.001		
<i>2word16</i>	14.6	<i>2word12</i>	13.1	U=69022.5, .012		<i>basic12</i>	27.4	<.001		
				11.6	U=86435.5, <.001	<i>2word16</i>	36.8	<i>basic16</i>	30.1	.03
<i>basic16</i>	13.7	<i>basic12</i>	11.6	U=95571.5, .003				<i>basic12</i>	27.4	<.001
						<i>3class12</i>	35.3	<i>basic12</i>	27.4	.002

Password storage				
Omnibus $\chi^2=61.87, p<.001$				
cond 1	%	cond 2	%	p-value
<i>3class16</i>	60.2	<i>2word12</i>	51.4	.002
			51.3	.002
			50.0	<.001
			49.9	<.001
			45.4	<.001
<i>comp8</i>	56.9	<i>basic20</i>	50.0	.029
			49.9	.02
			45.4	<.001
<i>3class12</i>	54.9	<i>basic12</i>	45.4	<.001

Table 7.5: These show significant differences in study I password-recall usability, with the significantly more usable condition on the cond 2 column. This includes Part Two recall time for participants who recalled their password in five attempts without a reminder, password storage rates for Part Two participants, and agreement with password recall being difficult.

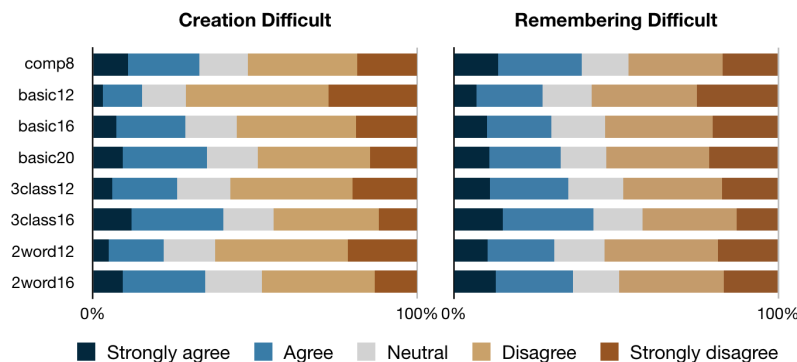


Figure 7.2: Participant agreement with “Creating a password that meets the requirements given in this study was difficult” and “Remembering the password I used for this study was difficult.” Significant differences are in Tables 7.4 and 7.5.

requirements.

Many participants failed to meet length requirements, with 57.3% of participants in *basic20* using less than 20 characters. 26.3% of participants in *comp8* used too few character classes, compared to between nine and ten percent of participants in other conditions that required non-letter characters. This suggests that participants struggle more to create a password with four classes compared to three. The largest source of failure in *comp8* was the dictionary check. Only *comp8* had a dictionary check, so we looked at how many passwords in other conditions would have been prevented by that check. These numbers are included in the same column in Table 7.1 with an asterisk. This was 23.4% of passwords in *3class12*, 18.5% in *basic12*, and less than 10% in any other condition. Recall that the dictionary check strips away non-letter characters and checks the remaining letters against a dictionary. We speculate that the longer conditions were less likely to fail the dictionary check because longer passwords tend to have more letters than a single dictionary word.

Part One Recall

Participants recalled their passwords after filling out a brief survey in Part One. 93.5% of participants correctly entered their password on the first attempt and this varied by condition ($\chi^2_7=27.241$, $p<0$). Participants in *basic12* (95.7%) were significantly more likely to enter their passwords correctly on the first try than those in *2word16* (92.9%), *3class12* (93.1%), *3class16* (92.1%), or *basic20* (92.5%) (HC FET, $p<.038$).

Part Two Recall

In Part Two recall, participants could use a password reminder to display their password. 15.5% of participants used this feature, and this did not vary significantly by condition ($\chi^2_7=8.31$, $p=0.306$). Among no-storage participants, 21.4% used the reminder, and this also did not vary by condition ($\chi^2_7=7.72$, $p=0.358$). 80.1% of participants successfully entered their password in five attempts without using the reminder, and this did not vary significantly by condition ($\chi^2_7=7.75$, $p=0.356$). These participants took an average of 1.3 tries, and this also did not vary significantly by condition ($\chi^2_7=12.96$, $p=0.073$).

As a measure of usability, we looked at how long participants spent entering their passwords on their first successful attempt. We looked only at no-storage participants who did not use the reminder. Median times varied from *basic12* (11.6 seconds) to *3class16* (16.2 seconds), with significant differences in Table 7.5. Overall, participants in *3class16* took the most time to recall their passwords successfully. Participants in *basic12* and *2word12* took the least time to enter their passwords on the first successful attempt, despite their being longer than *comp8*.

To measure subjective user difficulty, we asked participants whether they agreed with the statement, “Remembering the password I used for this study was difficult.” The least difficult condition for recall was *basic12* (27.4%), and the most difficult were *comp8* (39.3%) and *3class16* (42.9%). Figure 7.2 depicts the results, and Table 7.5 shows significant differences.

Substring	Using	Cracked Using	Cracked -Using	<i>p</i> -value
1234	4.9%	44.5%	14.4%	< .001
password	3.0%	44.1%	15.0%	< .001
love	1.9%	15.3%	15.9%	.864
123456789	1.7%	47.5%	15.3%	< .001
2013	1.6%	20.4%	15.8%	.132
this	1.6%	21.1%	15.8%	.124
turk	1.5%	39.6%	15.5%	< .001
char	1.1%	40.0%	15.6%	< .001

Table 7.6: Substrings in at least 1% of passwords. The first column shows the percent of passwords containing the substring. The next two show percentages of passwords cracked containing and not containing it. The fourth shows a χ^2 test on the difference. The presence of “2013” likely results from the study being conducted in that year.

7.2.5 Password Patterns

This section looks at common themes and patterns in study passwords, to help explain how users created passwords under different requirements. We identified markers of weak passwords, which can help to identify weak passwords during creation. We found a small set of common password substrings that indicate weaker passwords. We examined whether and how participants exceed minimum password requirements. And we manually examined passwords and found a number of common themes, such as love and animals.

Common Substrings

We identified the most common substrings in the study’s passwords. We first made a list of all substrings of 4 to 12 characters present in at least one percent of the passwords. We removed any substrings that did not exist in at least one percent of study passwords without already being part of another, longer substring on the list. For example, we removed “sword,” which was almost always present as part of “password.” This left the eight substrings in Table 7.6. Overall, 1,944 passwords (14.1%) contain at least one of the eight substrings. We looked at password-cracking rates for passwords with and without each substring. Table 7.6 shows the five substrings such that passwords containing any of them were significantly more likely to be cracked. This finding suggests future research on proactively checking prospective passwords and rejecting any password that contains a substring associated with weak passwords.

Meeting the *comp8* Requirements

29.0% of passwords in *comp8* fulfilled the symbol requirement only by placing “!” at the end of the password. Likewise, 57.7% of passwords in *comp8* used an uppercase letter as their first character and used no other uppercase letter. Passwords doing either of these were significantly more likely to be cracked (34.6% to 6.5% cracked) ($\chi^2_1=190.4864$, $p<.001$). This suggests that *comp8* can be a much more effective condition when its requirements are not met in minimal ways.

Going Beyond the Requirements

Table 7.1 shows that participants often exceeded minimum length and character-class requirements. Each condition has a median length above its minimum, and all conditions have a median of at least two digits. 66.4% of participants exceeded their minimum required length, ranging from 59.3% of participants in *basic12* to 76.8% in *comp8*. Perhaps not surprisingly, passwords that exceeded the minimum length requirements were less likely to be cracked (12.0% to 23.4%) ($\chi^2_1=295.9375$, $p<.001$).

We also looked at exceeding the minimum number of character classes, omitting *comp8* because it already required all four character classes. 62.4% of non-*comp8* participants used more than the minimum number of character classes. 38.7% of participants in *2word16* and 38.5% in *2word12* used at least three character classes. Over two thirds of passwords in each of the basic and 3class conditions exceeded their minima. Passwords exceeding the minimum were significantly less likely to be cracked, 10.2% to 21.3% ($\chi^2_1=277.5778$, $p<.001$).

Character Distribution in 3class12

Participants often responded to the three-character-class requirement of *3class12* by placing characters other than lowercase letters at the beginning or end of their passwords. In fact, fewer than 2% of passwords in *3class12* – 33 of 1653 – began and ended with lowercase letters. Those 33 passwords were particularly difficult to guess. After 10^8 guesses, none were guessed. After 10^{12} guesses, only one of them was guessed (“family-4ever”). This suggests that encouraging or requiring users to distribute their different character classes more evenly might make character-class requirements more effective.

Semantic Analysis

To gain a better understanding of the semantic content of user-generated passwords, we manually looked at 100 randomly chosen passwords per condition from participants who finished Part Two. Participants who included words were more likely to place non-letter characters between words, rather than within them. We found that names, dates, and sequences of characters (such as “1234” and “qwerty”) were common. We also saw a number of study-related words, as well as references to animals, love, and pop culture. Consistent with those themes, looking at all passwords and ignoring case, 42 passwords contained “monkey” and 294 passwords contained “love.” Future research might explore encouraging participants to choose words from a wider range of themes, and to add special characters within words.

7.3 Using Findings from Study I to Create Study II Conditions

This section discusses how the findings of study I led to the conditions of study II. Table 7.10 summarizes security and usability findings for study I, comparing each other condition to *comp8*. Both *3class12* and *2word16* performed significantly better than *comp8* on several usability metrics, were more secure after 10^{12} guesses, and did not perform worse than *comp8* on any metric. Comparing *3class12* and *2word16*, passwords in *2word16* were more difficult to guess. However, passwords in *3class12* were easier to create, and were still significantly more secure than those

in *comp8*. In addition, *3class12* is more similar to policies commonly found in the wild, which may help make it a better baseline. Therefore, study II used *3class12* as its baseline condition. An exploration of conditions based on *2word16* remains promising future work.

Section 7.2.5 showed that a small set of substrings were markers of a password being more likely to be cracked. For example, 47.5% of passwords containing “123456789” were cracked, compared to 15.3% without that substring. Therefore, we introduced the *blacklist* requirement, which requires that passwords not contain any blacklisted substrings. While users may not be familiar with the concept of a substring blacklist, we hoped that familiarity with common dictionary checks would help them grasp it. Also, in practice, the blacklist requirement using a short substring blacklist would be more feasible to conduct client-side than a traditional dictionary check with a large dictionary. We present the blacklist we used in study II below. 69.2% of study I passwords would have passed this blacklist check, ranging from 57.4% of *basic20* to 83.0% of *comp8*.

As indicated in Section 7.2.5, fewer than 2% of *3class12* passwords began and ended with lowercase letters, but those that did were especially difficult to guess. We call the requirement that passwords begin and end with a lowercase letter the *pattern* requirement. We hypothesized that users creating a password under the pattern requirement would more evenly distribute non-lowercase letters throughout their password.

2class12, *3class12*— These conditions required 12 characters and two and three character classes. In study I, *3class12* was more usable and secure than *comp8*. We expected that a condition adding further requirements to *3class12* would be less usable than *3class12* itself. Therefore, we created *2class12*. The other conditions in study II were created by adding requirements to *2class12*.

2class16— This condition extended the length requirement of *2class12* by four characters. *3class16* passwords were difficult for participants to create and recall, but they were stronger than in any other study I condition. We were interested in whether reducing the character-class requirements could make the policy more usable.

2list12, *2s-list12*— These conditions combined the blacklist requirement with the requirements of *2class12*. Participants in *2list12* saw an explicit list of blacklisted substrings, and participants in *2s-list12* were simply asked to avoid using common substrings.³ We used the following blacklist.

```
123!, amazon, character, monkey, number, survey, this, turk
Any year between 1950 and 2049
The same character four or more times in a row
Any four consecutive characters from password
Any four sequential digits (e.g., 5678)
Any four sequential letters in the alphabet (e.g., wxyz)
Any four consecutive characters on the keyboard (e.g., wsxc)
```

2pattern12— This condition added the pattern requirement to *2class12*. As described above, the pattern required was that the password started and ended with a lowercase letter.

2list-patt12, *2s-list-patt12*— These conditions combined the requirements of *2list12* and *2s-list12* with the pattern requirement.

³Do not include words commonly found in passwords (e.g. *password*), keyboard patterns (e.g. *qazx*), or other common patterns (e.g. *5678*)

Condition	Partic- ipants	Len. (med- ian)	Up. (med- ian)	Low. (med- ian)	Digit (med- ian)	Sym. (med- ian)	Fail (%)	Length (%)	Class (%)	Blacklist (%)	Pattern (%)
<i>3class12</i>	1121	13	1	8	3	1	43.0	37.6	8.4	35.6*	98.0*
<i>2class12</i>	1131	13	1	8	3	1	41.2	37.2	1.7	32.8*	97.3*
<i>2class16</i>	1096	17	1	12	3	1	51.9	47.5	2.0	38.3*	96.2*
<i>2list12</i>	1113	13	1	8	3	1	46.7	29.4	1.2	16.3	96.0*
<i>2s-list12</i>	1099	13	1	8	3	1	52.3	33.5	1.7	19.6	97.0*
<i>2pattern12</i>	1076	14	1	9	2	1	70.5	35.2	1.3	28.5*	59.1
<i>2list-patt12</i>	1059	14	1	9	2	1	69.1	27.9	1.8	13.3	55.9
<i>2s-list-patt12</i>	1045	14	1	9	2	1	76.7	35.3	2.6	21.8	58.1

Table 7.7: Password attributes and creation failure on the first try. Length and character counts are medians. A password can fail in multiple ways. We omit failure from blank fields and confirmation mismatch. “Blacklist” and “Pattern” show percents of participants who failed those checks for conditions with those checks. For other conditions (marked with *), they show the percentage of final passwords that would have failed.

7.4 Study II Findings

This section presents the findings of the second of our two studies. Section 7.4.1 presents participant demographics. We present security results in Section 7.4.2 and usability results in Section 7.4.3. Section 7.4.4 discuss patterns we observed in cracked passwords.

7.4.1 Participants

We collected data from December 2013 to January 2014. 9,707 participants began the study, 8,740 finished Part One, and 5,111 returned for Part Two within three days of being notified. Table 7.7 shows the number of participants per condition. Participant age did not vary significantly by condition ($\chi^2_7=10.069$, $p=0.185$) (mean=30.59, standard deviation = 10.45, median = 28). Reported gender did not vary significantly either ($\chi^2_1=4.821$, $p=0.682$) (47.9% male, 51.2% female).

7.4.2 Security Results

Figure 7.3 depicts the proportion of passwords in each condition that were guessed as the number of guesses increased. Table 7.10 shows significant differences in after 10^8 and 10^{12} guesses. After 10^{12} guesses, *3class12* and *2class12* performed significantly worse than any other conditions. *2list12* and *2s-list12* did not differ from each other but were worse than the remaining four conditions. *2pattern12* performed worse than the remaining three conditions. Those three conditions – *2class16*, *2list-patt12*, and *2s-list-patt12* – performed significantly better than any other condition, and did not differ significantly from one another. Strength after 10^{12} guesses divides the conditions into four clusters, consistent with a visual inspection of Figure 7.3.

Table 7.8 shows cracking differences after 10^8 guesses. This smaller number of guesses simulates a more resource-constrained attacker, or a scenario in which the service provider quickly detects a breach and resets its passwords. *3class12* and *2class12* performed worse than the blacklist

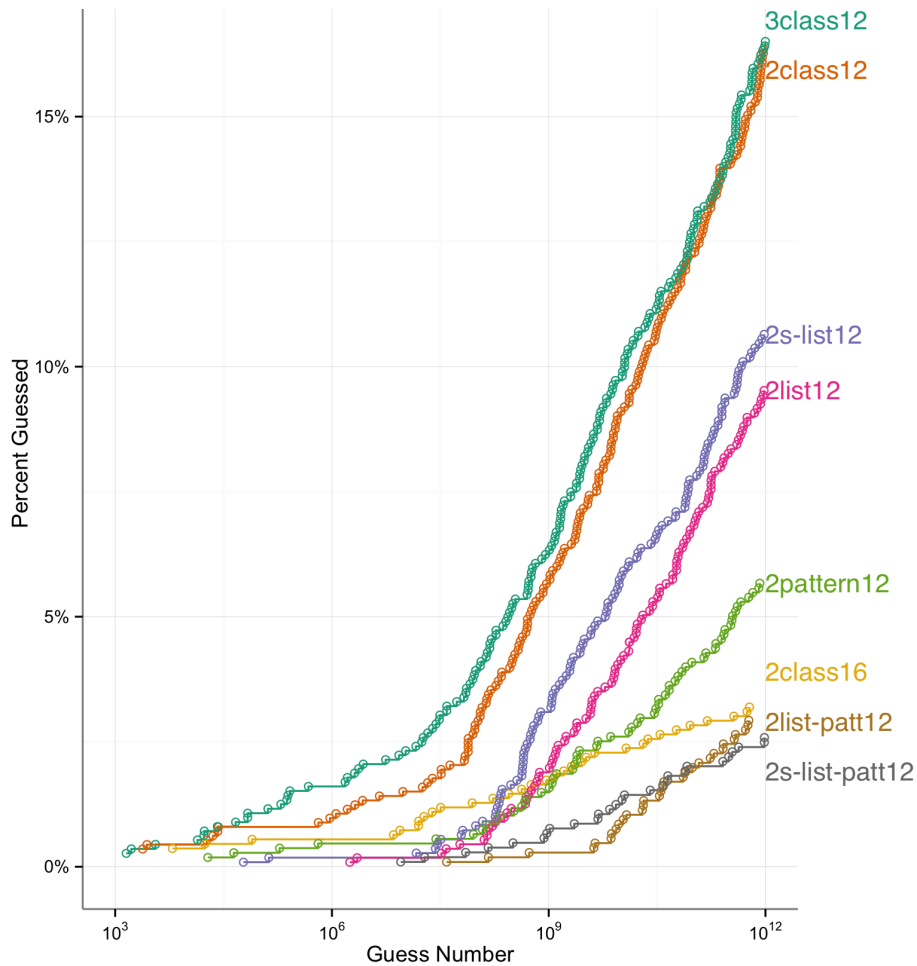


Figure 7.3: The percentage of passwords cracked in each condition by the number of guesses made in log scale. Our cutoff for guess numbers was 10^{12} . Table 7.8 shows significant differences in cracking rates between conditions.

or pattern conditions, though fewer than five percent of passwords in any condition were guessed.

7.4.3 Usability Results

Overall, *2class12* and *3class12* had similar usability metrics and were more usable than the other conditions. The blacklist check made password creation more difficult but did not affect recall. The pattern requirement negatively affected creation and recall usability.

Study Dropout

Table 7.8 shows significant differences in Part One dropout rates among all participants who began the study. Participants in the pattern conditions were the most likely to drop out, which may

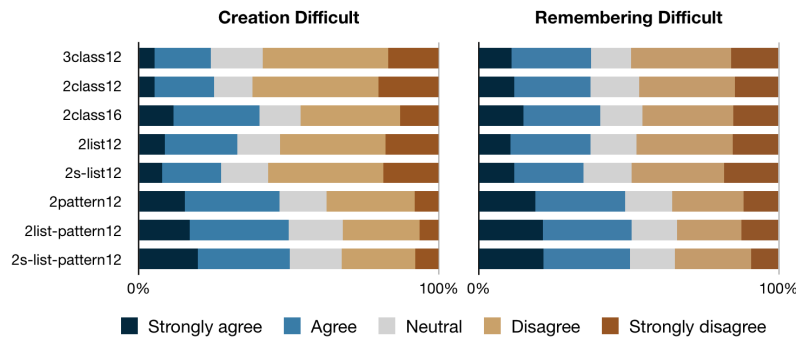


Figure 7.4: Participant agreement with “Creating a password that meets the requirements given in this study was difficult” and “Remembering the password I used for this study was difficult.”

suggest they were more frustrated. There was no significant difference in rates of returning for Part Two ($\chi^2_7=5.826$, $p=0.56$) or completing Part Two ($\chi^2_7=5.97$, $p=0.543$).

Password Storage

58.5% of part-two participants stored their password. Table 7.9 shows significant differences across conditions. Participants in the pattern conditions were the most likely to store their passwords, which may indicate actual or expected recall difficulty.

Password Creation

Table 7.8 shows significant differences in password-creation attempts. Participants in the pattern conditions took significantly more attempts than in any non-pattern condition. Fewer than one third of participants in pattern conditions successfully created a password on the first try. In the other conditions, this ranged from 48% for *2s-list12* to 59% for *2class12*.

As we did in the first study, to learn about perceived password-creation difficulty, we asked participants whether they agreed with the statement, “Creating a password that meets the requirements given in this study was difficult.” Figure 7.4 depicts responses and Table 7.9 shows pairwise differences. Password creation under the the pattern conditions was usually more difficult than under the non-pattern conditions.

Anecdotal evidence from our previous research suggested that users were fond of incorporating the term “monkey” into their passwords. To examine this more scientifically, we detected passwords with text matching or similar to “monkey.” After these participants finished Part One, we asked, “We couldn’t help but notice that you have a Monkey-ish word in your password. Please tell us why you included [text] in your password.” We detected 17 passwords with a monkey-ish phrase. Participants reported liking monkeys, finding them cute, and having “monkey” be a nickname or pet name. This is further anecdotal evidence that users often make passwords related to things they like.

Creation Failure

Table 7.7 shows types of password-creation failures on the first attempt. Participants struggled to meet the pattern requirement. Over two-thirds of participants with this requirement failed on their first attempt, and over half of these failures were due to the pattern requirement itself. The blacklist requirement appears to have prevented fewer passwords than the dictionary check of *comp8*.

Part One Recall

In Part One recall, 93.2% of participants entered their passwords correctly on the first attempt. This was not significantly different between conditions ($\chi^2_7=5.101, p=.648$). Table 7.9 shows significant differences in password-entry time for participants who correctly entered their password on the first try. Passwords in *2class16* took longest to enter, followed by *2pattern12*. *2class12* took significantly less time than any other condition. However, the effect size is small. Median times ranged from *2class12* (7.2 seconds) to *2class16* (8.9 seconds).

Part Two Recall

5,111 participants returned and finished Part Two within three days of being invited back. 14.9% of them used the reminder and this did not vary significantly by condition ($\chi^2_7=6.833, p=0.446$). 80.0% of participants entered their password correctly within five attempts without the reminder, and this also did not vary by condition ($\chi^2_7=5.401, p=0.611$). Among these success participants, the number of recall attempts did not vary significantly ($\chi^2_7=3.009, p=0.884$). Among the 2,120 no-storage participants, there was also no significant difference in using the reminder ($\chi^2_7=9.727, p=0.205$) or successfully recalling the passwords ($\chi^2_7=9.518, p=0.218$).

To understand perceived difficulty, we asked participants whether they agreed with, “Remembering the password I used for this study was difficult.” While the above observed metrics for password recall did not vary by condition, perceived difficulty did. Agreement was 47.4% to 49.1% for pattern conditions and 38.5% to 32.6% for the non-pattern conditions. Table 7.9 shows significant differences. Each pattern condition had significantly more difficulty than any non-pattern condition.

7.4.4 Password Patterns

This section examines how participants in study II met and exceeded their password-composition requirements.

Common Substrings

Analyses presented in Section 7.2.5 found eight substrings that were present in at least one percent of study I passwords. This led to the blacklist requirement for some study II conditions. The substring “love” is in 1.4% of passwords created in conditions with the blacklist requirement. No other substring was in one percent or more of these passwords. This substring was not correlated with passwords being more likely to be cracked (FET, $p = .743$). This shows that the blacklist requirement did help prevent participants from using common substrings in their passwords. This

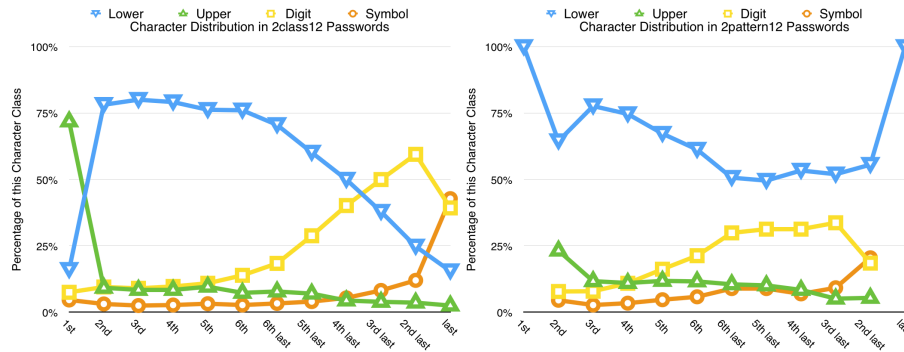


Figure 7.5: For the first and last six characters, this depicts the percentage of each character class in *2class12* and *2pattern12*.

may help explain why *2s-list12* and *2list12* passwords were less likely to be cracked than *2class12* passwords.

Going Beyond the Requirements

65.4% of passwords exceeded their minimum length requirement of 12 or 16 characters. They were significantly less likely to be cracked than passwords that did not exceed the minimum (5.7% to 14.0%) ($\chi^2_1=174.0752$, $p<.001$). 84.3% of passwords exceeded their minimum character-class requirement. These passwords were significantly less likely to be cracked than passwords that did not exceed the minimum (7.1% to 16.2%) ($\chi^2_1=120.4505$, $p<.001$).

The Pattern Requirement and Character Distribution

The pattern requirement was intended to cause participants to distribute non-lowercase-letter characters throughout their passwords, rather than putting most of them at the beginning or end. To measure how effective this was, we compared character-class distributions in *2pattern12* and *2class12* passwords, which differed only in having the pattern requirement.

The *structure* of a password is a representation of its character classes [98]. For example, the structure of password “P4ssword!” is “UDLLLLLS”. We examined how often passwords in *2class12* and *2pattern12* had unique patterns. Because *2class12* had more participants, we did not make a direct comparison of the number of unique passwords in each condition. We instead took a random sample of 1000 participants from each condition and counted how many of them had structures unique among the 1000. We repeated this experiment 1000 times. For *2class12*, an average of 63.4% of passwords had unique structures. For *2pattern12*, it was 81.8%. This is evidence that the pattern requirement leads participants to more diverse password structures.

Figure 7.5 visualizes character-class distributions of *2class12* and *2pattern12*. It depicts character classes of the first and last six characters because those conditions both require 12 characters. Passwords in *2class12* have spikes of special characters at the start and end. While *2pattern12* passwords do not have an even distribution of character classes, they appear to be better mixed than in *2class12*. This may explain how the pattern requirement led to stronger passwords.

Cracked after 10⁸ guesses Omnibus $\chi^2_7=112.708, p<.001$					Part One dropout rate Omnibus $\chi^2_7=58.579, p<.001$											
cond 1	%	cond 2	%	p-value	cond 1	%	cond 2	%	p-value							
<i>3class12</i>	3.9	<i>2class16</i>	1.2	<.001	<i>2s-list-patt12</i>	14.0	<i>2class16</i>	9.6	.021							
		<i>2s-list12</i>	0.8	<.001			<i>2s-list12</i>	9.5	<.013							
		<i>2pattern12</i>	0.7	<.001			<i>2list12</i>	8.1	<.001							
		<i>2list12</i>	0.4	<.001			<i>3class12</i>	8.0	<.001							
		<i>2s-list-patt12</i>	0.3	<.001			<i>2class12</i>	6.7	<.001							
		<i>2list-patt12</i>	0.1	<.001			<i>2list-patt12</i>	12.6	<i>2list12</i>	8.1	.008					
<i>2class12</i>	2.8	<i>2s-list12</i>	0.8	.007	<i>3class12</i>	8.0			.005							
		<i>2pattern12</i>	0.7	.001	<i>2class12</i>	6.7			<.001							
		<i>2list12</i>	0.4	<.001	<i>2pattern12</i>	11.3	<i>2class12</i>	6.7	.002							
		<i>2s-list-patt12</i>	0.3	<.001												
<i>2list-patt12</i>	0.1	<.001	<i>2class16</i>	1.2	<i>2list-patt12</i>	0.1	.031									
Cracked after 10¹² guesses Omnibus $\chi^2_7=328.517, p<.001$					Password creation attempts Omnibus KW $\chi^2_7=795.632, p<.001$											
cond 1	%	cond 2	%	p-value	cond 1	count	cond 2	count	p-value							
<i>3class12</i>	16.5	<i>2s-list12</i>	10.6	<.001	<i>2s-list-patt12</i>	2.6	<i>2list-patt12</i>	2.4	U=604263.5, .001							
							<i>2pattern12</i>	2.4	U=612719, .001							
							<i>2s-list12</i>	1.9	U=754699, <.001							
							<i>2class16</i>	1.8	U=767909.5, <.001							
							<i>2list12</i>	1.8	U=791452, <.001							
							<i>3class12</i>	1.6	U=837478, <.001							
<i>2class12</i>	16.4	<i>2s-list12</i>	10.6	<.001	<i>2s-list-patt12</i>	2.4	<i>2s-list12</i>	1.9	U=863035, <.001							
							<i>2class16</i>	1.8	U=710967.5, <.001							
							<i>2list12</i>	1.8	U=723734, <.001							
							<i>3class12</i>	1.6	U=748822, <.001							
							<i>2class12</i>	1.6	U=794217.5, <.001							
							<i>2class12</i>	1.6	U=819334.5, <.001							
<i>2s-list12</i>	10.6	<i>2pattern12</i>	5.7	<.001	<i>2s-list-patt12</i>	2.4	<i>2s-list12</i>	1.9	U=726433, <.001							
							<i>2class16</i>	1.8	U=739575.5, <.001							
							<i>2list12</i>	1.8	U=765386, <.001							
							<i>3class12</i>	1.6	U=812124.5, <.001							
							<i>2class12</i>	1.6	U=838019.5, <.001							
							<i>2s-list12</i>	1.9	U=688868.5, <.001							
<i>2list12</i>	9.5	<i>2pattern12</i>	5.7	.006	<i>2s-list-patt12</i>	1.9	<i>3class12</i>	1.6	U=713024.5, <.001							
							<i>2class16</i>	1.6	U=675942, <.001							
							<i>2list-patt12</i>	2.9	<.001							
							<i>2class12</i>	1.6	U=700279.5, <.001							
							<i>2pattern12</i>	5.7	<i>2class16</i>	3.2	.038	<i>2s-list-patt12</i>	1.8	<i>3class12</i>	1.6	U=661462.5, .036
														<i>2list-patt12</i>	2.9	.013
<i>2class12</i>	1.6	U=684801, <.001														

Table 7.8: Significant differences between conditions in study II. The left-hand side shows differences in proportions of passwords cracked after 10⁸ and 10¹² guesses, with the significantly more secure condition in the cond 2 column. The right-hand side shows significant usability differences. These include differences in the Part One dropout rate and the number of attempts needed to create a satisfactory password. The more usable condition is in the cond 2 column.

Agreement with creation difficult					Part One recall time				
Omnibus $\chi^2=405.645, p<.001$					Omnibus KW $\chi^2=117.625, p<.001$				
cond 1	%	cond 2	%	p-value	cond 1	med	cond 2	med	p-value
<i>2s-list-patt12</i>	50.2	<i>2class16</i>	40.1	<.001	<i>2class16</i>	8.9	<i>2list-patt12</i>	8.4	U=543322.5, .039
		<i>2list12</i>	32.8	<.001			<i>2list12</i>	8.0	U=596162, <.001
		<i>2s-list12</i>	27.4	<.001			<i>3class12</i>	7.7	U=608448.5, <.001
		<i>2class12</i>	25.1	<.001			<i>2s-list12</i>	7.7	U=588180, <.001
<i>2list-patt12</i>	50.0	<i>3class12</i>	24.1	<.001	<i>2pattern12</i>	8.8	<i>2class12</i>	7.2	U=657899.5, <.001
		<i>2class16</i>	40.1	<.001			<i>2list12</i>	8.0	U=573492, .003
		<i>2list12</i>	32.8	<.001			<i>3class12</i>	7.7	U=584217.5, <.001
		<i>2s-list12</i>	27.4	<.001			<i>2s-list12</i>	7.7	U=565331, <.001
<i>2pattern12</i>	46.8	<i>2class12</i>	25.1	<.001	<i>2s-list-patt12</i>	8.7	<i>2class12</i>	7.2	U=632595.5, <.001
		<i>3class12</i>	24.1	<.001			<i>2list12</i>	8.0	U=546893.5, .033
		<i>2class16</i>	40.1	.015			<i>3class12</i>	7.7	U=557563.5, .002
		<i>2list12</i>	32.8	<.001			<i>2s-list12</i>	7.7	U=539154.5, .006
<i>2class16</i>	40.1	<i>2s-list12</i>	27.4	<.001	<i>2list-patt12</i>	8.4	<i>2class12</i>	7.2	U=601437, <.001
		<i>2class12</i>	25.1	<.001			<i>2list12</i>	8.0	U=606090.5, .001
		<i>3class12</i>	24.1	<.001			<i>3class12</i>	7.7	U=595388, .021
		<i>2s-list12</i>	27.4	<.001			<i>2s-list12</i>	7.7	U=582572.5, .01
<i>2list12</i>	32.8	<i>2class12</i>	25.1	<.001	Proportion of storage participants				
		<i>3class12</i>	24.1	<.001	Omnibus $\chi^2=57.391, p<.001$				
		<i>2s-list12</i>	27.4	.044	cond 1	%	cond 2	%	p-value
Agreement with recall being difficult					<i>2s-list-patt12</i>	67.5	<i>2class16</i>	56.7	.002
Omnibus $\chi^2=85.906, p<.001$					<i>2s-list12</i>		<i>2s-list12</i>	56.5	.002
cond 1	%	cond 2	%	p-value	<i>3class12</i>		<i>3class12</i>	52.7	<.001
<i>2list-patt12</i>	49.1	<i>2class12</i>	35.4	<.001	<i>2class12</i>		<i>2class12</i>	50.8	<.001
		<i>3class12</i>	36.0	<.001	<i>2list-patt12</i>	64.0	<i>3class12</i>	52.7	.002
		<i>2list12</i>	35.7	<.001	<i>2class12</i>		<i>2class12</i>	50.8	<.001
		<i>2class12</i>	35.4	<.001	<i>2pattern12</i>	61.7	<i>3class12</i>	52.7	.031
<i>2s-list-patt12</i>	49.0	<i>2s-list12</i>	32.6	<.001	<i>2class12</i>		<i>2class12</i>	50.8	.002
		<i>2class16</i>	38.5	.002	<i>2list12</i>	59.6	<i>2class12</i>	50.8	.032
		<i>3class12</i>	36.0	<.001					
		<i>2list12</i>	35.7	<.001					
<i>2pattern12</i>	47.4	<i>2class12</i>	35.4	<.001					
		<i>2s-list12</i>	32.6	<.001					
		<i>2class16</i>	38.5	.019					
		<i>3class12</i>	36.0	<.001					
		<i>2list12</i>	35.7	<.001					
		<i>2class12</i>	35.4	<.001					
		<i>2s-list12</i>	32.6	<.001					

Table 7.9: Significant usability differences. The more usable conditions are in the cond 2 column. The left-hand side shows agreement with password creation and recall being difficult. The right-hand side shows Part One recall timing (for participants who correctly entered their password on the first try) and password storage.

7.5 Discussion

This section discusses our findings from both studies. We succeeded in finding password-composition policies that offer advantages over the traditional *comp8* policy, without being worse in any metric we analyzed. We also found that substring blacklists can improve password strength without making recall more difficult or error-prone. The following sections summarize our findings, discuss findings between studies, and provide recommendations for service providers.

7.5.1 Results Summary

We conducted two studies on password-composition policies that require longer passwords. Table 7.10 shows security and usability metrics from both studies. For each metric, each other condition is compared with the base condition for its study – *comp8* for study I and *3class12* for study II. The darker red color indicates an unfavorable significant comparison with the base condition, and the lighter blue color indicates a favorable comparison.

Study I examined password policies with different minimum lengths, and policies that combined length and character-class requirements. Conditions *3class12* and *2word16* compared favorably to the traditional *comp8* policy. Their passwords were easier to create and to recall. Further, their passwords were less likely to be guessed after 10^{12} guesses, and no more likely after 10^8 guesses. Other conditions in the study had advantages, but compared unfavorably with *comp8* in one or more metric.

Study II examined conditions that added further length and character-class requirements to *2class12*. Adding the blacklist or pattern requirement to *2class12* made the resulting passwords more secure. The pattern requirement made password creation and recall more difficult and error-prone. The blacklist requirement made passwords more difficult to create, but did not make recall more difficult.

7.5.2 Comparing *2word16* and *2s-list12*

Passwords in both *2word16* and *2s-list12* were more secure than those in *3class12*, without being more difficult to recall. This is a compelling reason to compare *2word16* and *2s-list12*. However, we collected data for study I and study II at different times. To determine whether we could reasonably compare *2word16* and *2s-list12* directly, we compared the set of *3class12* participants who finished Part Two of either study. These two sets of *3class12* participants did not differ significantly in password strength, difficulty with password creation or recall, or attempts needed to create or recall their passwords ($p \leq .1$). The similarity of *3class12* between studies made us comfortable comparing *2word16* and *2s-list12* in this subsection.

We compared the 981 *2word16* participants who finished Part Two of study I to the 648 *2s-list12* participants who finished Part Two of study II. After 10^8 guesses, their proportions of guessed passwords did not differ significantly ($\chi_1^2=2.012, p=.156$). After 10^{12} guesses, *2word16* passwords were less likely to be guessed (6.3% to 10.5%) ($\chi_1^2=8.697, p=.003$). Participants in *2s-list12* were less likely to find password creation difficult (27.3% to 35.2%) ($\chi_1^2=10.695, p=.001$), and took fewer attempts to create their passwords, (1.8 to 2.0 mean attempts) (KW $\chi_1^2=15.666, p < 0$). There was no significant difference in finding password recall difficult ($\chi_1^2=2.892, p=.089$) or part-two recall attempts (KW $\chi_1^2=0.047, p=.829$). Neither condition stood out as clearly superior.

Condition	Part One completion (%)	Password storage (%)	Mean creation attempts	Agree creation difficult (%)	Part One recall attempts	Part One entry time (s)	Part Two recall attempts	Part Two entry time (s)	Agree recall difficult (%)	Cracked@10 ⁸ (%)	Cracked@10 ¹² (%)
Study I											
<i>comp8</i>	83.0	56.9	2.4	32.8	1.1	7.1	1.7	13.2	39.3	3.5	24.8
<i>basic12</i>	94.5	45.4	1.5	15.2	1.1	6.2	1.6	11.6	27.4	9.6	28.8
<i>basic16</i>	93.9	49.9	1.8	28.5	1.1	7.4	1.6	13.7	30.1	7.0	13.1
<i>basic20</i>	93.9	50.0	1.9	35.2	1.2	9.0	1.6	15.3	32.9	4.5	7.5
<i>2word12</i>	92.0	51.4	1.9	21.9	1.1	6.8	1.6	13.1	31.0	6.6	24.2
<i>2word16</i>	92.1	51.3	2.1	34.7	1.1	8.4	1.7	14.6	36.8	2.0	6.7
<i>3class12</i>	92.0	54.9	1.5	26.0	1.1	7.4	1.7	14.8	35.3	4.8	17.1
<i>3class16</i>	90.5	60.2	1.9	40.3	1.1	8.8	1.7	16.2	42.9	0.8	2.7
Study II											
<i>3class12</i>	92.0	52.7	1.6	24.1	1.1	7.7	1.8	15.28	36.0	3.9	16.5
<i>2class12</i>	93.3	50.8	1.6	25.1	1.1	7.2	1.7	15.09	35.4	2.8	16.4
<i>2class16</i>	90.4	56.7	1.8	40.1	1.2	8.9	1.7	18.60	38.5	1.2	3.2
<i>2list12</i>	91.9	59.6	1.8	32.8	1.1	8.0	1.7	14.97	35.7	0.4	9.5
<i>2s-list12</i>	90.5	56.5	1.9	27.4	1.1	7.7	1.8	15.64	32.6	0.8	10.6
<i>2pattern12</i>	88.7	61.7	2.4	46.8	1.1	8.8	1.7	19.00	47.4	0.7	5.7
<i>2list-patt12</i>	87.4	64.0	2.4	50.0	1.1	8.4	1.7	18.66	49.1	0.1	2.9
<i>2s-list-patt12</i>	86.0	67.5	2.6	50.2	1.1	8.7	1.7	19.38	49.0	0.3	2.6

Table 7.10: A summary of findings. Study I results are shown first, with each condition compared to *comp8*. Study II results compare each condition to *3class12*. Lighter blue indicates being significantly better than the control, and darker red indicates being worse. No shading indicates no significant difference.

7.5.3 Create a Substring Blacklist

Using a substring blacklist made passwords more secure without making recall more difficult. The ideal contents of a substring blacklist depend on context. For example, passwords in our studies often contained the substring “turk” because we conducted them on MTurk. Most websites would not benefit from having “turk” in their substring blacklist, unless they were websites about Constantinople.

Given a set of known passwords, creating an optimal set of k substrings to preclude the maximum number of passwords from the set can be reduced to the maximum coverage problem, which is known to be NP-hard. Let each password be an element. Each substring can be considered to be a subset of those passwords. Any password containing that substring is considered to be in the subset of elements associated with that substring. The objective then becomes to create a blacklist of k substrings (or subsets) such that the maximum number of passwords (elements) is covered.

Our technique for creating a substring blacklist from a password corpus might be useful to service providers. We first created a list of all password substrings with lengths between four and the length of the shortest password in the corpus. Then, we removed from the list any substring in fewer than 1% of passwords. This can be adjusted to increase or decrease the final blacklist

size. We then made a second pass through the list and removed any substrings that did not occur in at least 1% of passwords without being part of a larger substring in the list. For example, if “password” and “sword” are both on the list, we would remove “sword” unless it appears in 1% of passwords outside of “password.”

We applied this blacklist-creation algorithm to the 289,039 passwords with at least eight characters in the Yahoo password set [32]. The only two substrings common to at least 1% of these passwords were “1234” and “love.” An algorithmic approach to creating a substring blacklist can lead to over-fitting, and we recommend manually improving the blacklist. This can include, for example, adding the name of the website and other terms related to the service.

7.5.4 Recommendations for Service Providers

Although password policies similar to *comp8* are traditionally considered “strong,” *3class12* and *2word16* were both more secure and more usable. Therefore, our findings suggest that service providers currently using a policy similar to *comp8* can increase both security and usability for their users. We found that *2class12* and *3class12* had very similar usability and security results. In fact, 90% of *2class12* participants created passwords that met the *3class12* requirements.

The blacklist requirement made passwords less likely to be guessed. It made passwords more difficult to create, but not more difficult to recall. Therefore, the blacklist requirement is promising for settings in which password creation is generally not frequent, and that require strong passwords. It may also be promising for institutions, such as universities, that are unlikely to turn away prospective users by having more onerous password policies.

We also found that a number of the requirements we studied had downsides that limit their usefulness to service providers. Policies with length as their only requirement were generally usable, but many participants in with those policies made easily guessed passwords. While *3class16* led to fairly strong passwords, it was significantly less usable than *comp8*. The pattern made passwords more difficult to guess, but also made passwords more difficult to create and to recall.

Chapter 8

Can Creation-Time Feedback Help Users Create Passwords?

8.1 Introduction

Chapters 6 and 7 have explored password-composition policies for user-created passwords. Compared to a simple policy that requires only eight characters, we found that stricter policies lead to stronger passwords. On the other hand, users often struggled to create a password meeting complex password requirements. In this paper, we explore *requirements feedback*: real-time password-creation feedback telling users whether they have met password-composition requirements. We intend this to help users create passwords that meet strict password-composition requirements. Some service providers already offer requirements feedback: For example, Yahoo displays green check marks indicating satisfied requirements (Figure 8.1). Another mechanism this chapter explores for helping users create passwords that meet strict requirements is *guidance*: guiding users through a multi-step password-creation process. An example of guidance used in practice, from Dashlane, is in Figure 8.2.

Previous work on user-facing feedback has focused specifically on password meters that provide estimates of password strength [23, 94]. In this chapter, we present the first scientific analysis of the usability of requirements feedback and guidance mechanisms intended to help users successfully navigate difficult requirements. Using a 6,435-participant, between-subjects online study, we considered real-time requirements feedback for three strict composition policies. For one of these policies, we also examined two approaches to multi-step password-creation. In *guidance*, participants were guided to enhance a simple password by adding components until it met all requirements. In *insertion*, inspired by Forget et al. [29], participants created a simple password and the system inserted random characters to meet all requirements. We tested how these approaches affect users' ability to create conforming passwords quickly and correctly, users' perceptions of these approaches, and how the approaches impact the security and memorability of the resulting passwords.

We found that requirements feedback helped participants create passwords meeting strict requirements without making errors, and in some cases gave participants more confidence in the

This chapter is a partial reproduction of [82], co-authored with Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, Alain Forget, Saranga Komanduri, Michelle Mazurek, William Melicher, Sean M. Segreti, and Blase Ur.



Figure 8.1: An example of real-time password-creation feedback in the wild, from Yahoo.

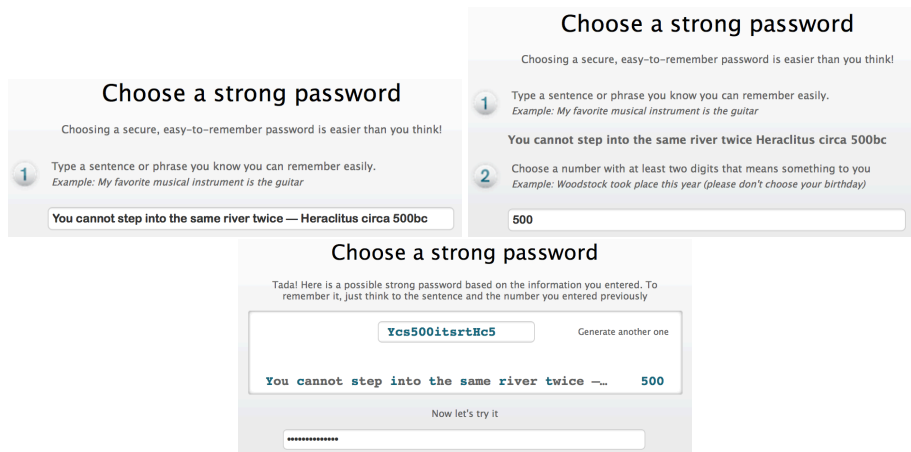


Figure 8.2: An example of a multi-step password-creation process, from Dashlane.

strength of their passwords. Passwords created with requirements feedback were as strong as those without. Thus, requirements feedback may help make complex password-composition requirements, including those studied in prior chapters, more palatable to users.

Compared to having participants create a password under a strict policy in a single step, both the guidance and insertion techniques led to a reduction in password security. While user sentiment toward the process of creating passwords with guidance was more positive than creating a password without guidance, the resulting passwords were less likely to exceed the stated requirements and more likely to be cracked after a large number of guesses. Likewise, randomly inserting characters into passwords leaves them vulnerable to a brute-force attack across the inserted characters. Overall, our results demonstrate that looking only at password requirements is insufficient, as the presentation of those requirements, including requirements feedback and guidance, can affect both usability and security. This finding suggests that the password-composition policies studied in previous chapters can benefit from requirements feedback.

We begin by discussing our research questions in Section 8.2. We discuss our conditions in Section 8.3. We then present our usability and security results in Section 8.4. Finally, we discuss the implications of our findings for system administrators managing password-creation mechanisms in Section 8.5.

8.2 Research Questions

In this section, we present the research questions that our study addresses. The unifying theme behind these questions is examining how strict password requirements can be made more usable through visual elements, requirements feedback, guidance, and automatic insertion of random characters. All of the research questions in this study were applied to passwords with requirements that include at least 12 characters and 3 character classes – we call these the *base* requirements.

Question Q_1 : *How do the blacklist and pattern requirements affect password security and usability when applied to requirements that already demand longer passwords with multiple character classes?* The *blacklist* requirement prohibits passwords from containing any substring in a 41,329-string blacklist. *Pattern* requires that passwords begin and end with a lowercase letter. These requirements are grounded in prior work that has found users often include common substrings in passwords, and often begin and end them with required non-lowercase character classes (Chapter 7). Based on previous findings in Chapters 6 and 7, we expected adding either requirement to the base requirement would increase strength and decrease usability.

Question Q_2 : *Does an organization-branded look-and-feel help users better create or recall passwords with length and character class requirements?* We collected data in two conditions that displayed the same information and had the same requirements, but differed in their branding and text color. Our base condition took its look-and-feel from a university, including its wordmark and color scheme, while the base-plain condition used the default HTML rendering. We wondered if the distinct visuals of the former may help participants recall their passwords, perhaps by acting as a memory cue [17]. This has real-world implications because users often create and subsequently recall passwords on websites with distinct branding, and select predictable passwords based on such visual cues [17].

Question Q_3 : *How does requirements feedback affect password creation, recall, and strength when applied to strong password requirements?* We compared sets of conditions that differed only in whether participants received real-time requirements-compliance feedback during password creation. We examined the effects of this requirements feedback on password security and usability. We compared passwords with only the base requirement to those also using the blacklist or pattern requirement.

Question Q_4 : *Does a three-step password-creation process, using either guidance or insertion, make it easier to create passwords that have strict requirements?* We compared three conditions with the base and pattern requirements that included requirements feedback. One condition asked participants to create passwords in a single step, while two conditions used three steps. In one of the three-step conditions, we guided participants through a password-creation process (guidance) such that they created a simple password and then were asked to add more characters. In the other three-step condition, participants created a simple password and then we randomly inserted two more characters (insertion). The insertion mechanism is inspired by [29] and is explained in more detail in Section 8.3.

8.3 Methodology

Next we discuss our data-collection and analysis procedures. Our protocol is described in Chapter 4, using the SHELF framework discussed in Chapter 3.

Condition	pattern	blacklist	feedback	brand	3-step
<i>Base</i>				University	
<i>Base_{rt}</i>			✓	University	
<i>Blacklist</i>		✓		University	
<i>Blacklist_{rt}</i>		✓	✓	University	
<i>Base-plain</i>				Plain	
<i>Pattern</i>	✓			University	
<i>Pattern_{rt}</i>	✓		✓	University	
<i>Guide</i>	✓		✓	University	Participant adds
<i>Insert</i>	✓		✓	University	System adds

Table 8.1: Requirements of each condition. All require at least 12 characters and three character classes. The blacklist requirement prohibits certain substrings in passwords. Pattern requires starting and ending with lowercase letters. 3-step uses a multiple-step process to create the password.

8.3.1 Measuring Password Strength

Our cracking procedures are described in Section 4.2. We included in our training data words from the Google corpus [10] and data from prior studies. We used the PCFG algorithm to make guesses with at least twelve characters and three character classes, as required by all of our conditions. Some of our conditions required passwords to begin and end with a lowercase letter; these were trained as a separate group and only guesses starting and ending with a lowercase letter were made. We used two folds for each group of passwords being cracked and generated guesses to at least 2×10^{13} per condition.

Our *Insert* condition, detailed in Section 8.3.2, asked participants to create a password of at least ten characters that started and ended with a lowercase letter. The system then added random characters of 43,200 different possible combinations. At first, the traditional PCFG approach was ineffective against *Insert*, cracking only one password after 10^{12} guesses. However, we then took a different cracking approach. We used the PCFG algorithm to crack the pre-enhancement passwords, and then multiplied the number of guesses used to guess the pre-enhancement passwords by 43,200. This simulated an attacker who knows the random-character insertion algorithm and brute-forces through all possible permutations. Our security results are based on this more effective technique in order to simulate a more powerful and aware attacker. As with the other conditions, we performed two-fold cracking on this condition.

8.3.2 Conditions

Conditions, listed in Table 8.1, were assigned round-robin. All conditions share the *base* requirements – that passwords have at least 12 characters and three character classes. Chapter 7 suggests these are usable yet strong requirements.

8.3.3 Password Creation in One Step

Base used only the base requirements. *Pattern* adds the *pattern* requirement that passwords begin and end with lowercase letters, as shown in Figure 8.3. *Blacklist* also contains the *blacklist* requirement that passwords not contain substrings from a 41,329-string blacklist, as follows:

Figure 8.3: Password-requirements presentations for *Pattern* and *Pattern_{rt}* as “password!” is entered.

- 123!, amazon, character, monkey
- number, survey, this, turk
- Any year between 1950 and 2049
- The same character four or more times in a row
- Any four consecutive characters from password
- Any four sequential digits
- Any four sequential letters in the alphabet
- Any four consecutive characters on the keyboard

Base_{rt}, *Blacklist_{rt}*, and *Pattern_{rt}* are analogues of the above three conditions, except they employ requirements feedback. As participants entered their passwords, listed requirements were accompanied with either a green check mark (for a fulfilled requirement) or a red message indicating why the requirement is not fulfilled. A comparison between *Pattern* and *Pattern_{rt}* is in Figure 8.3.

This feedback is consistent with the recommendations of Moshfeghian and Ryu, who examined popular websites and found many of them presented poor guidance. The recommended using instructions with clear language near password fields, providing real-time feedback, and using red-colored feedback to indicate errors [63].

Base and our other conditions are displayed with Carnegie Mellon University’s branding (colors, fonts, and wordmark). *Base-plain* is identical to *Base*, except it employs no branding, instead using default HTML rendering, to address Q_2 .

8.3.4 Password Creation in Three Steps

While participants in the aforementioned seven conditions created a password in one step, two conditions – *Guide* and *Insert* – employed a three-step process. Both use requirements feedback and led to passwords meeting the same requirements as *Pattern_{rt}*. The first two steps of *Guide* are shown in Figure 8.4, and those of *Insert* are in Figure 8.5.

In *Step 1*, participants were asked, “To start, please enter a password with at least 10 characters. It can be a word, and it needs to start and end with a lowercase letter.” In *Step 2*, passwords were enhanced to meet the requirements of *Pattern*. In *Guide*, participants were shown their password in an editable text field and asked to enhance it until meeting the requirements. In *Insert*, participants

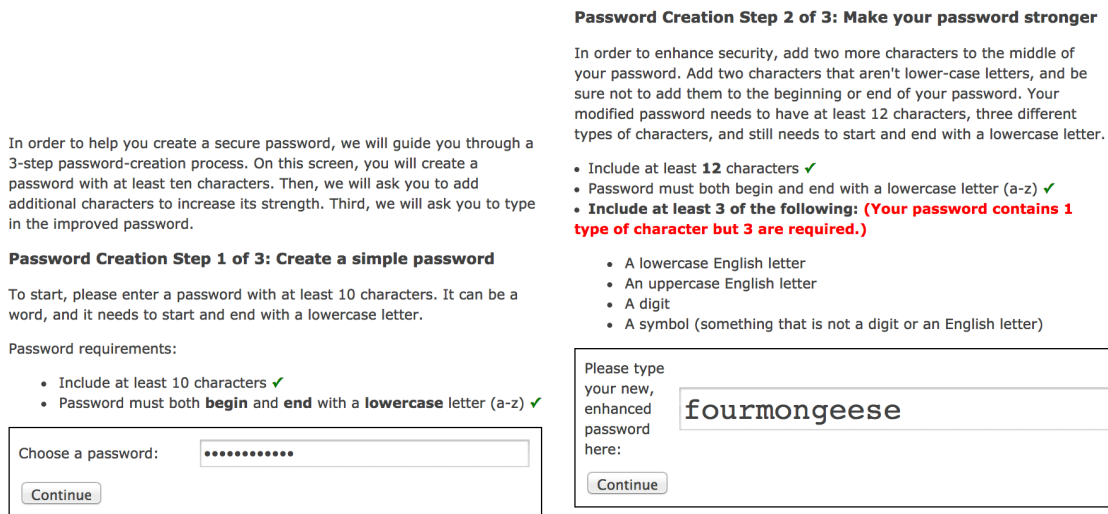


Figure 8.4: The first two steps of password creation in *Guide*.

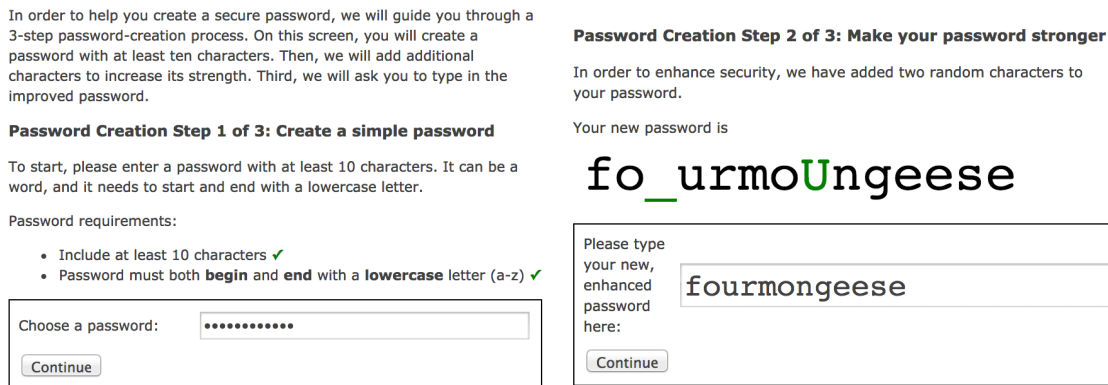


Figure 8.5: The first two steps of password creation in *Insert*.

were shown their system-enhanced password with the two new characters in green. Participants saw a text box with their non-enhanced password and were asked to modify it to match what was above. In *Step 3*, we asked participants to confirm their passwords in a blank password field.

To insert two random characters in *Insert*, we first selected two characters of two different character classes, chosen from 9 symbols @!\$ * #. - &_ , 8 possible digits, and 24 uppercase letters (removing *O*, *0*, *I*, and *l*, since they may be confused with each other). We then inserted these characters between the first ten characters of the initial password. There are 960 different ways to choose characters of two different character classes from these three sets (with order mattering). There are 45 different ways to select two spaces between the first ten characters. Thus, there are $960 \times 45 = 43,200$ different ways of inserting characters.

condition pairs	Creation errors		Creation time		Creation difficult		Creation annoying	
	mean	p-val	med (sec)	p-val	% agree	p-val	% agree	p-val
Q1: Impact of blacklist and pattern requirements								
<i>Base – Blacklist</i>	0.7 – 0.8	.152	61 – 88	<.001	24 – 32	.002	55 – 61	.212
<i>Base – Pattern</i>	0.7 – 1.5	<.001	61 – 122	<.001	24 – 50	<.001	55 – 77	<.001
<i>Blacklist – Pattern</i>	0.8 – 1.5	<.001	88 – 122	<.001	32 – 50	<.001	61 – 77	<.001
Q2: Impact of branded look-and-feel								
<i>Base – Base-plain</i>	0.7 – 0.6	.039	61 – 60	1	24 – 23	1	55 – 57	1
Q3: Impact of password-creation feedback								
<i>Base – Base_{rt}</i>	0.7 – 0.3	<.001	61 – 60	1	24 – 22	1	55 – 55	1
<i>Blacklist – Blacklist_{rt}</i>	0.8 – 0.4	<.001	88 – 86	.621	32 – 29	.618	61 – 56	.281
<i>Pattern – Pattern_{rt}</i>	1.5 – 0.6	<.001	122 – 109	.019	50 – 49	1	77 – 77	1
Q4: Impact of guiding and insertion through creation								
<i>Guide – Pattern_{rt}</i>	0.8 – 0.6	.114	116 – 109	.036	29 – 49	<.001	67 – 77	<.001
<i>Insert – Pattern_{rt}</i>	0.7 – 0.6	.023	101 – 109	.019	17 – 49	<.001	45 – 77	<.001

Table 8.2: Password-creation metrics and comparisons.

8.3.5 Statistical Testing

The statistical tests used are described in Section 4.4. Rather than testing each possible pair of conditions, we test selected pairs of conditions to help answer our research questions. Except for Q_1 , all comparisons are between conditions that have the same password requirements. The pairwise comparisons that we perform are as follows.

Q_1 : *Base–Blacklist*; *Base–Pattern*; *Blacklist–Pattern*

Q_2 : *Base–Base-plain*

Q_3 : *Base–Base_{rt}*; *Blacklist–Blacklist_{rt}*; *Pattern–Pattern_{rt}*

Q_4 : *Pattern_{rt}–Guide*; *Pattern_{rt}–Insert*

8.4 Results

In this section, we present the findings of our study. We first discuss our participants in Section 8.4.1. We present password strength in Section 8.4.2, and user perception of password strength in Section 8.4.3. We present usability results for password creation and recall in Section 8.4.4. Result metrics and comparisons for password creation are summarized in Table 8.2. Metrics for password characteristics are in Table 8.3. Other metrics are summarized in Table 8.4.

8.4.1 Participants

We collected data in May and June 2014. A total of 7,262 participants began our study and 6,435 finished Part One. All who finished Part One were invited to return for Part Two. 3,934

condition pairs	Exceed min class		Password length		Cracked after 2×10^{13} guesses		Storage	
	%	p-val	mean char	p-val	%	p-val	%	p-val
Q1: Impact of blacklist and pattern requirements								
<i>Base – Blacklist</i>	71 – 67	.176	13.6 – 13.8	.285	28 – 21	.012	49 – 57	.091
<i>Base – Pattern</i>	71 – 57	<.001	13.6 – 14.1	<.001	28 – 8	<.001	49 – 69	<.001
<i>Blacklist – Pattern</i>	67 – 57	<.001	13.8 – 14.1	.003	21 – 8	<.001	57 – 69	.003
Q2: Impact of branded look-and-feel								
<i>Base – Base-plain</i>	71 – 73	.449	13.6 – 13.5	1	28 – 30	.835	49 – 52	1
Q3: Impact of password-creation feedback								
<i>Base – Base_{rt}</i>	71 – 53	<.001	13.6 – 13.6	1	28 – 30	.835	49 – 50	1
<i>Blacklist – Blacklist_{rt}</i>	67 – 51	<.001	13.8 – 13.6	.285	21 – 24	.525	57 – 55	1
<i>Pattern – Pattern_{rt}</i>	57 – 39	<.001	14.1 – 14.2	1	8 – 10	.525	69 – 61	.154
Q4: Impact of guiding and insertion through creation								
<i>Guide – Pattern_{rt}</i>	22 – 39	<.001	13.8 – 14.2	.008	16 – 10	.02	62 – 61	1
<i>Insert – Pattern_{rt}</i>	15 – 39	<.001	13.7 – 14.2	<.001	29 – 10	<.001	76 – 61	<.001

Table 8.3: Password characteristics and comparisons.

total participants finished Part Two within three days of being invited. Among our 6,435 Part One participants, the median age was 28 years. 99.2% of our participants disclosed their gender; among these, 46.7% were male and 52.5% female. 95.7% of respondents stated their highest degree, of which 45.9% held a Bachelor’s degree or higher.

Participants in *Pattern_{rt}* (84.7%) who began Part One were less likely to finish it than those in *Guide* (90.8%) or *Insert* (93.8%) (HC FET, $p=.001$). The higher dropout rates may suggest greater user difficulty and frustration. 64.1% of participants who completed Part One returned for Part Two within five days and this did not vary by condition ($\chi^2_8=2.77$, $p=0.948$). 95.3% who started Part Two within five days finished and this too did not vary by condition ($\chi^2_8=4.64$, $p=0.795$).

8.4.2 Password Strength

We observe an overall cracking rate of 21.7% after 2×10^{13} guesses, varying significantly by condition, as shown in Table 8.3. The percentage of passwords cracked after each guess are shown in Figure 8.6. As observed in Table 8.3, we found no effect of real-time requirements-compliance feedback on password strength (see Q_3). On the other hand, both *Pattern* and *Blacklist* performed better than *Base*, and *Pattern* performed better than *Blacklist* (see Q_1). Passwords created under *Pattern_{rt}* were significantly less likely to be cracked than those created under *Guide* or *Insert*. Thus our three-step password creation processes both decreased password strength compared to creating a password with the same requirements in a single step (see Q_4).

While *Insert* was vulnerable to the partial-brute-force approach described in the Methodology, it did perform well against a more traditional PCFG attack. Using a traditional PCFG approach, after 10^{12} guesses, over 5% of each other condition had been cracked, compared to 0.1% of *Insert*.

condition pairs	Finished Part One		Part One recall time		Part two recall attempts		Part two recall difficult	
	%	p-val	med (sec)	p-val	mean	p-val	% agree	p-val
Q1: Impact of blacklist and pattern requirements								
<i>Base – Blacklist</i>	88 – 88	1	10 – 10	1	1.9 – 1.9	1	36 – 43	.368
<i>Base – Pattern</i>	88 – 84	.131	10 – 11	<.001	1.9 – 1.7	.244	36 – 54	<.001
<i>Blacklist – Pattern</i>	88 – 84	.131	10 – 11	.004	1.9 – 1.7	.83	43 – 54	.004
Q2: Impact of branded look-and-feel								
<i>Base – Base-plain</i>	88 – 91	.384	10 – 9	1	1.9 – 1.9	1	36 – 32	.944
Q3: Impact of password-creation feedback								
<i>Base – Base_{rt}</i>	88 – 89	1	10 – 10	1	1.9 – 1.9	1	36 – 35	.983
<i>Blacklist – Blacklist_{rt}</i>	88 – 89	1	10 – 10	1	1.9 – 2.0	1	43 – 40	.983
<i>Pattern – Pattern_{rt}</i>	84 – 85	1	11 – 11	1	1.7 – 1.7	1	54 – 46	.094
Q4: Impact of guiding and insertion through creation								
<i>Guide – Pattern_{rt}</i>	91 – 85	.001	10 – 11	.181	2.0 – 1.7	.088	42 – 46	.944
<i>Insert – Pattern_{rt}</i>	94 – 85	<.001	12 – 11	1	2.0 – 1.7	.146	53 – 46	.26

Table 8.4: User-behavior metrics and comparisons.

However, *Insert* appeared much weaker once we instead applied PCFG to its pre-splice passwords and multiplied the resulting guess numbers by the total number of splicings, 43,200, to simulate brute-forcing the possible splicings. We see that using a relatively small amount of random text to increase password strength is vulnerable to being brute forced.

Our study used 9 different symbols, chosen to be distinct and easy to recognize. While increasing the number of symbols would have increased security, it would not have been a substantial improvement. Using 19 symbols rather than 9 would have resulted in 72,000 different insertion configurations instead of 43,200. Using 32 symbols would have led to 109,440 different configurations. Using the same cracking technique of using PCFG to guess pre-enhancement passwords and then brute-forcing the inserted characters, using 32 symbols would lead to a cracking rate of 26.1%, a minor improvement compared to the current 28.6%.

Exceeding Minimum Requirements

Beyond cracking results, we looked at the structural components of passwords. While all passwords required 12 characters and 3 characters classes, participants were free to exceed either. The percentage of participants who exceeded the minimum three-character-class requirement is shown in Table 8.2. *Pattern_{rt}* participants were more likely to exceed the minimum than those in *Guide* or *Insert*. In addition, we found that participants in all three real-time-feedback conditions were less likely to use four character classes than those in the same conditions without real-time feedback. This may be due to participants with real-time feedback feeling that they are “done” once they see the green check mark beside the requirement (see Q_3).

Although all conditions required a minimum of 12 characters, 59.1% of participants created longer passwords, with a mean password length of 13.8 characters. Length by condition and sig-

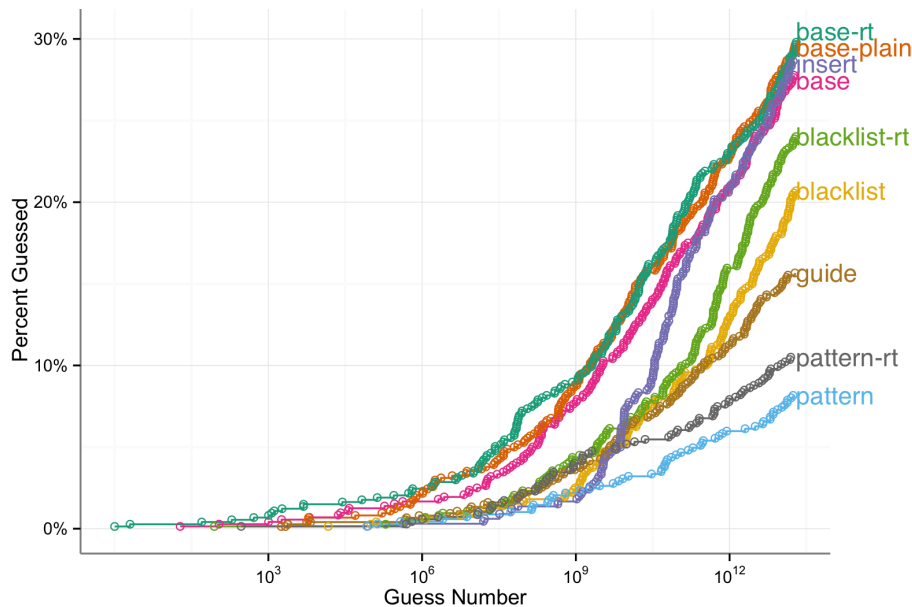


Figure 8.6: The percentage of passwords cracked in each condition by the number of guesses made in log scale. Our cutoff for guess numbers is 2×10^{13} . Significant comparisons are in Table 8.4.

nificant differences are shown in Table 8.2. While both *Guide* and *Insert* had shorter passwords than *Pattern_{rt}* (see Q_4), we did not see significant length differences between conditions with and without real-time feedback. Previous work found that increasing password length by one lowercase character makes a password 70% as likely to be guessed [58]. Thus, the length difference between *Pattern_{rt}* and the 3-step conditions may contribute substantially to the security differences between these conditions.

8.4.3 User Perception of Password Strength

In order to measure perception of how feedback affected password security, we asked our participants in a Likert question whether they agreed with the statement, “The feedback and instructions I saw while creating my password led me to create a stronger password than I would have otherwise”. Responses are shown in Figure 8.7. The only significant pairwise differences were that *Pattern_{rt}* participants (84.3%) were more likely to agree than those in either *Pattern* (74.2%) or *Insert* (67.4%) (HC FET, $p < .001$). One possible explanation for the low agreement from *Insert* participants is that they anticipated creating a weaker initial password, because they expected to have its strength increased. Alternatively, they may not have perceived much security in adding two random characters (see Q_4).

For further insight into participant perception of password strength, we asked whether they agreed with, “If my main email provider had the same password requirements as used in this study, my email account would be more secure”. Responses are shown in Figure 8.7. The sole pairwise significant difference was that participants in *Pattern_{rt}* (64.9%) were more likely to agree than those in *Insert* (51.2%) (HC FET, $p < .001$). No other comparison was significant. 79.6% of

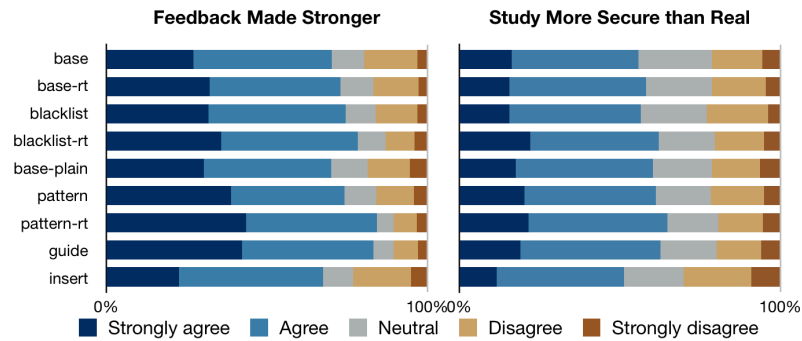


Figure 8.7: Participant agreement with “The feedback and instructions I saw while creating my password led me to create a stronger password than I would have otherwise” and “If my main email provider had the same password requirements as used in this study, my email account would be more secure”.

our participants indicated that their primary email account was through a web email provider. The relatively low agreement for this question is surprising given that the study requirements are much more demanding than most web email providers.

8.4.4 Usability

This section presents results for usability metrics and self-reported sentiment. Section 8.4.4 focuses on the password-creation process. Section 8.4.4 looks at Part One recall, and Section 8.4.4 looks at Part Two recall.

Password Creation

To understand password creation, we begin by looking at self-reported user sentiment. Then, we look at password creation time and whether and how participants fail to create a password meeting their requirements. Finally, we look at the effects of web-page branding.

Figure 8.8 illustrates participants’ agreement with password creation being difficult and annoying respectively, with pairwise differences in agreement shown in Table 8.2. We did not detect any significant difference in perceived password-creation difficulty or annoyance between a condition and its real-time-feedback counterpart. Thus, our real-time feedback did not cause participants to perceive password creation as any more or less difficult or annoying (see Q_3). Creating a password under $Pattern_{rt}$ was both more difficult and more annoying than doing so under either $Guide$ or $Insert$. This suggests that our 3-step conditions reduced the difficulty and annoyance of the $pattern$ requirement (see Q_4).

Table 8.2 shows significant differences in how long participants took creating their passwords. $Pattern_{rt}$ took significantly less time than $Guide$ but significantly more time than $Insert$. This may be due to the fact that $Insert$ participants did not have to decide which special characters to include (see Q_4). Furthermore, requirements feedback helped participants create passwords in $Pattern$ more quickly (see Q_3).

Table 8.2 compares how many errors participants made while creating a password in each condition. We consider creation errors a metric of password-creation difficulty. We observe that requirements feedback helps participants successfully create a password with fewer errors for all

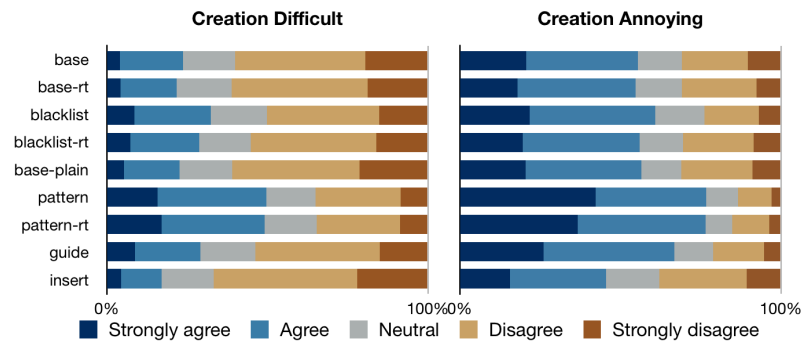


Figure 8.8: Participant agreement with “Creating a password that meets the requirements given in this study was difficult/annoying”.

three pairs of conditions with requirements feedback. This suggests that real-time requirements-compliance feedback helps participants to adhere to strict password requirements (see Q_3).

To understand better the impact of guiding and insertion (see Q_4), we examined the types of errors users made when creating passwords in each of our four conditions that had pattern requirements. The *Pattern* condition had the highest error rate of any condition, with an average of 1.5 creation errors per participant (across all of their creation attempts). The most common error by far was failure to meet the pattern requirement, with an average of 1.1 pattern errors per participant. With the addition of realtime feedback, participants in the *Pattern_{rt}* condition made significantly fewer errors, with only .6 errors and .3 pattern errors per participant. While participants in the *Guide* condition had a significantly lower dropout rate and reported finding password creation easier and less annoying than *Pattern_{rt}* participants, they did not make significantly fewer errors or create their passwords more quickly. Likewise, we saw similar sentiment improvements for participants in the *Insert* condition as well as significant improvement in password creation time, without a reduction in the overall error rate. In the insert condition there were an average of .7 errors overall and .07 pattern errors. This analysis suggests that real-time feedback reduces the high error rate associated with the pattern requirement, but our 3-step approaches did not result in further error-rate reductions.

We did not observe any significant effects of Carnegie Mellon University branding (see Q_2). To understand what impact branding may have had, we looked for the presence of context-related keywords in both *Base* and *Base-plain*. *Base* was more likely to contain keywords related to the university (1.3% to 0.7%), but the difference was not significant ($\chi^2_1=0.73$, $p=0.393$). On the other hand, *Base-plain* passwords were more likely to contain generic study-related keywords,¹ (2.3% to 1.0%) but this was also not significant ($\chi^2_1=3.198$, $p=0.074$).

Part One Recall

Participants were asked to recall their passwords in Part One after completing a brief survey. 92.9% of participants did so correctly on the first try, taking 1.1 attempts on average, with significant pairwise differences between conditions. The median time for Part One recall was 11 seconds, and significant differences are shown in Table 8.4. Passwords in *Pattern* took signifi-

¹“turk,” “amazon,” “mechanical,” “survey,” “study,” or “research”

cantly more time to enter than those in either *Base* or *Blacklist*. This could be an indication of participants struggling more to remember passwords created under these conditions.

Part Two Recall

In this section, we look at data collected from the 3,934 participants who returned for Part Two within three days of being invited. 59.9% of participants successfully entered their password on the first attempt, and this did not vary significantly by condition ($\chi^2_8=13.844$, $p=.086$), as shown in Table 8.4.

Participants who mentioned storing their passwords when asked in the surveys, or whom we detected pasting or autofilling when attempting to recall their passwords are considered *storage* participants. Others are considered *no-storage* participants. 58.5% of returned participants are storage participants, with significant differences shown in Table 8.4. Participants in *Insert* were more likely to store their passwords than those in *Pattern_{rt}*. Likewise, those in *Pattern* were more likely to do so than participants in either *Base* or *Blacklist*. It appears that the pattern requirement, and the insertion of random characters, cause participants to be less able to memorize their passwords, or at least anticipate being less able to do so and therefore to write them down. 57.1% of no-storage participants entered their passwords successfully in one attempt, and this did not vary by condition ($\chi^2_8=12.189$, $p=0.143$).

In addition to observing participants' behavior during Part Two recall, we also asked whether they agreed with the statement, "Remembering the password I used for this study was difficult". Significant differences are shown in Table 8.4. *Pattern* was significantly more difficult to recall than either *Base* or *Blacklist*.

8.5 Discussion

We now address each of our research questions in light of our study's findings. We evaluated three approaches to help users cope with strict password-composition policies: requirements feedback, guidance, and insertion. We found that requirements feedback helps prevent user errors while creating strong passwords. On the other hand, our multi-step password-creation processes, guidance and insertion, both made password-creation easier, but resulted in weaker passwords. While prior passwords research has often focused on which sets of requirements lead to strong passwords, most past research has not looked at the impact of presentation and instructions (beyond password meters), or at ways to help users cope with strict requirements. We believe these findings will be valuable to service providers who wish to make their increasingly strict password-composition requirements easier for users to swallow.

8.5.1 Q1: Impact of blacklist and pattern requirements

As expected, the blacklist and pattern requirements increase password strength but decrease usability. Furthermore, the pattern requirement leads to more security but less usability than the blacklist requirement. As shown in Table 8.3, *Base* passwords were significantly more likely to be cracked after 2×10^{13} guesses than *Blacklist*, which in turn were significantly more likely to be cracked than *Pattern*.

Among these three conditions, *Pattern* proved least usable: participants had more difficulty both creating and recalling their passwords. We found that *Pattern* participants were error-prone in password creation, and they reported finding password creation both difficult and annoying at significantly higher rates (Table 8.2). Participants in this condition also spent more time recalling their passwords in Part One and were more likely to store their passwords (Table 8.4). While error rates during Part Two recall did not differ significantly, *Pattern* participants were more likely to report difficulty in Part Two recall (Table 8.4). The usability differences between *Blacklist* and *Base* were less pronounced, and are generally limited to password creation. Adding a blacklist requirement may be a reasonable way to increase security with only modest usability losses during password creation.

8.5.2 Q2: Impact of branded look-and-feel

We originally thought that branding would help users to remember their password (among the many other passwords people must routinely manage), although security could suffer if users created passwords that included words related to a brand. However, we found no evidence for a difference between branded and plain presentation in any of our analyses. While we found no difference with the branding we used, a different visual presentation might have had more impact. Our branded design contained no images other than a Carnegie Mellon University wordmark. Branding that included images, or a brand that was more familiar to participants may have had a different effect. Further investigation into the effects of stronger branding on password behavior could prove interesting.

8.5.3 Q3: Impact of password-creation feedback

Our findings show some upside and limited downside to giving participants requirements feedback. We found that requirements feedback made password creation less error-prone for all three pairs of conditions we compared. As shown in Table 8.2, requirements feedback participants were less likely to use a fourth character class. This may be due to the feedback giving users the feeling of being “done.” In conditions without requirements feedback, participants may not have realized when the requirements were met and so added additional character classes to be sure. While this trend could potentially have some adverse effect on security, we found no significant difference in guessability after 2×10^{13} guesses.

In addition, adding requirements feedback appears to sometimes increase perception of password strength. This is valuable, because prior research has found that users who are more invested in password security may make better passwords [58]. Overall, adding requirements feedback seems to provide a reduction in user error, an increase in perception of strength, and little to no impact on password security. Thus, requirements feedback seems to be a useful feature to add to password creation interfaces.

8.5.4 Q4: Impact of guiding and insertion

We tested the impact of our 3-step conditions by comparing *Pattern_{rt}* with both *Guide* and *Insert*. All three conditions had the same requirements, but the latter two used an interactive three-step process to create the password over several steps. Participants in *Guide* and *Insert* both found

password creation less annoying and less difficult than *Pattern_{rt}*, as seen in Table 8.2. They were also more likely to complete Part One of the study.

However, despite all three conditions enforcing the same requirements, passwords in *Guide* and *Insert* were significantly more likely to be cracked than those in *Pattern_{rt}* — over twice as likely in the case of *Insert* (Table 8.3). This is an interesting finding, because it demonstrates that looking only at password-composition requirements, as a number of prior passwords-research papers have done, is insufficient to paint an accurate picture of resulting security.

We observed that participants in the 3-step conditions made shorter passwords with fewer character classes (Table 8.2), resulting in passwords that were more easily cracked. One possible explanation for these weaker passwords is that participants did not feel a sense of ownership over their passwords. Prior research has suggested that passwords are a way that users feel personally responsibility for computer security [85]. Perhaps, this sense of ownership and responsibility was diminished because the system appeared to be more of an active participant in the password-creation process. Another explanation is that users may have trusted that the system was helping them create a strong password, and may have focused on following the system’s instructions rather than on trying to increase the security of their passwords.

Further research might explore whether other ways of guiding participants through password-creation can retain usability gains without sacrificing security. For example, a variation on the *Guide* condition might ask users to create a simple password with at least 11 characters, rather than 10, to account for the fact that users following a traditional 1-step password-creation process are more likely to exceed minimum length requirements. Alternatively, the guidance might encourage users to exceed minimum requirements by telling users that this is a good way to increase the security of their password. The *Guide* condition that we tested specifically told users to add “two more characters to the middle of your password” and did not suggest that they could add more than two characters or that doing so would increase security even more. It is clear that the details associated with password creation instructions matter and that instructions and procedures should be tested to determine their impact on both usability and security.

Chapter 9

A Reflection on our Collected Data

In the course of the studies presented in this thesis, we have collected data from 38,402 participants. Within each study, we use its data to contrast conditions. In this chapter, I examine data from all four studies. This provides a better understanding of our usability metrics, and how those metrics relate to one another. This also contributes a better understanding of the people taking MTurk studies. Section 9.1 examines how observed metrics and survey responses correlate with self-expressed user sentiment. Section 9.2 looks at participant demographics from across our four studies. Section 9.3 reports on mobile device usage in our studies. And Section 9.4 looks at data about real email accounts of our participants.

Several survey questions changed between studies. Therefore, several of our analyses used only data from a subset of studies. I explicitly point this out in each case. In some analyses, I was missing data for a few participants, and analyzed the rest.

9.1 Factors Correlating with User Sentiment

In this analysis, I used logit regression [79] to find correlation between self-reported user-sentiment data, and other study factors. My objective was to learn more about how observed and self-reported factors affected subjective participant experience. My analysis controlled for study and condition. I included the 22,548 participants who finished Part Two of any study to have more complete data for each participant in the analysis. Table 9.1 shows the regression factors.

I performed five separate logit regressions. Each regression used the same 33 independent factors and had a different binary dependent factor. The five binary dependent factors were agreement with statements about study experience. These included three statements on password-creation (or assignment) usability (e.g., “Creating a password that meets the requirements given in this study was {annoying/difficult/fun}”); one on password-recall difficulty (e.g., “Remembering the password I used for this study was difficult”); and one on perceived password security (e.g., “If my main email provider had the same password requirements as used in this study, my email account would be more secure”). The exact wording of these statements varied between studies.

The 33 independent factors in each regression are in Table 9.1. These factors are based on either observed behavior (e.g., password-recall attempts) or responses to survey questions. Because I included participants from each study, I only included factors based on observations or questions that were in all four studies. Some survey questions changed wording between studies. In those

cases, I homogenized the questions and responses. The independent regression factors were taken from 11 observations and survey questions. For multiple-choice and choose-all-that-apply survey questions, each response option chosen by one percent of participants became a binary factor. I removed 25 participants who were missing responses to a user-sentiment question. I removed another 465 participants who were either missing a response to a question, or whose response was not given by at least one percent of participants.

9.1.1 Analysis Results

Table 9.1 shows the regression factors and results. Table 9.1 also shows the average values for independent factors. These include the mean value for numeric factors; the percentage true for binary factors; and percent agreement for Likert factors. All but two of the 23 independent factors were significantly correlated with at least one dependent factor.

For each independent factor, the p-value for correlation with each dependent factors is in grey. If the correlation is significant, I show the logit regression coefficient. For binary independent factors, a significant coefficient greater than one means that the independent factor being true is correlated with the dependent factor being true and vice versa. For example, participants who looked up their password and typed it in were 87% more likely to find password recall difficult. For numeric independent factors, increasing their value by one is correlated with multiplying the likelihood of the dependent factor being true by the amount of the coefficient. For example, for each additional Part Two recall attempt, the participant was 36% more likely to find Part Two recall difficult.

9.1.2 Analysis Discussion

Table 9.1 shows that many different factors correlate with user sentiment. Participants who found remembering their own real email password difficult were more likely to have negative sentiment about creating and recalling their study password. This suggests that there may be some segment of the population who struggle with passwords in general. Future work might explore categorizing users based on their comfort level with passwords.

Three of the survey questions in Table 9.1 had an option of “I prefer not to answer.” Only two or three percent of participants chose this for any question. In each case, that response was correlated with being less likely to agree with, “If my main email provider had the same password requirements as used in this study, my email account would be more secure.” One possible explanation is that a small segment of the study population is especially conscious of privacy and security. They may be less willing to divulge information in our study, and more likely to create stronger passwords for their real email accounts.

About a third of participants indicated entering their real email password several times per day. These participants were more likely to find creating their study password fun, and less likely to find it annoying. It is possible that these participants are more accustomed to working with passwords, making their study experience more positive.

Older participants were less likely to find password creation difficult or annoying, and less likely to find password recall difficult. Female participants were more likely to find password creation difficult, and less likely to think their study passwords were stronger than their real email

passwords. Participants without a technical background were less likely to find password creation fun. This shows that demographic factors do affect the subjective participant experience.

Participants who entered their study passwords from memory found recall less difficult. Participants who did not use the reminder were also less likely to find recall difficult. Participants who took more tries to recall their password were more likely to find it difficult. These correlations emphasize the link between behavior and sentiment.

This analysis has examined correlations between self-expressed user sentiment on the one hand, and survey responses and observed user behavior on the other. While these correlations sometimes appear as expected, other correlations were not obvious. The findings suggest that passwords research should continue to collect both self-reported sentiment data and observed data. They also highlight the importance of the “I prefer not to answer” option, because participants selecting this option are infrequent but appear to have sentiment that differed from the majority of participants. Because demographics correlated with sentiment in several ways, the findings highlight the importance of continuing to collect demographic information.

The importance of the user-sentiment factors themselves varies by how the accounts are used. For example, password-creation usability is more important for accounts in which users are required to change passwords more often. User perception of password security might be more important for financial services, because users may be more likely to use those services if they trust their security. Overall, these findings suggest that user sentiment is related to several factors. Finding a single observed variable that determined user sentiment would have made future survey-creation easier. However, numerous factors appear to be connected with how users perceive different steps in the password lifecycle.

Factor	Value	Annoying	Creation Difficult	Fun	Recall Difficult	Secure
Remembering the password I use for my real email account is difficult						
Agree	7%	<.001 1.29	<.001 1.52	<.001 1.33	<.001 2.57	.054
How often do you type in your real email password?						
I prefer not to answer	2%	.005 0.69	.006 0.64	.564	<.001 0.49	.001 0.66
Several times per day	33%	<.001 0.85	.939	<.001 1.25	.513	.041
Once per day	15%	.384	.822	.059	.888	.008 1.14
Never	4%	.025	.035	.608	.001 0.72	.311
Less than once a month	8%	.020	.090	.005 0.78	.284	.746
Once per week	8%	.020	.265	.431	.292	.725
Once per month	4%	.057	.054	.447	.902	.329
A few times per month	12%	.022	.910	.100	.236	.097
How did you just enter your password for this study?						
I used a password manager	2%	<.001 0.50	<.001 0.58	.293	.001 0.67	.001 0.68
I used the I forgot my password link	15%	.189	.263	<.001 0.57	<.001 2.81	.867
I typed it in from memory	49%	.327	.082	.745	<.001 0.24	.592
I looked it up and typed it in	15%	.112	.607	.025	<.001 1.64	.458
I copied and pasted it	8%	.397	.462	.708	<.001 1.55	.803
My browser automatically filled it in	5%	.025	.752	.026	.243	.088
Observed Factors						
Creation attempts	2	<.001 1.73	<.001 1.53	<.001 0.83	<.001 1.12	.446
Not Use Password Reminder	85%	.210	.053	<.001 0.58	.008 0.73	.819
Part two recall attempts	2	.069	.436	.877	<.001 1.34	.065
Part one recall attempts	1	.104	.372	.802	.025	.831
Demographics						
Age	31	<.001 0.98	<.001 0.98	.335	.001 0.99	.038
Female	50%	.921	<.001 1.23	.133	.116	<.001 0.88
I prefer not to answer	1%	.026	.772	.655	.954	<.001 0.51
Technical						
No	77%	.241	.341	<.001 0.76	.928	.532
I prefer not to answer	2%	.034	.181	.649	.052	<.001 0.65
Storage Behavior						
Yes, on paper	21%	.002 0.75	.485	.097	.405	.016
Other	4%	.067	.050	.864	<.001 0.62	.154
Yes, electronically	23%	.095	.927	.753	.175	.059
No	55%	.376	.663	.032	.199	.463

Table 9.1: Correlated between factors and self-reported sentiment. P-values are in light grey for all factors and coefficients are shown for significant factors ($\alpha = .01$). Wordings are altered to fit the table.

9.2 Demographics

We asked participants demographics questions in order to understand who was taking our studies. In this section, I present a synopsis of that demographics data. This data serves to provide a context for the results of our research. More importantly, it provides information about the MTurk worker population. This can help other researchers contextualize their own MTurk research, and help researchers creating studies determine whether MTurk is appropriate for their work.

Mean participant age was 30.0 years (median 27). 49.0% of participants were female and 49.8% male; 1.2% selected, “I prefer not to answer.” As a rough measure of technical background, we asked, “Are you majoring in or do you have a degree or job in computer science, computer engineering, information technology, or a related field?” Responses were “Yes” (22.6%), “No” (75.3%), and “I prefer not to answer” (2.1%).

For the study in Chapter 6, we did not filter participants by country. SurveyGizmo automatically performed country detection. People from over 100 countries participated in this study. 51.3% of the 8,000 participants were located in the United States. The next most common country were India (29.5%), Canada (2.1%), United Kingdom (1.7%), Serbia (1.6%), and the Philippines (1.4%). No other nation comprised one percent of our participants.

For the other three studies, we used MTurk to filter out participants not located in the United States. Based on SurveyGizmo’s detection, 97.7% of participants were located in the United States. However, a small proportion of participants, from over 100 countries, appear to have bypassed MTurk’s country filter. The most common was India, with 0.2% of participants. This indicates that MTurk does a fair job of filtering out users located outside the United States. However, if it is critical to a study that all participants be located within the United States, then it is likely advisable to use a secondary check.

We asked participants in Chapters 7 and 8 “What is the highest level of education that you have completed?” 34.1% of participants indicated a high school degree and 16.1% responded with an associate degree. 32.8% of participants indicated a bachelor’s degree and 10.9% indicated a more advanced degree. This shows a diverse range of educational backgrounds among our study participants.

9.3 Mobile Device Usage

We collected user agent string data in SHELF and via SurveyGizmo. To determine which participants were taking the study on a mobile device, we searched the user agent strings we collected during the Part One survey for text that would identify a mobile device, as we described in Section 5.3.1. For the first two studies, 1.1% of participants were using a mobile device in Part One. For the latter two studies, this increased to 4.0%.

We also looked at mobile device usage for participants in the latter two studies who returned for Part Two. Among participants who would return and finish Part Two, 3.5% used a mobile device during Part One. When they returned for Part Two, 6.7% of those participants used a mobile device. Some participants took Part One on a traditional device, and switched to a mobile device for Part Two. A possible explanation is that some participants may have received an email to return on their mobile devices, and took Part Two on that device.

In the Part Two survey, in the studies in Chapters 7 and 8, we asked participants “On what sort of computer or device have you just entered your password?” Of the 18,941 responses, 57.6% responded with a laptop computer, 33.2% with a desktop, 4.9% with a smartphone, and 3.13% with a tablet. The percentage of participants who reported using a mobile device is slightly higher than the percentage we detected using a mobile device. Manually looking at their user agent strings, there did not appear to be a sign that they were using a mobile device. Some participants may have configured their devices to use non-mobile user agent strings to avoid being shunted to mobile websites when browsing the web. Some participants may have been using mobile browsers that did not indicate being mobile browsers. Further, some participants may have reported a small laptop as a mobile device.

Part Two participants in Chapters 7 and 8 who reported recalling their password on a mobile device took more attempts to do so (1.7 to 1.9 attempts) (KW $\chi^2_1=27.11$, $p<.001$). They were not more likely to find recall difficult ($\chi^2_1=0.321$, $p=0.571$). These findings suggest that researchers continue to monitor what kind of devices participants are using to recall their passwords. These findings also suggest future work focused on how device usage affects password recall.

9.4 Real Email Passwords

Our studies asked participants to imagine creating a new password for their real email account. We asked questions about their real email accounts both to prime them to think about those accounts, and to learn more about those accounts. In order to understand the type of email accounts our participants were using, we asked them during the studies in Chapters 7 and 8. The majority of participants, 78.9%, were using a web email account. 8.1% were using a university account, and 3.9% were using a work account.

We asked participants who returned for Part Two whether they agreed with, “Remembering the password I use for my real email account is difficult.” Only 7.1% of participants found recalling their real email passwords difficult. Conversely, 35.3% of participants reported that recalling their study passwords difficult. This difference could be the result of study password requirements being more demanding than most real email service providers. However, this could also be the result of participants being more familiar with their real email passwords.

Several questions asked participants how they entered their real email passwords. The first two studies (presented in Chapters 5 and 6) asked participants in Part Two about password storage. 53.7% indicated not writing down their real email password. 22.9% stored it electronically, and 19.3% on paper. The latter two studies (presented in Chapters 7 and 8) asked participants about how they most often entered their real email password. 69.0% of participants typed it from memory. We also asked participants in the latter two studies which computer or device they most often used to enter their password. The most common responses were a laptop (55.4%) or desktop (33.7%) computer. Mobile devices were less common – smartphones were 6.3% and tables 2.5%. These results help shed light on how users treat their real email passwords.

We asked participants in the first two studies how many characters of each class were in their real email password. Based on median values, the average real email password had six lowercase letters, two digits, and no uppercase letters or symbols. Because we thought participants might be concerned about disclosing that much detail about their real email passwords, we adjusted this question for the study presented in Chapter 7. That question asked participants to choose their

password length from a range of options. The most common range was between seven and nine characters (39.5%), followed by 10 to 12 (26.9%). 11.3% of participants preferred not to answer, indicating that a sizable number of participants did not want to disclose potentially sensitive password details. This recommends that passwords research offer participants the option to avoid answering potentially sensitive questions, especially when those questions are not necessary for answering the study's research questions.

The studies in Chapters 5 and 6 asked participants, "Approximately how long ago did you last change your real email password?" We changed this question slightly for the study in Chapter 7 to ask when they last created or changed their password. 17.1% of participants reported creating or changing their password within the past month. Over a quarter of participants had done so within the past six months. However 18.8%, had not created or changed their password within the past year. These findings show a range of password-creation frequencies. System administrators should consider how often their users create passwords when evaluating password-study results. In particular, the more frequently users create passwords, the more password-creation usability should be weighted.

In order to understand password-related behavior better, the study in Chapter 7 asked participants with how many people they had shared their real email password. 35.1% of these 22,491 participants declined to answer this question. This emphasizes that many participants are uncomfortable answering sensitive password-related questions. Among those who did answer, 65.5% shared it with no one else and 25.3% shared it with one other person. 5.5% shared their password with two people, and few shared it with more than two. The fairly high rate of sharing with exactly one person might be somewhat explained by [89], which found that couples often shared passwords as a sign of trust.

All of our studies have included password recall as a usability metric. This metric is more important in contexts where users enter their passwords more often. We asked participants in Part Two how often they typed in their real email password. 32.7% of participants reported typing in their real email password several times per day and another 14.6% once per day. This suggests that password-recall usability is an important metric for evaluating password policies. System administrators should consider how frequently their own users enter passwords when applying study results to their own systems.

The studies in Chapters 5 and 6 asked participants in Part Two, "Have you ever used the password that you created for this study for any other account?" The majority, 74.2%, responded that they had never used the password before. 19.3% reported making a password that was similar to a prior password, and only 5.0% reported exact reuse. The higher proportion of participants reusing a similar password, rather reusing a password exactly, may result from our studies having more strict requirements than most real-life service providers. Participants were more likely to agree with Part Two recall being difficult if they had created an entirely new password (39.8%) than if they had reused or partially reused a password (32.7%) ($\chi^2_1=66.93, p<.001$).

Chapter 10

Practical Recommendations for System Administrators

This thesis has presented several human-subjects studies contrasting password-composition policies and other password-creation factors. In this section, I distill into a few pages those findings I consider especially important for the system administrators who are charged with selecting their organizations' password-composition policies. I assume the reader wishes to select a policy resulting in relatively secure passwords. The reader might wish to improve an existing policy or the reader might wish to create an entirely new password-composition policy. Florencio et al. [27] recently wrote a detailed primer that discusses several findings of interest to system administrators. This chapter focuses on helping administrators selecting a password-composition policy.

A password-composition policy is only one part of a complete approach to protecting user accounts. A strong password-composition policy can help protect users in the specific case of an attacker who has stolen a hashed password file and is making guesses against that file. In addition, passwords stand out as a piece of system security over which individual users often feel some ownership [85]. Moreover, a strong and usable password policy can help protect users while reducing their workload, and help administrators by reducing the number of password-reset requests.

I discuss threat models in Section 10.1. This includes the attack model against which password-composition policies can help defend, and attack models where they are not helpful. I also discuss other factors for the administrator to consider to protect against password cracking. I discuss factors that should be considered for choosing a password-composition policy in Section 10.2. I conclude this chapter with Section 10.3.

10.1 Password Cracking and What You Can Do About It

Password-composition policies are one component of helping to protect user accounts. This section presents an attack model against which a good password-composition policy can help defend. This section also discusses several steps that an administrator can take to protect users and their passwords, beyond using a good password-composition policy.

10.1.1 Offline Attacks

An attacker making live guesses on a website is conducting an *online attack*. To protect against this, the system can monitor website activity and lock out a user after multiple failed password attempts. In the attack model we discuss in this thesis, the attacker has acquired a copy of the password file that stores user passwords. The passwords in that file should be salted and hashed, as we will discuss below. However, the attackers can still use that file to determine, or crack, user passwords; this is called an *offline attack*.

In an offline attack, the attackers make a guess for what a given user's password might be. Then, the attackers salt and hash that guess using the same salt and hash as the file. If the result matches what is in the file, then the attackers know they have guessed the user's password, barring an unlikely hash collision. Attackers with powerful hardware can generate a larger number of guesses in less time. Fortunately, a good password-composition policy can help keep users' passwords safe even after a large number of guesses.

10.1.2 Password Hashes

A hash is a one-way function; the function takes in a password and returns an unrelated string. Because a hash is a function, the same password input always produces the same output. With a properly designed hash, an attacker cannot translate hashed passwords back into user passwords. If the passwords in the file are not hashed, then an attacker who has acquired the password file has already acquired the user passwords.

Administrators should salt their users' passwords before hashing them. A salt is a small string, different for each user, added to each password before it is hashed. Using a salt makes cracking hashed passwords more difficult for the attacker. For example, if Alice and Bob both make the same password, if those passwords are not salted before they are hashed, then Alice and Bob will have identical password hashes. By salting both of their passwords, their hashed passwords look different. This prevents the attacker from, for example, cracking Alice's password and then automatically knowing Bob's password.

Not all hashes are equal. Because attackers need to hash each guess they make, using a slower hash function makes it take much longer for attackers to crack passwords. Because an attacker often needs to make many guesses to crack a password, increasing the per-guess hashing time can slow the attacker very much while only adding a small slowdown to authenticating legitimate users.

bcrypt is a slow hashing scheme well-suited for storing passwords.¹ I estimate that a laptop would take over 2,000 years to hash 10^{12} password guesses using bcrypt. I also estimate that 10^{12} hashes would take a hacktivist group about 207 days, and a criminal organization closer to 27 days. The time taken to hash passwords can be increased exponentially by adjusting an input parameter, the cost factor. As computational power increases, the computational capacity required to crack passwords can be increased to match [72]. Another advantage of bcrypt is that a number of implementations already exist, and they include automated password salting. For example, a version of bcrypt has become the default for authentication with Ruby on Rails.²

¹<http://bcrypt.sourceforge.net>

²api.rubyonrails.org/classes/ActiveModel/SecurePassword/ClassMethods.html

10.1.3 Other Factors for Password Security

There are many ways that attackers could compromise user accounts, some of which do not depend on password strength. For example, if a system allows authentication via “security questions,” then accounts are no more secure than those questions. Other attacks, such as shoulder-surfing and phishing, can allow attackers to learn users’ passwords directly, meaning that their complexity is not helpful.

It is also worth mentioning password expiration, the practice of requiring new passwords after a period of time. The motivation behind forcing password changes is that if an attacker has acquired a victim’s password but has not yet acted, then changing the password would thwart the attack. Periodic password expiration differs from explicitly revoking passwords in response to an incident. A forced password reset is a good step if there is reason to believe that a password or password file has become compromised.

However, the value of routine password expiration is questionable. Unless a password or its hash is known to an attacker, forcing a reset does little good. Researchers examined genuine university passwords and found that users replaced an expired password with one very similar. The most common transformations included incrementing or moving around a digit and replacing one symbol with another. Given the old passwords, the authors were able to crack 41% of the subsequent password in a matter of seconds [100]. It does not appear that users created stronger passwords when forced to change them, and there is reason to believe that password expiration would be difficult on users without tangible security gains. It is likely more useful to implement monitoring of the server on which the hashed-and-salted password file is stored. This way, if a breach is detected, passwords can then be reset.

10.2 Password-Composition Policies

This section discusses findings from the research presented in this thesis in order to help system administrators create and improve password-composition policies. This section discusses some of the most successful password-composition policies that this thesis studied. For administrators who want to retain but improve their current password-composition policies, this section concludes with steps for improving existing password-composition policies.

10.2.1 Password Instructions and Feedback

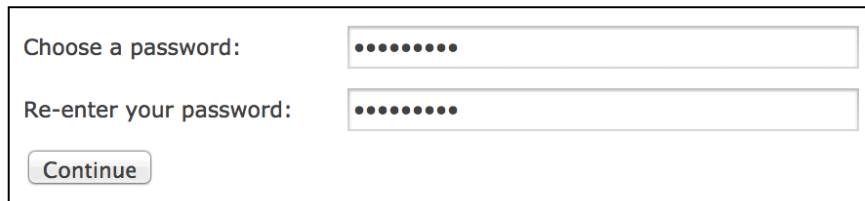
Giving users real-time password-requirements feedback as they create their passwords can help make a strict policy more palatable and reduce password-creation errors. This can be, for example, putting a green checkmark next to fulfilled requirements, and stating in red text which requirements are not yet met. Real-time feedback can be applied to new or existing password-composition policies. An example of real-time password-creation feedback is in Figure 10.1.

10.2.2 System-Assigned Passwords

System-assigned passwords allow the administrator more control over password strength. However, users struggle with them. Compared to user-selected passwords, system-assigned passwords

Password requirements:

- Include at least 12 characters **(Your password contains 9 characters but 12 are required.)**
- Password must both **begin** and **end** with a **lowercase** letter (a-z) **(Your password must begin and end with a lowercase letter)**
- Include at least 3 of the following: **(Your password contains 2 types of characters but 3 are required.)**
 - A lowercase English letter
 - An uppercase English letter
 - A digit
 - A symbol (something that is not a digit or an English letter)



The screenshot shows a web form for password creation. It has two input fields: "Choose a password:" and "Re-enter your password:". Both fields contain a password of "password!". The "Choose a password:" field has a red error message below it: "(Your password contains 9 characters but 12 are required.)". The "Re-enter your password:" field has a red error message below it: "(Your password must begin and end with a lowercase letter)". A "Continue" button is located below the input fields.

Figure 10.1: An example of real-time password-creation feedback, as “password!” is entered.

and passphrases were more likely to be difficult to learn and recall, and more likely to be written down. System-assigned passwords are unlikely to be suitable for a university setting, except for employees who access sensitive data. However, there are scenarios where system-assigned passwords may be a reasonable option. They allow the administrator to ensure that each user’s password meets a minimum standard of strength. They can also allow strong yet shorter passwords, which might be useful for scenarios where passwords must be entered frequently.

10.2.3 The Traditional “Strong” Policy

The studies in this thesis examined the traditional “strong” password-composition policy. Such a policy is currently in use at Carnegie Mellon University, and is recommended by the *InCommon* Federation.³ This thesis’ implementation of this policy required that passwords have at least eight characters and three different character classes (of uppercase letters, lowercase letters, digits, and symbols). It also required that passwords not be found in a password-cracking dictionary. This is called *comp8* because it required eight characters and a comprehensive set of other requirements. *comp8* is similar to a number of policies in use elsewhere. For example, *outlook.com* requires eight characters and two character classes.⁴ The Windows *complexity requirements*, if enabled, require at least three different character classes.⁵

The studies in this thesis found that *comp8* was neither especially secure nor easy to use. Passwords created under this policy stood up well against a resource-constrained attacker making a smaller number of guesses. However, once an attacker made more guesses, 10^{12} , about a quarter

³<http://www.incommon.org>

⁴<https://signup.live.com/>

⁵[http://technet.microsoft.com/en-us/library/hh994562\(v=ws.10\).aspx](http://technet.microsoft.com/en-us/library/hh994562(v=ws.10).aspx)

of the passwords were guessed. Part of why this policy was not very secure was that users often met its requirements in predictable ways. Most passwords started with a capital letter and used no other capital letter. Half used only one symbol and placed it either last or second-to-last; a fifth used ! as that one symbol.

This condition was also more onerous on participants than some more-secure policies. This includes observed usability metrics, such as the number of password-creation attempts. It also includes self-reported usability metrics, such as perceived password-creation difficulty. Fortunately, there are conditions that were more usable and more secure than *comp8*.

10.2.4 Combining Character and Length Requirements

Overall, our studies found that policies should require long passwords, with a minimum length of 12 to 16 characters. However, when participants were asked to make a password with only a length requirement, many opted to make very simple, easily guessed passwords such as “baseballbaseball.” Adding character-class requirements helped prevent these simple passwords. The following two password policies stood out as more usable and secure than *comp8*.

Both *2word16* and *3class12* were successful password policies that combined longer-length and character-class requirements. The *2word16* policy required 16 characters, and required that passwords contain two “words” – two strings of one or more letters divided by a string of one or more non-letters. Thus, “baseballbaseball” was not a valid password, but “baseball baseball” was. The *3class12* policy required at least 12 characters with three character classes. Both were more usable and more secure than *comp8*. We believe that either policy would be a good choice for many service providers. *3class12* was the more usable of the two, and *2word16* was more secure.

These findings indicate that service providers should not impose any password-length *maximum* that users are likely to reach. They also suggest that service providers should not restrict users from including different character classes in their passwords.

10.2.5 Using a Substring Blacklist

Many cracked passwords shared common substrings. Therefore, this thesis tested including a *substring* blacklist. Unlike a dictionary check, the substring blacklist prevents passwords from containing certain substrings. A set of blacklisted substrings might be as follows.

```
Service provider name, city, etc.
Any year between 1950 and 2049
The same character four or more times in a row
Any four consecutive characters from "password"
Any four sequential digits (e.g., 5678)
Any four sequential letters in the alphabet (e.g., wxyz)
Any four consecutive characters on the keyboard (e.g., wsxc)
```

Using the substring blacklist made passwords take more tries to create, and made password creation more difficult. However, it did not make password recall any more difficult or less successful. Thus, the blacklist trades creation-time usability for more security.

10.2.6 Improving an Existing Policy

There may be administrators who have a good justification for their current password-composition policies. Rather than using the findings in this thesis to replace the current policy, those administrators may wish to use these findings to enhance or improve their current password-composition policies. Real-time feedback can be applied to requirements to let users know whether they have fulfilled those requirements. For the requirements themselves, the findings in this thesis strongly suggest requiring longer passwords. The length-eight policies we tested had poor performance against an attacker capable of making a large number of guesses, such as 10^{12} . Password length appears to make passwords more secure against such an attacker. Therefore, an administrator adjusting an existing password-composition policy might consider increasing the length requirement to at least 12. In addition, the administrator should also consider the other password-related advice given in this chapter: storing passwords salted and hashed, monitoring live guesses and the server on which the password file is stored, and using a slow hashing algorithm like `bcrypt`.

10.3 Concluding Remarks

Password-composition policies are an important tool that system administrators can use to help keep users secure. These policies are not the only factor that administrators need to consider to protect user accounts, but they are an important part of protecting them. Administrators should consider policy tradeoffs and select a policy that works best for their own particular environments. The list below highlights recommendations from this chapter.

Key Recommendations:

- Any alternate authentication scheme, like security questions, should be as secure as passwords.
- Salt and hash stored passwords. Use unique salts and a slow hash like `bcrypt`.
- Don't require users to change their passwords unless you suspect a breach.
- Avoid the traditional "strong" policy requiring 8 characters and complex character-class requirements. There are alternatives that are more usable and more secure.
- For a usable and secure password policy, consider `3class12`: At least 12 characters and three different character classes.
- For higher-security environments, consider the more secure but less usable `2word16`: At least 16 characters with at least two "words" of letters separated by one or more non-letters.
- To increase security further, consider adding a substring blacklist to the password policy.
- To make password creation easier, consider giving users real-time feedback about whether they have met the requirements.

Chapter 11

Conclusion

This chapter summarizes the work in this thesis. I present major findings and highlight the primary contributions of each section. Then, I present the future work that our findings suggest.

11.1 Data-Collection Protocol

This subsection discusses my contribution to general online data collection. The next subsection discusses my specific contribution to the study of password policies. I helped to lead crowdsourced, large-scale human-subjects studies with randomly assigned, controlled conditions. Participants stepped through a sequence of webpages, which could vary based on their condition. This methodology combined advantages from other data-collection protocols, as discussed in Chapter 2. Similar to a laboratory study, it used randomly assigned, controlled conditions that enabled drawing causal conclusions. Similar to an online survey, it collected data from a very large number of participants across a broad geographic space, without needing researchers to monitor individual participants.

In order to facilitate studies using this methodology, I created a generalized framework for conducting online crowdsourced studies. Chapter 3 described this framework, called SHELF. SHELF helps researchers prepare structured online human-subjects experiments. SHELF then interfaces with MTurk to automate recruiting and paying participants. While data is being collected, SHELF helps monitor study progress, giving researchers information such as how many participants finished the study in each condition. After data collection, SHELF facilitates exporting study data for analysis.

11.2 Analyzing Password Policies

The work in this thesis applied the data-collection framework described in Chapter 3 to the study of password-composition policies. Chapter 4 described the methodology those studies shared. We recruited participants through MTurk and assigned them to different password-composition policies. We asked participants to create or learn their passwords. Participants created, or were assigned, a password under a password policy that varied by their condition. Participants filled out

a brief survey and then recalled their password. Two days later, we invited them to return for Part Two. Returning participants again recalled their password and then filled in another survey.

As discussed in Section 2.5, prior human-subjects passwords research tended to fit into one of two categories. Either the research was a laboratory study with a relatively small number of participants; or the research studied participants using their real-life passwords. We studied large groups of participants creating and using passwords under randomly assigned, controlled conditions. This enabled us to draw causal conclusions about the security and usability effects of different password-composition policies.

11.3 Studies of Password-Composition Policies

The research in this thesis has contributed a better understanding of the effects of password-composition policies. Chapters 5, 6, 7, and 8 illustrated how different policies affected user sentiment, user behavior, and password strength. Under very simple conditions, participants tended to create weak, easily guessed passwords. Conditions that led to stronger passwords than these simple conditions were also less usable. However, one policy being less usable than another did not necessarily mean that the less-usable policy led to stronger passwords. Some policies made password creation or recall more difficult without increasing password strength.

Chapter 5 examined system-assigned passwords and passphrases. This included short passwords and pronounceable passwords, as well as passphrases constructed of words drawn at random from dictionaries. Participants struggled with learning and recalling these system-assigned passwords and passphrases. Passphrases failed to out-perform passwords on any usability metric, including self-reported user sentiment and observed recall behavior. Passwords often appeared more usable than passphrases; for example, participants recalled passwords with fewer mistakes in less time. Post-facto analysis suggested that some metrics for passphrase recall could be improved through automatic error correction. However, even with this improvement, passphrases did not out-perform passwords.

Chapter 6 examined user-created passwords, focusing on password policies used in practice, including Carnegie Mellon University's policy. Password-composition policies made significant differences in both password security and usability. If a dictionary check was used, then the contents of the dictionary had a large impact on its effectiveness. A policy requiring sixteen characters was more usable, and less likely to be guessed after a large number of guesses, than a traditional "strong" policy that required eight characters, three character classes, and a dictionary check. Participants often met the requirements of the traditional policy in predictable ways, such as placing an uppercase letter at the beginning. However, there was a shortcoming of the length-16 policy as well: a large number of passwords created under this policy were very simple and quickly guessed.

A better policy would have the advantages of the length-16 requirement, but prevent many of the easily guessed passwords. Chapter 7 presented two studies that examined this. These studies contrasted minimum lengths, character-class requirements, substring blacklists, and structural requirements. Combining the length-16 requirement with a requirement that passwords contain two strings of letters separated by one or more non-letters helped reduce the number of weak passwords. In addition, requiring 12 characters with two or three character classes led to a generally favorable tradeoff between security and usability.

The first study in Chapter 7 found common characteristics among many weak passwords. For example, they often began and ended with characters other than lowercase letters, and many shared a relatively small set of substrings. The second study of Chapter 7 tested conditions that proactively prevented these markers of weak passwords. These included conditions that prohibited common substrings and conditions that required passwords start and end with lowercase letters. We also examined additional combinations of length and character-class requirements. The substring blacklist stood out because it made passwords less likely to be guessed without having any adverse effect on password recall.

The above studies focused on password-composition requirements. Chapter 8 presented a study on how those requirements might be made more palatable. It studied the security and usability effects of real-time password-creation feedback. This included, for example, placing a green checkmark next to fulfilled password-composition requirements, and presenting unfulfilled requirements in red text. Real-time requirements feedback helped participants create passwords under strict requirements with fewer mistakes. While most conditions in this thesis asked participants to create a password all at once, this study included conditions that guided participants through a multi-step password-creation process. Guiding participants through password creation made the process easier, but resulted in passwords that were more likely to be guessed.

11.4 Understanding Participants and their Sentiment

Each study chapter analyzed data by condition to contrast password policies. Chapter 9 examined data from all four studies to understand better participants and their sentiment. Section 9.1 analyzed how observed factors and survey responses correlated with user sentiment. User sentiment correlated significantly with a large number and variety of factors we collected.

Chapter 9 also presented information about study participants. This included demographic information, use of mobile devices for our studies, and information about the real email passwords of our participants. This provided context for the research in this thesis, and a large-scale demographics description for MTurk research.

11.5 Concrete Advice for Service Providers

Chapter 10 presented concrete advice for service providers. This advice distilled highlights from across the studies in this thesis into actionable recommendations. For example, several policies offered both security and usability advantages over the traditional “strong” password policy. . Because different use-cases place different demands on users, I discussed how service providers could best apply our findings to their own user base and services. I believe that this chapter can help service providers better protect their users.

11.6 Future Work

We have only begun to scratch the surface of password-creation feedback and guidance. Future work might make a further exploration of the space of real-time password requirements feedback. Chapter 8 demonstrated that one particular implementations of password guidance resulted in

weaker passwords. However, guided conditions had several usability benefits. Future work might examine additional implementations of password-creation guidance to retain those usability benefits without sacrificing password strength.

Chapter 5 studied system-assigned passwords and passphrases. The passphrases were composed of words drawn from dictionaries of common words. As the dictionaries increased in size, the entropy of the resulting passphrases increased. While the system-assigned passphrases were generally difficult to recall, increasing their entropy did not appear to increase that difficulty. Therefore, it is possible that passphrases constructed using very large dictionaries could still be usable. These high-entropy passphrases might be useful in scenarios in which all passwords need to be very secure. Future work might examine the usability of high-entropy system-assigned passphrases.

Many participants had a notion of password strength that did not match our empirical results. For example, participants over-estimated the strength of “traditional” comprehensive password requirements. Future work might study how users reason about password strength and the associated threat models. Chapter 8 demonstrated that what participants see as they make their passwords, in addition to the requirements themselves, can affect their passwords. It is possible that by learning more about how participants think about threats to their passwords, service providers will be able to provide more effective messaging during password creation.

Finally, the proliferation of mobile-device usage suggests new approaches to studying passwords and authentication. Participants overwhelmingly used desktop and laptop computers – devices with traditional keyboards. Future work might examine whether and how password-entry differs on mobile devices. In addition, mobile devices facilitate two-factor authentication, a promising development for increasing account security. For example, Google¹ and Microsoft² now offer a two-factor authentication or *two-step verification* service. When a user authenticates to a new device, the service provider sends that user a code on his or her mobile device. The user then enters the usual password as well as that one-time code. An attacker in this model would need to acquire both the user’s password and the user’s mobile device. Future research could investigate the usability implications of this sort of two-factor authentication.

¹<https://www.google.com/landing/2step/>

²<http://windows.microsoft.com/en-us/windows/two-step-verification-faq>

Bibliography

- [1] ADAMS, A., SASSE, M. A., AND LUNT, P. Making passwords secure and usable. In *HCI* (1997).
- [2] ADOBE. Customer security alert. <http://helpx.adobe.com/x-productkb/policy-pricing/customer-alert.html>, 2013.
- [3] BARD, G. V. Spelling-error tolerant, order-independent pass-phrases via the Damerau-Levenshtein string-edit distance metric. In *ACSW* (2007), pp. 117–124.
- [4] BERINSKY, A. J., HUBER, G. A., AND LEN, G. S. Using Mechanical Turk as a subject recruitment tool for experimental research. *Political Analysis* (2011).
- [5] BISHOP, M., AND V.KLEIN, D. Improving system security via proactive password checking. *Computers & Security* 14, 3 (1995), 233–249.
- [6] BONNEAU, J. The Gawker hack: How a million passwords were lost. <http://www.lightbluetouchpaper.org/2010/12/15/the-gawker-hack-how-a-million-passwords-were-lost/>, December 2010.
- [7] BONNEAU, J. You can never have too many passwords: Techniques for evaluating a huge corpus. *IEEE Symposium on Security and Privacy* (2012).
- [8] BONNEAU, J., HERLEY, C., VAN OORSCHOT, P. C., AND STAJANO, F. The quest to replace passwords: A framework for comparative evaluation of Web authentication schemes. In *IEEE SP* (2012).
- [9] BONNEAU, J., AND SHUTOVA, E. Linguistic properties of multi-word passphrases. In *USEC* (2012).
- [10] BRANTZ, T., AND FRANZ, A. The Google Web 1T 5-gram corpus. Tech. rep., Linguistic Data Consortium, 2006.
- [11] BRIGHT, P. ‘Military meltdown Monday’: 90k military usernames, hashes released. <http://arstechnica.com/tech-policy/2011/07/military-meltdown-monday-90k-military-usernames-hashes-released/>, 2011.
- [12] BROWN, A. S., BRACKEN, E., ZOCCOLI, S., AND DOUGLAS, K. Generating and remembering passwords. *Appl. Cognit. Psychol.* (2004).

- [13] BRYANT, K., AND CAMPBELL, J. User behaviours associated with password security and management. *Australasian Journal of Information Systems* 14, 1 (2006).
- [14] BUHRMESTER, M., KWANG, T., AND GOSLING, S. D. Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data? *Persp. Psych. Sci.* 6, 1 (2011), 3–5.
- [15] BURR, W. E., DODSON, D. F., NEWTON, E. M., PERLNER, R. A., POLK, W. T., GUPTA, S., AND NABBUS, E. A. Electronic authentication guideline. Tech. rep., National Institute of Standards and Technology, 2011.
- [16] BURR, W. E., DODSON, D. F., AND POLK, W. T. Electronic authentication guideline. Tech. rep., National Institute of Standards and Technology, 2006.
- [17] CHIASSON, S., FORGET, A., STOBERT, E., VAN OORSCHOT, P. C., AND BIDDLE, R. Multiple password interference in text passwords and click-based graphical passwords. In *CCS* (2009).
- [18] CRADDOCK, D. Hey! My friend's account was hacked! http://windowsteamblog.com/windows_live/b/windowslive/archive/2011/07/14/hey-my-friend-s-account-was-hacked.aspx, 2011.
- [19] CUBRILOVIC, N. RockYou hack: From bad to worse. <http://techcrunch.com/2009/12/14/rockyou-hack-security-myspace-facebook-passwords/>, December 2009.
- [20] DAVIES, M. The corpus of contemporary American English: 425 million words, 1990–present. <http://corpus.byu.edu/coca/>, 2008.
- [21] DELL'AMICO, M., MICHIARDI, P., AND ROUDIER, Y. Password strength: An empirical analysis. In *INFOCOM* (2010).
- [22] DOWNS, J. S., HOLBROOK, M. B., SHENG, S., AND CRANOR, L. F. Are your participants gaming the system? Screening Mechanical Turk workers. In *CHI* (2010).
- [23] EGELMAN, S., SOTIRAKOPOULOS, A., MUSLUKHOV, I., BEZNOSOV, K., AND HERLEY, C. Does my password go up to eleven? *CHI* (2013).
- [24] ENGBERG, D. Security notice: Service-wide password reset. <http://blog.evernote.com/blog/2013/03/02/security-notice-service-wide-password-reset/>, March 2013.
- [25] EXPERIAN. Illegal web trade of personal information soars to record highs. <https://www.experianplc.com/media/news/2014/illegal-web-trade-of-personal-information-soars-to-record-highs/>, October 2014.
- [26] FLORENCIO, D., AND HERLEY, C. A large-scale study of web password habits. In *WWW* (2007).

-
- [27] FLORENCIO, D., HERLEY, C., AND VAN OORSCHOT, P. An administrator's guide to Internet password research. In *USENIX LISA* (2014).
- [28] FLORENCIO, D., HERLEY, C., AND VAN OORSCHOT, P. Password portfolios and the finite-effort user: Sustainably managing large numbers of accounts. In *USENIX* (2014).
- [29] FORGET, A., CHIASSON, S., VAN OORSCHOT, P. C., AND BIDDLE, R. Improving text passwords through persuasion. In *SOUPS* (2008).
- [30] GASSER, M. A random word generator for pronounceable passwords. Tech. Rep. ESD-TR-75-97, The MITRE Corporation, 1975.
- [31] GAW, S., AND FELTEN, E. W. Password management strategies for online accounts. In *SOUPS* (2006).
- [32] GOODIN, D. Hackers expose 453,000 credentials allegedly taken from Yahoo service. <http://arstechnica.com/security/2012/07/yahoo-service-hacked/>, July 2012.
- [33] GRAWEMEYER, B., AND JOHNSON, H. Using and managing multiple passwords: A week to a view. *Interacting with Computers* 23, 3 (June 2011).
- [34] HEALTHY PASSWORDS. Bloggtoppen.se 90000 passwords revealed. http://www.healthypasswords.com/news.Bloggtoppen.se_90000_Passwords_Revealed.html, October 2011.
- [35] HERLEY, C., AND VAN OORSCHOT, P. A research agenda acknowledging the persistence of passwords. *IEEE Security and Privacy* 10, 1 (2012), 28–36.
- [36] HONAN, M. How Apple and Amazon security flaws led to my epic hacking. <http://www.wired.com/2012/08/apple-amazon-mat-honan-hacking>, August 2012.
- [37] HORTON, J. J., RAND, D. G., AND ZECKHAUSER, R. J. The online laboratory: Conducting experiments in a real labor market. *Experimental Economics* (2010).
- [38] INCOMMON FEDERATION. Identity assurance profiles bronze and silver v1.1, 2011.
- [39] INGLESANT, P., AND SASSE, M. A. The true cost of unusable password policies: Password use in the wild. In *ACM Conference on Human Factors in Computing Systems* (2010).
- [40] INTERNATIONAL BUSINESS TIMES. Sega hacked: 1.3 million users' information compromised. <http://www.ibtimes.com/sega-hacked-13-million-users-information-compromised-291925>, June 2011.
- [41] IPEIROTIS, P. G. Demographics of Mechanical Turk. Tech. rep., New York University, 2010.
- [42] JAKOBSSON, M., AND AKAVIPAT, R. Rethinking passwords to adapt to constrained keyboards. *IEEE MoST* (2012).

- [43] KEITH, M., SHAO, B., AND STEINBART, P. A behavioral analysis of passphrase design and effectiveness. *Journal of the Association for Information Systems* 10, 2 (2009), 63–89.
- [44] KELLEY, P. G., KOMANDURI, S., MAZUREK, M. L., SHAY, R., VIDAS, T., BAUER, L., CHRISTIN, N., AND CRANOR, L. F. The impact of length and mathematical operators on the usability and security of system-assigned one-time pins. In *Financial Cryptography and Data Security*. Springer, 2013, pp. 34–51.
- [45] KELLEY, P. G., KOMANDURI, S., MAZUREK, M. L., SHAY, R., VIDAS, T., BAUER, L., CHRISTIN, N., CRANOR, L. F., AND LOPEZ, J. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In *IEEE Security and Privacy* (2012).
- [46] KITTUR, A., CHI, E. H., AND SUH, B. Crowdsourcing user studies with Mechanical Turk. In *CHI* (2008).
- [47] KOMANDURI, S., SHAY, R., CRANOR, L. F., AND SCHECHTER, S. Telepathwords: Preventing weak passwords by reading users’ minds. In *USENIX* (2014).
- [48] KOMANDURI, S., SHAY, R., KELLEY, P. G., MAZUREK, M. L., BAUER, L., CHRISTIN, N., CRANOR, L. F., AND EGELMAN, S. Of passwords and people: Measuring the effect of password-composition policies. In *CHI* (2011).
- [49] KUMAR, N. Password in practice: An usability survey. *Journal of Global Research in Computer Science* 2, 5 (2011).
- [50] KUMPARAK, G. Vudu headquarters robbed, hard drives with private customer data stolen. <http://techcrunch.com/2013/04/09/vudu-headquarters-robbed-hard-drives-with-private-customer-data-stolen/>, April 2013.
- [51] KUO, C., ROMANOSKY, S., AND CRANOR, L. F. Human selection of mnemonic phrase-based passwords. In *SOUPS* (2006).
- [52] KURZBAN, S. A. Easily remembered passphrases: A better approach. *SIGSAC Rev.* 3 (1985).
- [53] LEONHARD, M., AND VENKATAKRISHNAN, V. A new attack on random pronounceable password generators. In *IEEE EIT* (2007).
- [54] LEONHARD, M. D., AND VENKATAKRISHNAN, V. N. A comparative study of three random password generators. In *IEEE EIT* (2007).
- [55] LORD, B. Keeping our users secure. <http://blog.twitter.com/2013/02/keeping-our-users-secure.html>, February 2013.
- [56] MALVONI, K., AND KNEZOVIC, J. Are your passwords safe: Energy-efficient bcrypt cracking with low-cost parallel hardware. In *USENIX Workshop on Offensive Technologies* (2014).

- [57] MASSEY, J. L. Guessing and entropy. In *IEEE International Symposium on Information Theory* (1994).
- [58] MAZUREK, M. L., KOMANDURI, S., VIDAS, T., BAUER, L., CHRISTIN, N., CRANOR, L. F., KELLEY, P. G., SHAY, R., AND UR, B. Measuring password guessability for an entire university. In *CCS* (2013).
- [59] MEDLIN, B. D., CAZIER, J. A., AND FOULK, D. P. Analyzing the vulnerability of us hospitals to social engineering attacks: how many of your employees would share their password? *International Journal of Information Security and Privacy (IJISP)* 2, 3 (2008), 71–83.
- [60] MEHLER, A., AND SKIENA, S. Improving usability through password-corrective hashing. In *SPIRE* (2006).
- [61] MILLER, G. Note on the bias of information estimates. *Info. Th. Psych.: Problems and Methods* (1955).
- [62] MILMAN, D. A. Death to passwords. http://blogs.computerworld.com/17543/death_to_passwords, 2010.
- [63] MOSHFEGHIAN, S., AND RYU, Y. S. A passport to password best practices. *Ergonomics in Design: The Quarterly of Human Factors Applications* 20, 2 (2012), 23–29.
- [64] MUNROE, R. xkcd: Password strength. <https://www.xkcd.com/936/>, 2012.
- [65] NIST. Federal information processing standards publication 181: Automated password generator (APG). Tech. rep., NIST, 1993.
- [66] NOTOATMODJO, G., AND THOMBORSON, C. Passwords and perceptions. In *ACIS* (2009).
- [67] PAUL, I. Update: LinkedIn confirms account passwords hacked. http://www.pcworld.com/article/257045/6_5m_linkedin_passwords_posted_online_after_apparent_hack.html, June 2012.
- [68] PERLROTH, N. LivingSocial hack exposes data for 50 million customers. <http://bits.blogs.nytimes.com/2013/04/26/living-social-hack-exposes-data-for-50-million-customers/>, April 2013.
- [69] PLIAM, J. O. On the incomparability of entropy and marginal guesswork in brute-force attacks. In *Progress in Cryptology—INDOCRYPT 2000*. Springer, 2000, pp. 67–79.
- [70] PORTER, S. N. A password extension for improved human factors. *Computers and Security* 1, 1 (1982).
- [71] PROCTOR, R. W., LIEN, M.-C., VU, K.-P. L., SCHULTZ, E. E., AND SALVENDY, G. Improving computer security for authentication of users: Influence of proactive password restrictions. *Behavior Research Methods, Instruments, & Computers* 34, 2 (2002), 163–169.

- [72] PROVOS, N., AND MAZIERES, D. A future-adaptable password scheme. In *USENIX* (1999), pp. 81–91.
- [73] RAGAN, S. Adobe confirms stolen passwords were encrypted, not hashed. <http://www.csoonline.com/article/2134124/network-security/adobe-confirms-stolen-passwords-were-encrypted-not-hashed.html>, November 2013.
- [74] RILEY, S. Password security: What users know and what they actually do. *Usability News* 8, 1 (Feb. 2006).
- [75] SCHNEIER, B. Myspace passwords aren't so dumb. <http://www.wired.com/politics/security/commentary/securitymatters/2006/12/72300>, 2006.
- [76] SCHREIER, J. Sony hacked again; 25 million entertainment users' info at risk. <http://www.wired.com/gamelifelife/2011/05/sony-online-entertainment-hack/>, May 2011.
- [77] SCHUMACHER, A. Making secure passwords. <http://blogs.creighton.edu/infosec/2011/08/10/making-secure-passwords/>, 2011.
- [78] SCHWARTZ, M. J. Sony hacked again, 1 million passwords exposed. <http://www.informationweek.com/security/attacks/sony-hacked-again-1-million-passwords-ex/229900111>, June 2011.
- [79] SELTMAN, H. *Experimental Design for Behavioral and Social Sciences*. Carnegie Mellon University, 2012.
- [80] SHANNON, C. E. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review* 5, 1 (1949).
- [81] SHANNON, C. E. Prediction and entropy of printed english. *Bell Systems Technical Journal* 30 (1951), 50–64.
- [82] SHAY, R., BAUER, L., CHRISTIN, N., CRANOR, L. F., FORGET, A., KOMANDURI, S., MAZUREK, M., MELICHER, W., SEGRETI, S. M., AND UR, B. A spoonful of sugar? The impact of guidance and feedback on password-creation behavior. In *CHI* (2015).
- [83] SHAY, R., AND BERTINO, E. A comprehensive simulation tool for the analysis of password policies. *International Journal of Information Security* 8, 4 (2009), 275–289.
- [84] SHAY, R., BHARGAV-SPANTZEL, A., AND BERTINO, E. Password policy simulation and analysis. In *ACM workshop on Digital Identity Management* (2007).
- [85] SHAY, R., ION, I., REEDER, R. W., AND CONSOLVO, S. “My religious aunt asked why I was trying to sell her viagra”: Experiences with account hijacking. In *CHI* (2014).

- [86] SHAY, R., KELLEY, P. G., KOMANDURI, S., MAZUREK, M. L., UR, B., VIDAS, T., BAUER, L., CHRISTIN, N., AND CRANOR, L. F. Correct horse battery staple: Exploring the usability of system-assigned passphrases. In *SOUPS* (2012).
- [87] SHAY, R., KOMANDURI, S., DURITY, A. L., HUH, P. S., MAZUREK, M. L., SEGRETI, S. M., UR, B., BAUER, L., CHRISTIN, N., AND CRANOR, L. F. Can long passwords be secure and usable? In *CHI* (2014).
- [88] SHAY, R., KOMANDURI, S., KELLEY, P. G., LEON, P. G., MAZUREK, M. L., BAUER, L., CHRISTIN, N., AND CRANOR, L. F. Encountering stronger password requirements: User attitudes and behaviors. In *SOUPS* (2010).
- [89] SINGH, S., CABRAAL, A., DEMOSTHENOUS, C., ASTBRINK, G., AND FURLONG, M. Password sharing: Implications for security design based on social practice. In *SIGCHI* (2007).
- [90] STANTON, J. M., STAM, K. R., MASTRANGELO, P., AND JOLTON, J. Analysis of end user security behaviors. *Comp. & Security* 24, 2 (2005), 124–133.
- [91] STOBERT, E., AND BIDDLE, R. The password life cycle: User behaviour in managing passwords. In *SOUPS* (2014).
- [92] STONE-GROSS, B., COVA, M., CAVALLARO, L., GILBERT, B., SZYDLOWSKI, M., KEMMERER, R., KRUEGEL, C., AND VIGNA, G. Your botnet is my botnet: Analysis of a botnet takeover. In *CCS* (2009).
- [93] TOOMIM, M., KRIPLEAN, T., PÖRTNER, C., AND LANDAY, J. Utility of human-computer interactions: Toward a science of preference measurement. In *CHI* (2011).
- [94] UR, B., KELLY, P. G., KOMANDURI, S., LEE, J., MAASS, M., MAZUREK, M., PAS-SARO, T., SHAY, R., VIDAS, T., BAUER, L., CHRISTIN, N., AND CRANOR, L. F. How does your password measure up? The effect of strength meters on password creation. In *USENIX Security* (2012).
- [95] VENKEN, S. LinkedIn leaked hashes password statistics. <http://pastebin.com/5pjggbMt>, June 2012.
- [96] VU, K.-P. L., PROCTOR, R. W., BHARGAV-SPANTZEL, A., TAI, B.-L. B., AND COOK, J. Improving password security and memorability to protect personal and organizational information. *International Journal of Human-Computer Studies* 65, 8 (2007), 744–757.
- [97] WEIR, M., AGGARWAL, S., COLLINS, M., AND STERN, H. Testing metrics for password creation policies by attacking large sets of revealed passwords. *CCS* (October 2010).
- [98] WEIR, M., AGGARWAL, S., DE MEDEIROS, B., AND GLODEK, B. Password cracking using probabilistic context-free grammars. In *IEEE Security and Privacy* (2009).
- [99] WILSON, S. Z. The protect IU blog—xkcd agrees: Use a passphrase. <http://protect.iu.edu/blog/2011/08/10/xkcd-agrees-use-passphrase>, 2011.

- [100] ZHANG, Y., MONROSE, F., AND REITER, M. K. The security of modern password expiration: An algorithmic framework and empirical analysis. In *CCS* (2010).
- [101] ZVIRAN, M., AND HAGA, W. J. Password security: An empirical study. *Journal of Management Information Systems* 15, 4 (1999), 161–185.

Appendix A

How Usable are System-Assigned Passphrases?

These are the surveys used in the study presented in Chapter 5.

A.1 Part One Survey

Learning my password was annoying.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Learning my password was difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Learning my password was fun.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Describe anything you did to help yourself remember your password.

Do you have a password or set of passwords you reuse in different places?

- Yes
- No
- I prefer not to answer

Do you have a password that you use for different accounts with a slight modification for each account?

- Yes
- No
- I prefer not to answer

Do you have an email password?

- Yes
- No

The questions on this page pertain to your real email password.

What is the domain for your primary email account (e.g. hotmail.com, gmail.com, cmu.edu)?

Thinking about the real password you use for your primary email account, how many of the following does it contain? Write "0" if there are none.

Uppercase letters:

Lowercase letters:

Numbers:

Symbols:

Approximately how long ago did you last change your real email password?

- Within the past month
- Within the past six months
- Within the past year
- More than a year ago

- More than 5 years ago
- Never
- I'm not sure
- I prefer not to answer

Does your main email provider require you to change your password periodically?

- Yes
- No
- I'm not sure
- I prefer not to answer

If my main email account assigned me a password like the one I used in this study, it would make my email account more secure.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

I would be annoyed if my main email account assigned me a password like the one I used in this study.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

If my main email account assigned me a password like the one I used in this study, it would be easier.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Are you willing to return and try to recall your password again in a few days?

- Yes
- No
- I prefer not to answer

If you have any additional feedback about passwords or this survey, please enter your comments here.

What is your gender?

- Female

- Male
- I prefer not to answer

How old are you?

Which of the following best describes your highest achieved education level?

- Some High School
- High School Graduate
- Some college, no degree
- Associates degree
- Bachelors degree
- Graduate degree (Masters, Doctorate, etc.)
- Other

Are you majoring in or do you have a degree or job in computer science, computer engineering, information technology, or a related field?

- Yes
- No
- I prefer not to answer

Are you majoring in or do you have a degree or job in art, architecture, design, photography, or a related field?

- Yes
- No
- I prefer not to answer

Are you majoring in or do you have a degree or job in math, physics, or engineering, or a related field?

- Yes
- No
- I prefer not to answer

What is your total household income?

- Less than \$10,000
- \$10,000 to \$19,999
- \$20,000 to \$29,999
- \$30,000 to \$39,999
- \$40,000 to \$49,999
- \$50,000 to \$59,999
- \$60,000 to \$69,999
- \$70,000 to \$79,999
- \$80,000 to \$89,999
- \$90,000 to \$99,999
- \$100,000 to \$149,999
- \$150,000 or more
- Prefer not to answer

Thank You!

A.2 Part Two Survey

Thank you for participating in this Carnegie Mellon University study. Please answer the following questions honestly. There are no right or wrong answers and everyone who finishes this task completely will receive his or her bonus payment.

How did you just enter your password for this study (please be honest – you get paid regardless, and this will help our research)?

- I typed it in from memory
- It was stored in my browser
- I cut and pasted it from a text file
- I looked it up in the place I had recorded it earlier and then I typed it in
- I use a password manager that filled it in for me
- I prefer not to answer
- Other:
- It was automatically filled in
- I forgot my password and followed the password reset link

Did you write down or store the password you created for this study (please be honest, you get paid regardless, this will help our research)?

- No
- Yes, on paper
- Yes, electronically (stored in computer, phone, etc.)
- Other
- I prefer not to answer

If you wrote down or stored your password for this study, how is it protected (choose all that apply)?

- I do not protect it
- I stored it in an encrypted file
- I hid it
- I stored it on a computer or device protected with another password
- I locked up the paper
- I always keep the password with me
- I wrote down a reminder instead of the actual password
- I keep the paper in an office or room that only I use

- I stored it on a computer or device that only I use
- Other
- I prefer not to answer
- I did not write down my password

Please describe how you store your password for this study, including what software you use or where you wrote it down.

What would you have done differently in protecting and remembering your password if this password were used for an account you would use outside this study?

- Did you imagine a scene related to the words or letters in your password to help you remember it?
- Yes
 - No

If so, describe the scene that you imagined.

- Did you think of a sentence or phrase based on the words or letters in your password to help you remember it?
- Yes
 - No

If so, describe the sentence or phrase that you used.

- Did you think of a story related to the words or letters in your password to help you remember it?
- Yes
 - No

If so, describe the story that you used.

What, if anything, about your new password makes it easy for you to remember?

- Do you have an email password?
- Yes
 - No

The questions on this page pertain to your real email password.

- When logging in with your real email password, do you refer to a written down or stored password?
- Yes

No

Prior to this survey, have you ever written down or stored your real email password?

- No
- Yes, on paper
- Yes, electronically (stored in computer, phone, etc.)
- Other
- I prefer not to answer

If you ever wrote down or stored your real email password, how was it protected (choose all that apply)?

- I did not write down or store it
- I did not protect it
- I stored it in an encrypted file
- I hid it
- I stored it on a computer or device protected with another password
- I locked up the paper
- I always kept the password with me
- I wrote down a reminder instead of the actual password
- I kept the paper in an office or room that only I use
- I stored it on a computer or device that only I use
- Other
- I prefer not to answer

To how many people have you given your real email password?

- 0
- 1
- 2-5
- 6-10
- More than 10

Consider the password you used for this study. If you were protecting/remembering a password for a real email account you would use outside the study, what would you have done differently?

- Nothing would have changed
- I would have written it down on paper
- I would not have written it down on paper
- I would have stored it electronically
- I would not have stored it electronically
- I would still write it on paper, but would secure the paper better
- I would have tried harder to remember it
- Other

How often do you type in your real email password (we are interested in when you type it in, not when your browser enters it automatically)?

- Never
- Several times per day
- Once per day
- Several times per week
- Once per week
- A few times per month
- Once per month
- Less than once a month
- I prefer not to answer

Remembering the password I use for my real email account is difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Remembering the password I used for this study was difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

If you have any additional feedback about passwords or this survey, please enter your comments here.

Appendix B

How Secure and Usable are Some Common Password Policies?

These are the surveys used in the study presented in Chapter 6. Questions were mandatory. Questions about real email, banking, and news websites were only shown to participants who indicated having an account of that type.

B.1 Part One Survey

Creating a password that meets the requirements given in this study was annoying.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Creating a password that meets the requirements given in this study was difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Creating a password that meets the requirements given in this study was fun.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

For this study, which of the following methods did you use to create your password (choose all that apply)?

- Based on a birthday
- Substituted symbols for some of the letters in a word or name (e.g. '@' instead of 'a')
- Based on an address
- Added symbols to the beginning or end of a word or name
- Removed letters from a word or name
- I prefer not to answer
- Based on the name of someone or something
- Added numbers to the beginning or end of a word or name
- Based on a phone number
- Based on a word in a language other than English
- Based on a word in English
- Password based on the first letter of each word in a phrase
- Substituted numbers for some of the letters in a word or name (e.g. '3' instead of 'e')
- Based on something else (please explain what):

Do you have a password or set of passwords you reuse in different places?

- Yes
- No
- I prefer not to answer

Do you have a password that you use for different accounts with a slight modification for each ac-

count?

- Yes
- No
- I prefer not to answer

Do you have an email password?

- Yes
- No

(If participant answers *Yes* then the following questions about the real email password are given)

The questions on this page pertain to your real email password.

If my main email account required me to change my password using the same requirements as used in this study, it would make my email account more secure.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

What is the domain for your primary email account (e.g. hotmail.com, gmail.com, cmu.edu)?

Thinking about the real password you use for your primary email account, how many of the following does it contain? Write "0" if there are none.

Uppercase letters
Lowercase letters
Numbers
Symbols

Approximately how long ago did you last change your real email password?

- Within the past month
- Within the past six months
- Within the past year
- More than a year ago
- More than 5 years ago
- Never
- I'm not sure
- I prefer not to answer

Does your main email provider require you to change your password periodically?

- Yes

- No
- I'm not sure
- I prefer not to answer

Do you have an online banking password?

- Yes
- No

(If yes, the above questions about real email passwords are repeated for real online banking passwords).

Do you have a password that you use to log on to newspaper/media websites to read the news (e.g. The Boston Globe, Times of India, ESPN)?

- Yes
- No

(If yes, the above questions about real email passwords are repeated for real news website passwords).

Are you willing to return and try to recall your password again in a few days?

- Yes
- No
- I prefer not to answer

If you have any additional feedback about passwords or this survey, please enter your comments here.

What is your gender?

- Female
- Male
- I prefer not to answer

How old are you?

Are you majoring in or do you have a degree or job in computer science, computer engineering, information technology, or a related field?

- Yes
- No
- I prefer not to answer

Thank you for taking our survey. Your response is very important to us.

B.2 Part Two Survey

Thank you for participating in this Carnegie Mellon University study. Please answer the following questions honestly. There are no right or wrong answers and everyone who finishes this task completely will receive his or her bonus payment.

How did you just enter your password for this study (please be honest, you get paid regardless, this will help our research)?

- I looked it up in the place I had recorded it earlier and then I typed it in
- I use a password manager that filled it in for me
- I forgot my password and followed the password reset link
- I typed it in from memory
- It was automatically filled in
- I cut and pasted it from a text file
- It was stored in my browser
- I prefer not to answer
- Other

When you created the password for this study, which of the following did you do?

- I created an entirely new password
- I prefer not to answer
- I reused a password I use for a different account
- I modified a password I use for a different account
- Other

Did you write down or store the password you created for this study? (please be honest, you get paid regardless, this will help our research)

- No
- Yes, on paper
- Yes, electronically (stored in computer, phone, etc.) I prefer not to answer
- Other

If you wrote down or stored your password for this study, how is it protected (choose all that apply)?

- I stored it on a computer or device protected with another password
- I locked up the paper
- I prefer not to answer
- I wrote down a reminder instead of the actual password
- I keep the paper in an office or room that only I use

- I hid it
- I stored it on a computer or device that only I use
- I do not protect it
- I stored it in an encrypted file
- I did not write down my password
- I always keep the password with me
- Other

Please describe how you store your password for this study, including what software you use or where you wrote it down.

What would you have done differently in creating, protecting, and remembering your password if this password were used for an account you would use outside this study?

Do you use the password you created for this study for any other account?

- Yes
- No
- I prefer not to answer

What about your new password makes it easy for you to remember?

Do you have an email password?

- Yes
- No

(If participant answers *Yes* then the following questions about the real email password are given)

The questions on this page pertain to your real email password.

When logging in with your real email password, do you refer to a written down or stored password?

- Yes
- No

Prior to this survey, have you ever written down or stored your real email password?

- No
- Yes, on paper
- Yes, electronically (stored in computer, phone, etc.) I prefer not to answer
- I prefer not to answer
- Other

If you ever wrote down or stored your real email password, how was it protected (choose all that apply)?

- I stored it on a computer or device protected with another password
- I locked up the paper
- I prefer not to answer
- I wrote down a reminder instead of the actual password
- I keep the paper in an office or room that only I use
- I hid it
- I stored it on a computer or device that only I use
- I do not protect it
- I stored it in an encrypted file
- I did not write down my password
- I always keep the password with me
- Other

To how many people have you given your real email password?

Consider the password you created for this study. If you were creating a password for a real email account you would use outside the study, what would you have done differently?

- I would have used an easier-to-remember password
- I would have created a longer password
- I would have used an easier-to-type password
- I would have used more symbols, upper case letters, or numbers
- I would have reused a password, but did not for this study
- Nothing would have changed
- Other

Consider the password you created for this study. If you were protecting/remembering a password for a real email account you would use outside the study, what would you have done differently?

- Nothing would have changed
- I would have written it down on paper
- I would not have written it down on paper
- I would have stored it electronically
- I would not have stored it electronically
- I would still write it on paper, but would secure the paper better
- I would have tried harder to remember it

Other

How often do you type in your real email password (we are interested in when you type it in, not when your browser enters it automatically)?

- Never
- Several times per day
- Once per day
- Several times per week
- Once per week
- A few times per month
- Once per month
- Less than once a month
- I prefer not to answer

Remembering the password I use for my real email account is difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Do you have an online banking password?

- Yes
- No

(If yes, the above questions about real email passwords are repeated for real online banking passwords).

Do you have a password that you use to log on to newspaper/media websites to read the news?

- Yes
- No

(If yes, the above questions about real email passwords are repeated for real news website passwords).

Remembering the password I used for this study was difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

If you have any additional feedback about passwords or this survey, please enter your comments here.

Appendix C

Can Longer Passwords be Secure and Usable?

These are the surveys used in the study presented in Chapter 7.

C.1 Part One Survey

Thank you for participating in this Carnegie Mellon University study. Please answer the following questions honestly. There are no right or wrong answers and everyone who finishes this task completely will receive his or her payment.

Creating a password that meets the requirements given in this study was annoying.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Creating a password that meets the requirements given in this study was difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Creating a password that meets the requirements given in this study was fun.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

For this study, did you base your password on any of the following? You may choose more than one.

- Based on another password
- Based on an address
- Based on more than one word in English
- Based on the first letter of each word in a phrase
- Based on the name of someone or something
- Based on one word in English
- Based on a phone number
- Based on one or more words in a language other than English
- Based on a birthday
- I did not base my password on any of these
- I prefer not to answer

If you created your password based on a word or name, did you modify that word or name in any of the following ways to create your password? You may choose more than one.

- Added symbols to the beginning or end
- Removed letters
- Substituted symbols for some of the letters (e.g., '@' instead of 'a')
- Added numbers to the beginning or end
- Substituted numbers for some of the letters (e.g., '3' instead of 'e')
- My password is based on a word or name, but I did not do any of these
- The password is not based on a word or name
- I prefer not to answer

The questions on this page are about your **real email password**. Please think about that password for these questions.

How would you describe your primary email account?

- A work-based email account
- Another type of email account
- A university email account (e.g., name@cmu.edu)
- A web email account (e.g., Gmail, Hotmail, Yahoo! Mail)
- I don't know
- I prefer not to answer

I would prefer if my main email provider had the same password requirements as used in this study.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

If my main email provider had the same password requirements as used in this study, my email account would be more secure.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

It was more difficult to create my password for this study than it was to create my real email password.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

How many characters (letters, digits, and symbols) are in your real email password?

- 0-6
- 7-9
- 10-12
- 13-15
- 16-18
- 19+
- I don't know

- I prefer not to answer

Approximately how long ago did you last create or change your real email password?

- Within the past month
- Within the past six months
- Within the past year
- More than a year ago
- More than 5 years ago
- Never
- I'm not sure
- I prefer not to answer

What is your gender?

- Female
- Male
- I prefer not to answer

How old are you?

Are you majoring in or do you have a degree or job in computer science, computer engineering, information technology, or a related field?

- Yes
- No
- I prefer not to answer

What is the highest level of education that you have completed?

- High School Degree
- Associate Degree
- Bachelor's Degree
- Master's Degree
- Doctoral Degree
- I prefer not to answer
- Other

Is English your native language? This will not affect your eligibility for this study.

- Yes
- No
- I prefer not to answer

If you have any additional feedback about passwords or this survey, please enter your comments here.

C.2 Part Two Survey

Thank you for participating in this Carnegie Mellon University study. Please answer the following questions honestly. There are no right or wrong answers and everyone who finishes this task completely will receive his or her bonus payment.

How did you just enter your password for this study? Please be honest. You will be paid regardless and this will help our research.

- My browser automatically filled it in
- I typed it in from memory
- I looked it up and typed it in
- I copied and pasted it
- I used the "I forgot my password" link
- I used a password manager
- I prefer not to answer
- Other

Did you write down or store the password you created for this study? You may choose more than one. Please be honest. You will be paid regardless and this will help our research.

- I did not write down or store my password
- I wrote down my password on paper
- I stored my password on the computer
- I stored my password on my phone or another electronic device
- My password manager remembered my password
- My browser remembered my password
- I prefer not to answer
- Other

On what sort of computer or device have you just entered your password?

- Tablet
- Desktop computer
- Smartphone
- Laptop computer
- I prefer not to answer
- Other

Remembering the password I used for this study was difficult.

- Strongly agree
- Agree
- Neutral
- Disagree

- Strongly disagree

The questions on this page are about your **real email password**. Please think about that password for these questions.

When logging in with your **real email password**, how do you most often enter that password?

- I look it up and type it in
- I copy and paste it
- My browser automatically fills it in
- I type it in from memory
- I use a password manager
- I prefer not to answer
- Other

Have you ever written down or stored your **real email password**? You may choose more than one.

- I did not write down or store my password
- I wrote down my password on paper
- I stored my password on the computer
- I stored my password on my phone or another electronic device
- My password manager remembered my password
- My browser remembered my password
- I prefer not to answer
- Other

On what sort of computer or device do you most often enter your real email password?

- Tablet
- Smartphone
- Laptop computer
- Desktop computer
- I prefer not to answer
- Other

To how many people have you given your **real email password**?

How often do you type in your **real email password**? We are interested in when you type it in, not when your browser enters it automatically or when you paste it.

- Several times per day
- Once per day
- Several times per week
- Once per week
- A few times per month

- Once per month
- Less than once per month
- Never
- I prefer not to answer

Remembering the password I use for my **real email** account is difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Have you ever used the password that you created for this study for any other account?

- This is a password that I often use
- I have used the same password before, but not often
- I have used a very similar password before, but not often
- I have never used this password before
- This is very similar to a password that I often use
- I prefer not to answer

What, if anything, about the password you created for this study makes it easy for you to remember?

What, if anything, would you have done differently in creating, protecting, and remembering your password for this study if this same password were used for your real email account?

Consider the password you created for this study. If you were creating a password for your real email account under the same password-creation rules as used in this study, what would you have done differently? You may choose more than one.

- Nothing would have changed
- I would have used more symbols
- I would have used more uppercase letters
- I would have reused a password, but did not reuse a password for this study
- I would have used an easier-to-type password
- I would have used an easier-to-remember password
- I would have used a longer password
- I would have used more digits
- Other

If you have any additional feedback about passwords or this survey, please enter your comments here.

Appendix D

Can Creation-Time Feedback Help Users Create Passwords?

These are the surveys used in the study presented in Chapter 8.

D.1 Part One Survey

Thank you for participating in this Carnegie Mellon University study. Please answer the following questions honestly. There are no right or wrong answers and everyone who finishes this task completely will receive his or her payment.

Creating a password that meets the requirements given in this study was annoying.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Creating a password that meets the requirements given in this study was difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Creating a password that meets the requirements given in this study was fun.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

For this study, did you base your password on any of the following? You may choose more than one.

- Based on a keyboard pattern
- Based on a birthday
- Based on a phone number
- Based on the name of someone or something
- Based on one other password
- Based on more than one word
- Based on the first letter of each word in a phrase
- Based on more than one other password
- Based on a single word
- I did not base my password on any of these
- I prefer not to answer
- Other

The feedback and instructions I saw while creating my password led me to create a stronger password than I would have otherwise.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

The feedback and instructions I saw while creating my password were difficult to follow.

- Strongly agree
- Agree
- Neutral
- Disagree

Strongly disagree

I enjoyed seeing the feedback and instructions while I created my password.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

If you have any additional thoughts about the feedback and instructions you saw while you created your password, please enter your comments here.

The questions on this page are about your **real email password**. Please think about that password for these questions.

How would you describe your primary email account?

- A work-based email account
- Another type of email account
- A university email account (e.g., name@cmu.edu)
- A web email account (e.g., Gmail, Hotmail, Yahoo! Mail)
- I don't know
- I prefer not to answer

I would prefer if my main email provider had the same password requirements as used in this study.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

If my main email provider had the same password requirements as used in this study, my email account would be more secure.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

It was more difficult to create my password for this study than it was to create my real email password.

- Strongly agree
- Agree

- Neutral
- Disagree
- Strongly disagree

What is your gender?

- Female
- Male
- I prefer not to answer

How old are you?

Are you majoring in or do you have a degree or job in computer science, computer engineering, information technology, or a related field?

- Yes
- No
- I prefer not to answer

What is the highest level of education that you have completed?

- High School Degree
- Associate Degree
- Bachelor's Degree
- Master's Degree
- Doctoral Degree
- I prefer not to answer
- Other

If you have any additional feedback about passwords or this survey, please enter your comments here.

D.2 Part Two Survey

Thank you for participating in this Carnegie Mellon University study. Please answer the following questions honestly. There are no right or wrong answers and everyone who finishes this task completely will receive his or her bonus payment.

How did you just enter your password for this study? Please be honest. You will be paid regardless and this will help our research.

- My browser automatically filled it in
- I typed it in from memory
- I looked it up and typed it in
- I copied and pasted it
- I used the "I forgot my password" link
- I used a password manager

- I looked up a hint and typed in the password
- I prefer not to answer
- Other

Did you write down or store the password you created for this study? You may choose more than one. Please be honest. You will be paid regardless and this will help our research.

- I did not write down or store my password
- I wrote down my password on paper
- I stored my password on the computer
- I stored my password on my phone or another electronic device
- My password manager remembered my password
- My browser remembered my password
- I only stored a hint about the password, not the password itself
- I prefer not to answer
- Other

On what sort of computer or device have you just entered your password?

- Tablet
- Desktop computer
- Smartphone
- Laptop computer
- I prefer not to answer
- Other

Remembering the password I used for this study was difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

The questions on this page are about your **real email password**. Please think about that password for these questions.

When logging in with your **real email password**, how do you most often enter that password?

- I look it up and type it in
- I copy and paste it
- My browser automatically fills it in
- I type it in from memory
- I use a password manager
- I prefer not to answer

- Other

On what sort of computer or device do you most often enter your real email password?

- Tablet
- Smartphone
- Laptop computer
- Desktop computer
- I prefer not to answer
- Other

How often do you type in your **real email password**? We are interested in when you type it in, not when your browser enters it automatically or when you paste it.

- Several times per day
- Once per day
- Several times per week
- Once per week
- A few times per month
- Once per month
- Less than once per month
- Never
- I prefer not to answer

Remembering the password I use for my **real email** account is difficult.

- Strongly agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Have you ever used the password that you created for this study for any other account?

- This is a password that I often use
- I have used the same password before, but not often
- I have used a very similar password before, but not often
- I have never used this password before
- This is very similar to a password that I often use
- I prefer not to answer

What, if anything, about the password you created for this study makes it easy for you to remember?

What, if anything, would you have done differently in creating, protecting, and remembering

your password for this study if this same password were used for your real email account?

Consider the password you created for this study. If you were creating a password for your real email account under the same password-creation rules as used in this study, what would you have done differently? You may choose more than one.

- I would have used more symbols
- I would have used an easier-to-type password
- I would not have reused a password, but did reuse a password for this study
- I would have used more digits
- I would have used a shorter password
- I would have reused a password, but did not

reuse a password for this study

- I would have used a longer password
- I would have used fewer symbols, digits, or uppercase letters
- Nothing would have changed
- I would have used more uppercase letters
- I would have used an easier-to-remember password
- Other

If you have any additional feedback about passwords or this survey, please enter your comments here.