A Spoonful of Sugar? The Impact of Guidance and Feedback on Password-Creation Behavior

Richard Shay, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, Alain Forget, Saranga Komanduri, Michelle L. Mazurek*, William Melicher, Sean M. Segreti, Blase Ur

Carnegie Mellon University
Pittsburgh, PA
{rshay, lbauer, nicolasc, lorrie, aforget, sarangak, billy, ssegreti, bur}@cmu.edu

*University of Maryland
College Park, MD
mmazurek@cs.umd.edu

ABSTRACT

Users often struggle to create passwords under strict requirements. To make this process easier, some providers present real-time feedback during password creation, indicating which requirements are not yet met. Other providers guide users through a multi-step password-creation process. Our 6,435-participant online study examines how feedback and guidance affect password security and usability. We find that real-time password-creation feedback can help users create strong passwords with fewer errors. We also find that although guiding participants through a three-step password-creation process can make creation easier, it may result in weaker passwords. Our results suggest that service providers should present password requirements with feedback to increase usability. However, the presentation of feedback and guidance must be carefully considered, since identical requirements can have different security and usability effects depending on presentation.

Author Keywords

Passwords; Password-composition policies; Security policy; Usable security; Authentication

ACM Classification Keywords

D.4.6 Management Of Computing and Information Systems: Security and Protection—*Authentication*

INTRODUCTION

Service providers ask for complex passwords. For example, Yahoo requires passwords with at least eight characters and multiple character classes.[1] Users often have difficulty meeting complex password requirements and resort to predictable and insecure choices, such as appending digits or symbols, or including common substrings or keyboard patterns [15, 16, 19]. Some service providers now offer *requirements feedback*: real-time password-creation feedback telling users whether they have met the requirements, such as green check marks indicating satisfied requirements (Figure 1). Other service providers offer *guidance*: walking users through a multi-step password-creation process (Figure 2).

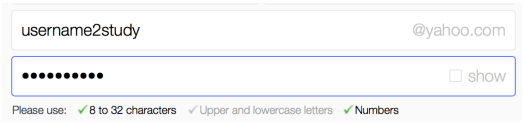

Figure 1: Yahoo's real-time password-creation feedback.

In this paper, we present the first analysis of the usability of requirements feedback and guidance mechanisms. Prior work has considered the security and usability effects of complex password-composition requirements [9] and examined feedback specifically in the context of password meters that provide estimates of password strength [3, 17]. We studied techniques 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 policy, we examined two approaches to multi-step password-creation. In *guidance*, we guided participants to enhance a simple password by adding components until it met all requirements. In *insertion*, inspired by Forget et al. [5], participants created a simple password and the system inserted random characters. 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 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 more palatable to users.

Compared to having participants create a password under a strict policy in a single step, both the guidance and inser-

[1] `https://edit.yahoo.com/registration`

CHI 2015, Apr 18-23 2015, Seoul, Republic of Korea
ACM 978-1-4503-3145-6/15/04.
`http://dx.doi.org/10.1145/2702123.2702586`

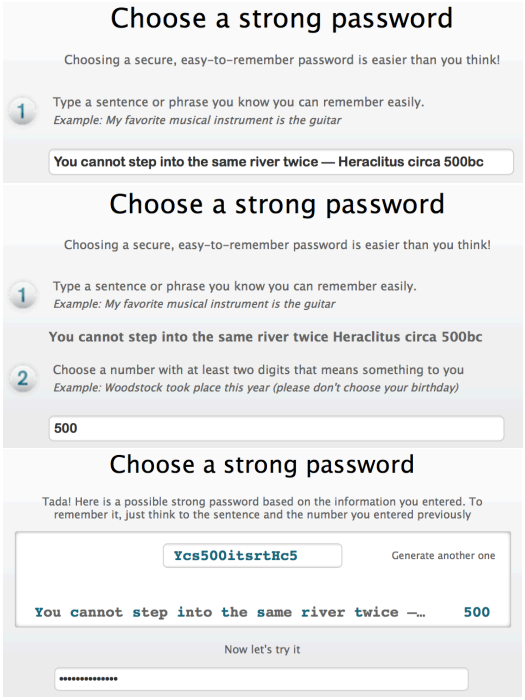

Figure 2: Dashlane's multi-step password-creation process.

tion 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 a few characters into passwords leaves them vulnerable to having those characters brute-forced. Overall, our results demonstrate that looking only at password requirements is insufficient. The presentation of those requirements can affect usability and security.

Next, we present related work and then our research questions and methodology. Then, we discuss our results and their practical implications.

RELATED WORK

Password-composition policies often include requirements on character length and content. Furnell found that the password restrictions of 10 popular websites were highly variable and sometimes conflicted [6, 7]. Recent studies compared the usability and security of passwords adhering to various policies. Many such studies collect passwords from the crowdsourcing service Amazon Mechanical Turk (MTurk). In one of our previous studies, we compared the security and usability of various password policies that required more characters than is typical in practice. We found that some policies requiring longer passwords led to more usable and secure passwords than a more typical policy that required fewer characters but had other more complex requirements. However, security suffered when participants placed required uppercase letters, digits, and symbols at the start and end of their passwords, and included common substrings and keyboard patterns [15]. Our current research builds on this work by examining mechanisms intended to make password requirements that prevent such predictable behavior easier for users to handle.

Several studies explored how to encourage users to choose more secure passwords. Furnell and Bär found that users chose more secure passwords when presented with text advice or a password-strength meter [8]. We found that more stringent-scoring meters effectively led to stronger passwords, but at the cost of usability [17]. Egelman et al. found that the mere presence of a password meter led to more secure passwords, but only for valued accounts [3]. Rather than providing feedback in the form of a password-strength meter, our research evaluates the impact of providing real-time feedback on compliance with requirements.

Forget et al. explored persuading users to create more secure passwords by adding random characters to user-created passwords. Users could *shuffle* to have a different set of random characters placed at different locations. The authors found that inserting two characters increased password security without perceptibly affecting usability. 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 [5]. Our research builds on this with a condition that also inserts random characters into participants' passwords. We use a much larger sample size, and test recall after a few days as well as a few minutes.

Moshfeghian and Ryu found that many popular websites poorly present password restrictions. The authors suggest best practices for password-creation design. Their advice includes placing password guidance and feedback in clear and concise language in close proximity to password fields, providing real-time feedback to reduce failed attempts, using red-colored feedback for unfulfilled requirements, and using globally recognizable visual elements (e.g., green check marks) to communicate feedback [12]. Our study's presentation of password requirements follows these best practices and provides empirical evidence of their effectiveness.

RESEARCH QUESTIONS

Our four research questions examine how strict password requirements can be made more usable through visual elements, requirements feedback, guidance, and automatic insertion of random characters. Our *base* set of strict requirements requires that passwords contain at least 12 characters and three character classes (three of either lowercase letters, uppercase letters, digits, or symbols).

Question Q_1: *How do the blacklist and pattern requirements affect password security and usability when applied to strict requirements?* The *blacklist* requirement prohibits passwords from containing any substring in a 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 [15]. Based on previous findings [9, 15], 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 create or recall passwords with strict 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 Carnegie Mellon University, including its wordmark and red-and-white color scheme, while the base-plain condition used default black-and-white HTML rendering without a logo or wordmark. We wondered whether the distinct visuals of the former may help participants recall their passwords, perhaps by acting as a memory cue. This has real-world implications because users often create and subsequently recall passwords on websites with distinct branding, and select predictable passwords based on visual cues [2].

Question Q_3: *How does requirements feedback affect password creation, recall, and strength when applied to strict 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.

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).

METHODOLOGY

In this section, we discuss our data collection and analysis procedures, which were approved by our institutional review board. Our experimental protocol is based on a protocol used successfully to contrast password-composition policies in prior research [9, 15].

We conducted an online between-subjects study on Amazon's Mechanical Turk crowdsourcing service (MTurk) in two parts. Participants were informed that this would be a two-part study with additional compensation for completing both parts. We paid participants 55 cents for completing *Part One* and 70 cents for completing *Part Two*. In Part One, we asked participants to imagine they were creating a new password for their email account. The requirements and feedback varied by condition. Participants then completed a short survey before we asked them to recall their password. In Part Two, we emailed participants two days later and invited them to recall their password and complete a second survey.

Except when looking at dropout rates, our analysis includes only participants who completed Part One. For metrics specific to Part Two, we consider only participants who returned and completed Part Two within three days of being sent the follow-up invitation. Thus, all Part Two participants had created their passwords two to five days earlier. Our usability metrics included self-reported sentiment and how many attempts participants needed to create and recall their password. We also measured password strength using a state-of-the-art password cracker, as described below.

Condition	Pattern	Blacklist	Feedback	Brand	Three-step
Base				University	
Base$_{rt}$			✓	University	
Blacklist		✓		University	
Blacklist$_{rt}$		✓	✓	University	
Base-plain				Plain	
Pattern	✓			University	
Pattern$_{rt}$	✓		✓	University	
Guide	✓		✓	University	Participant adds
Insert	✓		✓	University	System adds

Table 1: Requirements of each condition. All require at least 12 characters and three character classes. Pattern requires starting and ending with lowercase letters. The blacklist requirement prohibits certain substrings in passwords. Three-step uses a multiple-step process to create the password.

Conditions

Conditions, listed in Table 1, were assigned round-robin. They share the *base* requirements – that passwords have at least 12 characters and three character classes. Prior work suggests these are usable yet strong requirements [15].

Password Creation in One Step

Base used the base requirements. ***Pattern*** required passwords to begin and end with lowercase letters, as shown in Figure 3. ***Blacklist*** required that passwords not contain substrings from a 41,329-string blacklist. This blacklist was based on common substrings in cracked passwords in a prior study [15] and substrings we thought would be easily guessed. The blacklist prohibited:

- 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 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. Figure 4 shows *Pattern*$_{rt}$. To keep these conditions similar to their analogous conditions without requirements feedback, participants were permitted to submit non-compliant passwords (and then were told that they would have to try again to create a compliant password).

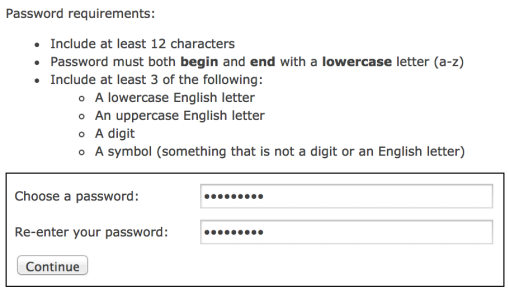

Figure 3: Password-requirements presentation for *Pattern* as "password!" is entered.

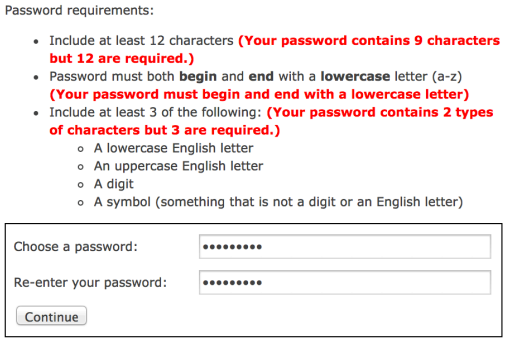

Figure 4: Password-requirements presentation for *Pattern*$_{rt}$ as "password!" is entered.

All conditions except ***Base-plain*** were 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 black-and-white HTML rendering, in order to address Q_2.

Password Creation in Three Steps

Two conditions – ***Guide*** and ***Insert*** – employed a three-step password-creation process. Both used requirements feedback and led to passwords meeting the same requirements as *Pattern*$_{rt}$. Figures 5 and 6 show the first two steps of *Guide* and *Insert*. The main difference between them is that *Insert* has a step in which the system randomly inserts characters into the password, while *Guide* instructs participants to insert additional characters.

In *Step 1*, we asked participants, "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 the password until meeting the requirements. In the second step for *Insert*, participants were shown their system-enhanced password, with the two newly added characters in green text. Below, participants were shown a text box with their non-enhanced password in plaintext, and were asked to modify it to match what was above. Finally, in *Step 3*, we asked participants to confirm their passwords in a blank password text field.

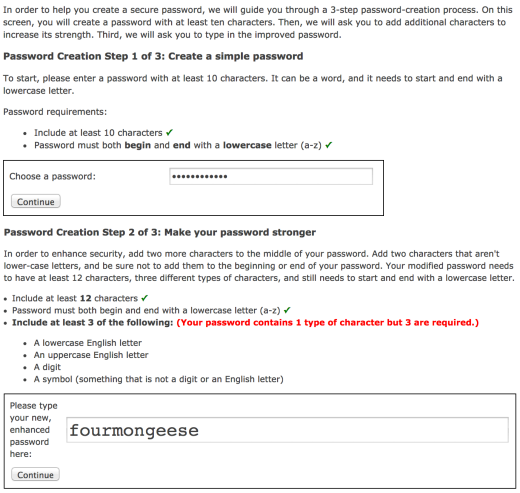

Figure 5: The first two steps of password creation in *Guide*.

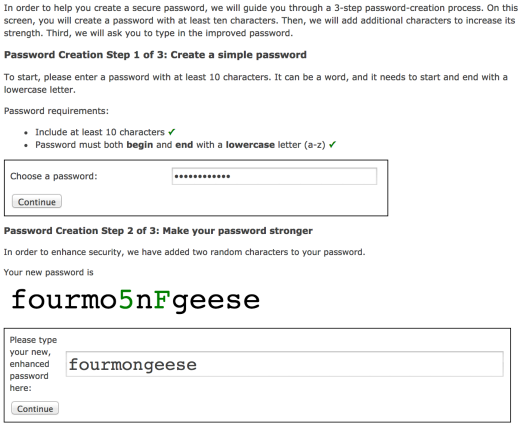

Figure 6: The first two steps of password creation in *Insert*.

To insert two random characters in *Insert*, we first selected two characters of two different character classes, chosen from nine symbols @!$ $*$ #. $-$ &_, eight digits, and 24 uppercase letters (removing *O*, *0*, *I*, and *1*, 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.

Statistical Testing

We perform omnibus tests to detect whether our conditions differ from one another. When they do, we perform a set of corrected pairwise tests. Rather than testing each pair of conditions, we test selected pairs to help answer our research questions. The pairwise comparisons 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*

We use a significance level $\alpha = .05$. Quantitative tests use Kruskal-Wallis (KW) for omnibus testing and Holm-Bonferroni-corrected (HC) Mann-Whitney U (MW) for pairwise testing. For their effect size, we used the "common language effect size" [10]. This is the probability that a random participant from one condition has a more favorable value than a random participant from the other condition. If they have the same value, a coin flip determines the winner. Categorical comparisons use the Chi Square test for omnibus comparisons and Holm-Bonferroni-corrected Fisher's Exact Test (FET) for pairwise comparisons. For effect size, we calculate the odds ratio for each significant pairwise comparison.

Measuring Password Strength

To measure password strength, we calculated *guess numbers* [9] for each password. A guess number is an estimate of the number of guesses that an attacker would require to guess successfully a participant's study password. This simulates an offline attack, in which an attacker has stolen a hashed password file and can make a number of guesses bounded only by computation speed and hardware availability. A sophisticated attacker will start by guessing the most likely passwords and proceed in order of decreasing probability. To generate guess numbers, we used the probabilistic context-free grammar (PCFG) password-cracking technique introduced by Weir et al. [20] and updated in subsequent work [9, 15]. The PCFG algorithm trains on structures and character strings from existing password corpora, creating a lookup table up to a certain given number of guesses. We trained the cracker on public data including words from the Google corpus [1], the Openwall cracking dictionary,[2] and the leaked MySpace [13] and RockYou [18] datasets. We also trained our strings on passwords collected during previous passwords-research 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 made guesses starting and ending with a lowercase letter. We divided each group of passwords into two subsets and used one as training data to crack the other (two *folds*). This simulated an attacker with access to passwords created under similar requirements. We generated at least 2×10^{13} guesses per condition.

Insert 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. The traditional PCFG approach was ineffective against *Insert*, cracking only one password after 10^{12} guesses. We then used a better-targeted approach for our security results. We used the PCFG algorithm to crack the pre-enhancement passwords, and then multiplied the number of guesses by 43,200. This simulated an attacker who knows the random-character insertion algorithm and brute-forces through all possible permutations.

Limitations

Limitations of our method of studying password-composition policies have been discussed in prior papers employing a similar approach [9,15]. As with any MTurk study, there is a concern about external validity, since participants created passwords for use only within the context of the study, and thus the passwords held little value to participants. However, two recent studies suggest that experimental studies may be a reasonable way to examine actual password behavior. Mazurek et al. contrasted characteristics of MTurk studies with genuine university passwords. They found that study passwords were very similar to university passwords across a number of metrics, suggesting that MTurk passwords were viable for learning about actual user behavior [11]. Fahl et al. also compared genuine university passwords with study passwords and found that the password sets resembled each other [4].

Our methodology is also temporally limited. We looked at password recall after a few minutes and after a few days. Actual password usage can involve using a password long after its creation, which we did not examine. We did not test every possible permutation of password requirements, guidance, and feedback. It is possible that other feedback formats would interact differently with other sets of requirements. In addition, we only tested one example of guidance and one of insertion. Our branding only included plain branding and that of a university. It is possible that some more dramatic branding scheme would have made a bigger difference.

RESULTS

This section presents our findings. We discuss our participants, followed by password strength, and then usability results for password creation and recall. Table 2 summarizes results for password creation, Table 3 summarizes password characteristics, and Table 4 presents other study metrics.

Participants

We collected data in May and June 2014; 7,262 participants began our study and 6,435 finished Part One. We invited those who finished Part One to return for Part Two. 3,934 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 participants disclosed their gender; 46.7% were male and 52.5% female. 95.7% stated their highest degree; 45.9% held a Bachelor's degree or higher.

Participants in *Pattern*$_{rt}$ (84.7%) were less likely to finish Part One 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 (χ^2_8=2.77, p=0.948). 95.3% of participants who started Part Two within five days finished, and this too did not vary by condition (χ^2_8=4.64, p=0.795).

[2] `openwall.org`

	Creation errors			Creation time			Creation difficult			Creation annoying		
condition pairs	mean	p-val	effect	med (sec)	p-val	effect	% agree	p-val	effect	% agree	p-val	effect
Q1: Impact of blacklist and pattern requirements												
Base – Blacklist	0.7 – 0.8	.152	–	**61** – 88	<.001	0.665	**24** – 32	.002	1.536	55 – 61	.212	–
Base – Pattern	**0.7** – 1.5	<.001	0.651	**61** – 122	<.001	0.802	**24** – 50	<.001	3.17	**55** – 77	<.001	2.643
Blacklist – Pattern	**0.8** – 1.5	<.001	0.628	**88** – 122	<.001	0.655	**32** – 50	<.001	2.065	**61** – 77	<.001	2.115
Q2: Impact of branded look-and-feel												
Base – Base-plain	0.7 – **0.6**	.039	0.534	61 – 60	1	–	24 – 23	1	–	55 – 57	1	–
Q3: Impact of password-creation feedback												
Base – Base$_{rt}$	0.7 – **0.3**	<.001	0.635	61 – 60	1	–	24 – 22	1	–	55 – 55	1	–
Blacklist – Blacklist$_{rt}$	0.8 – **0.4**	<.001	0.632	88 – 86	.621	–	32 – 29	.618	–	61 – 56	.281	–
Pattern – Pattern$_{rt}$	1.5 – **0.6**	<.001	0.701	122 – **109**	.019	0.546	50 – 49	1	–	77 – 77	1	–
Q4: Impact of guiding and insertion through creation												
Guide – Pattern$_{rt}$	0.8 – 0.6	.114	–	116 – **109**	.036	0.54	**29** – 49	<.001	2.35	**67** – 77	<.001	1.622
Insert – Pattern$_{rt}$	0.7 – **0.6**	.023	0.537	**101** – 109	.019	0.545	**17** – 49	<.001	4.684	**45** – 77	<.001	3.912

Table 2: Password-creation metrics and comparisons. For each metric, we provide an average value, the result of a significance test, and the effect size (as described in the Methodology). For significant comparisons, the significantly better value is in bold.

	Exceed min class			Password length			Cracked after 2×10^{13} guesses			Storage		
condition pairs	%	p-val	effect	mean (char)	p-val	effect	%	p-val	effect	%	p-val	effect
Q1: Impact of blacklist and pattern requirements												
Base – Blacklist	71 – 67	.176	–	13.6 – 13.8	.285	–	28 – **21**	.012	1.474	49 – 57	.091	–
Base – Pattern	**71** – 57	<.001	1.844	13.6 – **14.1**	<.001	0.58	28 – **8**	<.001	4.323	**49** – 69	<.001	2.308
Blacklist – Pattern	**67** – 57	<.001	1.508	13.8 – **14.1**	.003	0.553	21 – **8**	<.001	2.933	**57** – 69	.003	1.649
Q2: Impact of branded look-and-feel												
Base – Base-plain	71 – 73	.449	–	13.6 – 13.5	1	–	28 – 30	.835	–	49 – 52	1	–
Q3: Impact of password-creation feedback												
Base – Base$_{rt}$	**71** – 53	<.001	2.174	13.6 – 13.6	1	–	28 – 30	.835	–	49 – 50	1	–
Blacklist – Blacklist$_{rt}$	**67** – 51	<.001	1.898	13.8 – 13.6	.285	–	21 – 24	.525	–	57 – 55	1	–
Pattern – Pattern$_{rt}$	**57** – 39	<.001	2.06	14.1 – 14.2	1	–	8 – 10	.525	–	69 – 61	.154	–
Q4: Impact of guiding and insertion through creation												
Guide – Pattern$_{rt}$	22 – **39**	<.001	2.267	13.8 – **14.2**	.008	0.547	16 – **10**	.02	1.582	62 – 61	1	–
Insert – Pattern$_{rt}$	15 – **39**	<.001	3.684	13.7 – **14.2**	<.001	0.562	29 – **10**	<.001	3.416	76 – **61**	<.001	1.962

Table 3: Password characteristics and comparisons. *Exceed min class* is exceeding the minimum number of character classes.

Password Strength

We observe an overall cracking rate of 21.7% after 2×10^{13} guesses, varying significantly by condition, as shown in Table 3. The percentage of passwords cracked after each guess is shown in Figure 7. As observed in Table 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 (we did not continue to 10^{13}), over 5% of each other condition had been cracked, compared to 0.1% of *Insert*. 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.

We used nine symbols, chosen to be distinct and easy to recognize. While increasing the number of symbols would have increased security, it would not have been substantial. Using 19 symbols rather than nine would have resulted in 72,000 insertion configurations instead of 43,200. Using 32 symbols would have led to 109,440 configurations. Using the same cracking technique, using 32 symbols would lead to a cracking rate of 26.1%, compared to the current 28.6%.

Exceeding Minimum Requirements

In addition to looking at results of the PCFG cracker, we looked at the structural components of passwords. While all passwords required 12 characters and three characters classes, participants were free to exceed either requirement.

Table 3 shows percentages of participants who exceeded the minimum three-character-class requirement. *Pattern$_{rt}$* par-

	Finished Part One			Part One recall time			Part two recall attempts			Part two recall difficult		
condition pairs	%	p-val	effect	med (sec)	p-val	effect	mean	p-val	effect	% agree	p-val	effect
Q1: Impact of blacklist and pattern requirements												
Base – Blacklist	88 – 88	1	–	10 – 10	1	–	1.9 – 1.9	1	–	36 – 43	.368	–
Base – Pattern	88 – 84	.131	–	**10** – 11	<.001	0.568	1.9 – 1.7	.244	–	**36** – 54	<.001	2.087
Blacklist – Pattern	88 – 84	.131	–	**10** – 11	.004	0.554	1.9 – 1.7	.83	–	**43** – 54	.004	1.616
Q2: Impact of branded look-and-feel												
Base – Base-plain	88 – 91	.384	–	10 – 9	1	–	1.9 – 1.9	1	–	36 – 32	.944	–
Q3: Impact of password-creation feedback												
Base – Base$_{rt}$	88 – 89	1	–	10 – 10	1	–	1.9 – 1.9	1	–	36 – 35	.983	–
Blacklist – Blacklist$_{rt}$	88 – 89	1	–	10 – 10	1	–	1.9 – 2.0	1	–	43 – 40	.983	–
Pattern – Pattern$_{rt}$	84 – 85	1	–	11 – 11	1	–	1.7 – 1.7	1	–	54 – 46	.094	–
Q4: Impact of guiding and insertion through creation												
Guide – Pattern$_{rt}$	**91** – 85	.001	1.781	10 – 11	.181	–	2.0 – 1.7	.088	–	42 – 46	.944	–
Insert – Pattern$_{rt}$	**94** – 85	<.001	2.718	12 – 11	1	–	2.0 – 1.7	.146	–	53 – 46	.26	–

Table 4: User-behavior metrics and comparisons.

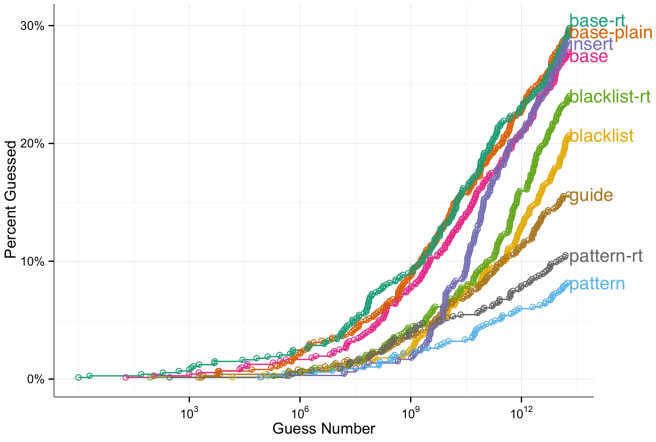

Figure 7: The percentage of passwords cracked in each condition by the number of guesses made in log scale. Our cutoff for guess numbers is 2×10^{13}. Table 3 shows comparisons.

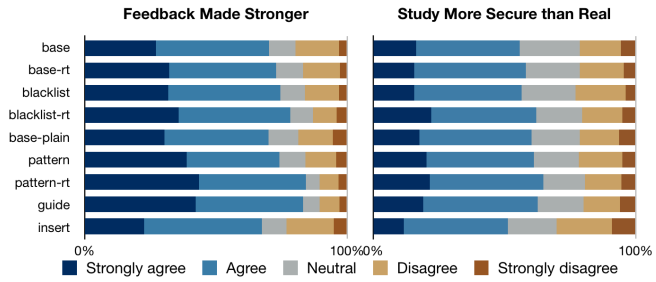

Figure 8: Participant agreement with "The feedback and instructions I saw while creating my password led me to create a stronger password than I would have otherwise." and "If my main email provider had the same password requirements as used in this study, my email account would be more secure."

ticipants were more likely to exceed the minimum than those in *Guide* or *Insert*. In addition, 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 significant differences are shown in Table 3. 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 letter makes a password 70% as likely to be guessed [11]. Thus, the length difference between *Pattern*$_{rt}$ and the three-step conditions may contribute substantially to the security differences between these conditions.

User Perception of Password Strength

To measure perception of how feedback affected password security, we asked our participants 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." Figure 8 shows the responses. 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). A 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).

We asked participants 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." Figure 8 shows responses. *Pattern*$_{rt}$ participants (64.9%) were more likely to agree than *Insert* participants (51.2%) (HC FET, p<.001). 79.6% of participants indicated that their primary email account was from a web email provider. The low agreement is surprising because the study requirements are much more demanding than most web email providers.

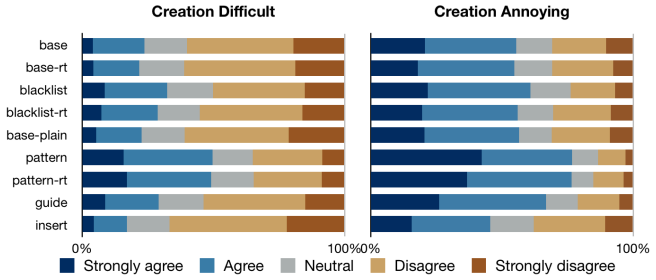

Figure 9: Participant agreement with "Creating a password that meets the requirements given in this study was difficult/annoying."

Usability

This section presents results for usability metrics and self-reported sentiment. We begin by focusing on the password-creation process, and then Part One and Part Two recall.

Password Creation

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

Password-Creation Sentiment

Figure 9 illustrates participants' agreement with password creation being difficult and annoying respectively, with pairwise differences in agreement shown in Table 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 three-step conditions reduced the difficulty and annoyance of the *pattern* requirement (see Q_4).

Password-Creation Time

Table 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). Requirements feedback helped participants create passwords in *Pattern*$_{rt}$ more quickly than in *Pattern* (see Q_3).

Password-Creation Errors

Table 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 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 the impact of guiding and insertion (see Q_4), we examined the errors participants made when creating passwords in each of our four conditions with the pattern requirement. *Pattern* had the highest error rate, with an average of 1.5 creation errors per participant. The most common error was failure to meet the pattern requirement, with an average of 1.1 such errors per participant. With the addition of real-time feedback, *Pattern*$_{rt}$ participants made significantly fewer errors: .6 overall errors and .3 pattern errors each. While *Guide* participants 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. We saw similar sentiment improvements for *Insert* participants, and significant improvement in password-creation time, without a reduction in overall error rate. *Insert* participants averaged .7 errors and .07 pattern errors. This suggests that requirements feedback reduces the high error rate associated with the pattern requirement, but our three-step approaches did not result in further error-rate reductions.

Effects of Branding

We did not observe any significant effects of the university branding (see Q_2). To understand the impact of branding, we looked for the presence of context-related keywords in *Base* and *Base-plain*. *Base* was more likely to contain keywords related to the university (1.3% to 0.7%),[3] but the difference was not significant (χ^2_1=0.73, p=0.393). On the other hand, *Base-plain* passwords were more likely to contain generic study-related keywords,[4] (2.3% to 1.0%) but this was also not significant (χ^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 4. Passwords in *Pattern* took significantly more time to enter than those in either *Base* or *Blacklist*. This may indicate participants struggling more to remember or type passwords created under these conditions.

Part Two Recall

This section looks at data 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 (χ^2_8 =13.844, p=.086), as shown in Table 4.

Participants who mentioned storing their passwords in the surveys, or whom we detected pasting or autofilling during password recall, are considered *storage* participants. Others are *no-storage* participants. 58.5% of returned participants are storage participants, with significant differences shown in Table 3. *Insert* participants were more likely to be storage participants than those in *Pattern*$_{rt}$. Those in *Pattern* were more likely to do so than participants in *Base* or *Blacklist*.

[3] "cmu," "carnegie," "mellon," or "university"

[4] "turk," "amazon," "mechanical," "survey," "study," or "research"

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 and therefore write them down. 57.1% of no-storage participants entered their passwords successfully in one attempt, and this did not vary by condition (χ^2_8=12.189, p=0.143).

We asked whether participants had already used their study passwords. 74.1% indicated creating a new one, and 23.0% indicated using a similar password. Few (4.4%) reused a password exactly, likely because our requirements differ from most service providers. *Pattern* participants were more likely to create a brand new password (84.0%) than *Base* (76.7%) or *Blacklist* (79.2%). This may help explain why *Pattern* participants were more likely to store their passwords.

We asked participants whether they agreed with the statement, "Remembering the password I used for this study was difficult." Table 4 shows significant differences. *Pattern* was more difficult to recall than either *Base* or *Blacklist*.

DISCUSSION

We address our research questions in light of our findings.

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 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 conditions, *Pattern* proved least usable: participants had more difficulty both creating and recalling their passwords. *Pattern* participants were error-prone in password creation, and they reported finding password creation both difficult and annoying at significantly higher rates (Table 2). Participants in this condition also spent more time recalling their passwords in Part One and were more likely to store their passwords (Table 3). While error rates during Part Two recall did not differ significantly, *Pattern* participants were more likely to report difficulty in Part Two recall (Table 4).

Usability differences between *Blacklist* and *Base* were less pronounced and generally limited to password creation. Adding a blacklist requirement may increase security without making password recall significantly more difficult.

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 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.

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 3, 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.

Requirements feedback seems reduce 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.

Q4: Impact of guiding and insertion

We tested the impact of our three-step conditions by comparing *Pattern*$_{rt}$ with both *Guide* and *Insert*. All three conditions had the same requirements. 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 difficult than *Pattern*$_{rt}$, as seen in Table 2. They were also more likely to complete Part One of the study. 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 3). This demonstrates that looking only at password-composition requirements is insufficient to paint an accurate picture of resulting security.

Participants in the three-step conditions made shorter passwords with fewer character classes (Table 3), resulting in more easily cracked passwords. One possible explanation is that participants did not feel a sense of ownership over their passwords. Prior research suggests that passwords are a way that users feel personal responsibility for computer security [14]. Perhaps this sense of ownership and responsibility was diminished because the system was more of an active participant in the process. Another explanation is that users may have trusted that the system was helping them create a strong password, and focused on following the instructions rather than on trying to increase password security.

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 *Guide* might ask participants to create a simple password with at least 11 characters, rather than 10, to account for the fact that participants following a traditional one-step password-creation process are more likely to exceed minimum length requirements. Alternatively, the guidance might encourage participants to exceed minimum requirements by

telling them that this is a good way to increase the security of their password. *Guide* specifically told participants to add "two more characters to the middle of your password" and did not suggest that they could add more. Overall, we observe 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.

CONCLUSION

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. Our multi-step password-creation processes – guidance and insertion – made password-creation easier, but resulted in weaker passwords. While prior passwords research 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.

ACKNOWLEDGMENTS

This research was supported in part by NSF grants DGE-0903659 and CNS-1116776, and by a gift from Microsoft Research.

REFERENCES

1. Brantz, T., and Franz, A. The Google Web 1T 5-gram corpus. Tech. Rep. LDC2006T13, Linguistic Data Consortium, 2006.
2. Chiasson, S., Forget, A., Stobert, E., Biddle, R., and P.C. van Oorschot. Multiple password interference in text and click-based graphical passwords. In *CCS* (2009).
3. Egelman, S., Sotirakopoulos, A., Muslukhov, I., Beznosov, K., and Herley, C. Does my password go up to eleven?: The impact of password meters on password selection. In *CHI* (2013).
4. Fahl, S., Harbach, M., Acar, Y., and Smith, M. On the ecological validity of a password study. In *SOUPS* (2013).
5. Forget, A., Chiasson, S., van Oorschot, P. C., and Biddle, R. Improving text passwords through persuasion. In *SOUPS* (2008).
6. Furnell, S. An assessment of website password practices. *Computers & Security 26*, 7 (2007), 445–451.
7. Furnell, S. Assessing password guidance and enforcement on leading websites. *Computer Fraud & Security 2011*, 12 (2011), 10 – 18.
8. Furnell, S., and Bär, N. Essential lessons still not learned? Examining the password practices of end-users and service providers. In *Human Aspects of Information Security, Privacy, and Trust* (2013), 217–225.
9. 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 SP* (2012).
10. Kerby, D. S. The simple difference formula: An approach to teaching nonparametric correlation. In *Innovative Teaching*. 2014.
11. 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).
12. 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.
13. Schneier, B. Myspace passwords aren't so dumb. `http://www.wired.com/politics/security/commentary/securitymatters/2006/12/72300`, 2006.
14. 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).
15. 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).
16. 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).
17. Ur, B., Kelley, P. G., Komanduri, S., Lee, J., Maass, M., Mazurek, M., Passaro, 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).
18. Vance, A. If your password is 123456, just make it HackMe. The New York Times, `http://www.nytimes.com/2010/01/21/technology/21password.html`, January 2010.
19. Weir, M., Aggarwal, S., Collins, M., and Stern, H. Testing metrics for password creation policies by attacking large sets of revealed passwords. In *CCS* (2010).
20. Weir, M., Aggarwal, S., de Medeiros, B., and Glodek, B. Password cracking using probabilistic context-free grammars. In *IEEE SP* (2009).