Of Passwords and People: Measuring the Effect of Password-Composition Policies

Saranga Komanduri[1], Richard Shay[1], Patrick Gage Kelley[1], Michelle L. Mazurek[1], Lujo Bauer[1], Nicolas Christin[1], Lorrie Faith Cranor[1], and Serge Egelman[2]

[1]Carnegie Mellon University
Pittsburgh, PA
{sarangak, rshay, pgage, mmazurek}@cmu.edu
{lbauer, nicolasc, lorrie}@cmu.edu

[2]National Institute of Standards and Technology
Gaithersburg, MD
serge.egelman@nist.gov

ABSTRACT
Text-based passwords are the most common mechanism for authenticating humans to computer systems. To prevent users from picking passwords that are too easy for an adversary to guess, system administrators adopt password-composition policies (e.g., requiring passwords to contain symbols and numbers). Unfortunately, little is known about the relationship between password-composition policies and the strength of the resulting passwords, or about the behavior of users (e.g., writing down passwords) in response to different policies. We present a large-scale study that investigates password strength, user behavior, and user sentiment across four password-composition policies. We characterize the predictability of passwords by calculating their entropy, and find that a number of commonly held beliefs about password composition and strength are inaccurate. We correlate our results with user behavior and sentiment to produce several recommendations for password-composition policies that result in strong passwords without unduly burdening users.

Author Keywords
Security, Usability, Passwords, Policy

ACM Classification Keywords
D.4.6 Security and Protection: Authentication; H.1.2 User/ Machine Systems: Human factors

General Terms
Experimentation, Human Factors, Measurement, Security

INTRODUCTION
To prevent attackers from predicting users' text-based passwords, and hence impersonating users, system administrators typically require that users select passwords according to a password-composition policy designed to make users' passwords harder to predict. Such a policy may require, for example, that passwords exceed a minimum length, that they contain uppercase letters and symbols, and that they do not contain dictionary words.

Unfortunately, it is difficult to define precisely the relationship between the components of a password-composition policy and the predictability of the resulting passwords, in large part because of a lack of empirical data on passwords and the policies under which they were created. Even the best current guidelines for designing password-composition policies, for instance, are based on theoretical estimates [4] or small-scale laboratory studies (e.g., [12, 20]).

What makes designing an appropriate password-composition policy even trickier is that such policies affect not only the passwords users create, but also users' behavior. For example, certain password-composition policies that lead to more-difficult-to-predict passwords may also lead users to write down their passwords more readily, or to become more averse to changing passwords because of the additional effort of memorizing the new ones. Such behavior may also affect an adversary's ability to predict passwords and should therefore be taken into account when selecting a policy.

With this paper, we take a significant step toward improving our understanding of how password-composition policies influence the predictability of passwords, as well as how they affect user behavior and sentiment. We describe the results of a two-part user study with more than 5,000 participants. In the first part we required each participant to create a password under one of four different password-composition policies. In the second part we asked participants to recall their passwords at least two days later. We also surveyed users to capture their sentiment toward a password-composition policy, as well as to to learn about their password-related behavior (e.g., whether and how they recorded the password).

Using the collected data, we characterize the predictability of passwords created under various password-composition policies by computing their entropy. Our results are the first entropy estimates derived from a large-scale empirical study that allow for comparison of entropy across different password-composition policies. Combining these results

Copyright 2011 Association for Computing Machinery. ACM acknowledges that this contribution was authored or co-authored by an employee, contractor, or affiliate of the U.S. Government. As such, the Government retains a nonexclusive, royalty-free right to publish or reproduce this article, or to allow others to do so, for Government purposes only.
CHI 2011, May 7–12, 2011, Vancouver, BC, Canada.
Copyright 2011 ACM 978-1-4503-0267-8/11/05....$10.00.

with our survey data, we compare the effects of different password-composition policies more comprehensively than was previously done, yielding a number of interesting findings. For instance, we compared two password-composition policies: one required only that passwords be at least 16 characters long; the other required at least eight characters but also an uppercase letter, a number, a symbol, and a dictionary check. According to the best available guidelines [4], these two policies should result in passwords of approximately the same entropy. We find, however, that the 16-character policy yields significantly less predictable passwords, and that it is, by several metrics, less onerous for users. We believe this and other findings will be useful both to security professionals seeking to establish or review password-composition policies, and to researchers examining how to make passwords more secure and usable.

In the next section, we provide background and survey related work. Next, we describe the methodology of our study, followed by the demographics of our participants and an explanation of how we analyzed our data. We then individually describe the results we derive for password entropy and for user behavior (including password-composition strategies, password storage and memorability, user sentiment, and ecological validity). We conclude with a discussion in which we weave together the individual results to give a more complete picture of the effects of different password-composition policies, and we also draw out the most significant findings.

BACKGROUND AND RELATED WORK

We evaluate tradeoffs between the empirical strength of passwords generated under a variety of password-composition policies and the pain those policies cause users. Others have examined similar tradeoffs. In a lab study, Proctor et al. measured creation time, memorability, and crackability of passwords under various composition policies, finding that stricter policies can make passwords harder to crack but also harder to create and remember. Overall, they found that increasing minimum length was more effective than applying content constraints [12]. We collected similar data for a much larger set of users, allowing us to estimate the entropy associated with various password-composition policies.

Several studies report that if password-composition policies are too demanding (e.g., when it is difficult to generate an acceptable password, or when passwords must be changed frequently), users will adopt coping strategies that can reduce both security and productivity [1, 7, 16, 18]. Vu et al. found that generating passwords from the first letter of each word in a sentence reduced vulnerability to cracking as compared to traditional rules about including numbers, symbols and capitals, but at some cost to memorability [20].

Florêncio and Herley report that stricter password-composition policies do not correlate with protection of high-value assets, but instead with how well insulated organizations are from customer service concerns [6]. Schechter et al. designed a popularity oracle to replace traditional password-composition policies with a tool that rejects passwords that have already been selected by too many other users [13].

Others have attempted to quantify password strength using real-world data, without examining the impact of policy restrictions. In 1989, 40% of 14,000 UNIX accounts were cracked using guesses derived from associated usernames or account numbers and dictionaries [2]. Florêncio and Herley, reporting in 2007 on data collected from half a million Windows Live users, calculated an average password strength of 40.54 bits, with stronger passwords associated with accounts with higher apparent importance [5]. In 2010, Zhang et al. found that 41% of passwords from a university system could be cracked in under three seconds each, given knowledge of expired passwords from the same account [22].

Similar studies have evaluated password characteristics using self-reported data. In a 1999 survey of military users, with no apparent compositional policies, Zviran and Haga found that most passwords contained four to seven characters. Eighty percent of participants claimed to use only alphabetic characters, with only 13% using alphanumeric passwords and less than a percent using symbols [23]. Ten years later, a survey of university users under a strict password-composition policy requiring at least eight characters, including at least one each of lowercase and uppercase letters, numbers, and symbols found an average password length of more than ten and almost three numbers per password [17]. These studies, along with our results, suggest that passwords have become more complex over time, whether due to increasingly strict policies, user education, or other factors.

While these empirical studies suggest an increase in password complexity over time, formally measuring password complexity, as we propose, requires defining a complexity metric. One possible metric is the information entropy of the password: a measure (in bits) of the expected value of information contained in the password. Shannon introduced the notion of information entropy, and subsequently developed a method to estimate entropy in printed English using n-grams [14]. Massey showed that entropy provides a lower bound on the expected number of guesses required to identify information; this result connects the entropy in passwords with an attacker's ability to guess them using a brute-force attack [9]. Miller showed that given good sampling but insufficient samples, entropy approximations always underestimate the true entropy; Paninski derived a threshold at which the sample size is sufficient, based on the number of observed categories [10, 11]. Paninski's method informed our choice of an appropriate sample size for our study.

Similar to previous work [17], we use a variation of Shannon's method for calculating entropy [15] that allows us to approximate the entropy of a password space from a much smaller dataset than traditional methods. Florêncio and Herley estimated entropy for a large set of real passwords they observed, calculating the theoretical maximum for each password given the set of character types used [5]. This calculation does not account for the actual distribution of passwords in use, because they were only able to collect transient information about passwords rather than store the passwords themselves. Our method calculates entropy more precisely, but we used passwords created for the study rather than pass-

words for real accounts; in the Discussion section we address how this affects our results. Weir et al. suggest that entropy by itself might not be the best metric for evaluating the security of a given password [21]. Along with computing entropy, we also briefly explore the resistance of passwords to guessing by a modern password-cracking algorithm.

METHODOLOGY

We conducted a two-part online study using Amazon's Mechanical Turk service.[1] We recruited 5,000 participants, each assigned randomly to one of five conditions. In the first part, we asked participants to create a password, complete an online survey, and then enter the password. Two days later we asked them to return to our website, login using their password, and complete a second survey.

Study Overview

We advertised our study on Mechanical Turk as a two-part "brief study" with a "bonus opportunity." We paid participants between 25 and 55 cents for the first part, and between 50 and 70 cents for the second part. The consent form informed participants that the study would survey users about their behavior visiting secure websites.

We gave participants a *scenario* and asked them to create a password that complied with a a *password-composition policy*. The scenarios and policy varied by condition, and are discussed below. We asked participants to create a password and enter it twice. We informed participants who entered passwords that failed to meet the requirements about which requirement they failed to meet and asked them to retry. After entering an acceptable password, we presented participants with a survey requesting demographic information, Likert questions about the password-creation process, and questions about the password-creation strategy employed.

We asked participants to enter the password they had just created. If a participant was unsuccessful in recalling her password in five attempts, the password was displayed. We then displayed a code for participants to enter into Mechanical Turk to receive payment. We told them they would be contacted to complete follow-up surveys, but not when we would contact them.

Two days after completing the first part, we sent participants emails asking them to return for the second part via an included URL. On return, we asked participants again to recall their passwords. The password-entry screen included an "I forgot my password" link to a page with an "Email password reset link" button and a note informing them that resetting a password "may take a few minutes." Participants who clicked the button were emailed a link to a page displaying their password. Participants who failed to recall their password after five tries were also shown their password on the screen. Participants were then presented with a second survey, which included additional questions about password creation, storage, and usage.

[1]Products are identified to specify the experimental procedure adequately. This is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology.

Conditions

Our pilot studies informed our selection of five experimental conditions. These conditions allowed us to explore four sets of password composition requirements across two scenarios.

Instructions in the *survey scenario* read: "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*, we told participants, "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!"

Condition basic8survey. We gave participants the survey scenario and the password-composition policy, "Password must have at least 8 characters." NIST predicts that such passwords would have 18 bits of entropy [4].

Condition basic8. We gave participants the email scenario and the password-composition policy, "Password must have at least 8 characters." This condition had the same password composition policy as the basic8survey condition, allowing us to observe the impact of framing on participant behavior.

Condition basic16. We gave participants the email scenario and the password-composition policy, "Password must have at least 16 characters." This condition required passwords twice as long as the basic8 condition, allowing us to observe the impact of increasing password length. NIST predicts that such passwords would have 30 bits of entropy [4].

Condition dictionary8. We gave participants the email scenario and the password-composition policy, "Password must have at least 8 characters. It may not contain a dictionary word." We performed a dictionary check consisting of removing all non-alphabetic characters from the password and looking up the remaining letters in a dictionary,[2] ignoring case. This method is known to be used in practice, including at CMU. According to NIST, adding the dictionary check raises the entropy of 8-character passwords to 24 bits [4].

Condition comprehensive8. We gave participants the email scenario and 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 type of dictionary check as in the dictionary8 condition. This condition was designed to reproduce NIST's comprehensive password

[2]http://download.openwall.net/pub/ wordlists

composition requirements [4]; NIST estimates that such 8-character passwords have 30 bits of entropy, similar to 16-character passwords with no composition requirements.

DEMOGRAPHICS AND ANALYSIS

A total of 6,212 participants began our study over a 48-day period from August to October 2010. 5,103 participants completed part one. We rejected four participants because their completion times were outside two standard deviations of the mean. We sent an email to participants two days after they completed part one inviting them to return for part two. 3,056 returned and 2,889 completed part two within three days of receiving the email. Any participant who returned more than three days after being emailed received payment, but we excluded their part two responses from our analysis.

Our entropy calculations require exactly the same number of passwords per condition to assure comparability across conditions. Therefore, we selected the first 1,000 participants in each condition who completed part one, regardless of whether they returned for part two. The remainder of this paper discusses the data from these 5,000 participants.

Fifty-one percent of our participants indicated they were male, 47% female, and 2% did not disclose gender. The mean age for our participants is 29.83 ($\sigma = 9.96$). Thirty-three percent reported studying or working in computer science, information technology, or a related field. We found no statistically significant difference in age, gender, or technical experience between conditions using a chi-square test.

In analyzing our data, we used Fisher's Exact Test (FET) to compare proportions when the sample size was tractable, and chi-square tests when it was not. Post-hoc comparisons were corrected for multiple-testing using the Bonferroni-Holm method when appropriate. Permutation testing was used to analyze differences between entropy estimates. No standard test exists for this purpose. The space of permutations, $\binom{2000}{1000}$ in most cases, was too large to perform an exact test, so random sampling of the permutation space was performed with 5,000 permutations per test.

PASSWORD COMPOSITION AND ENTROPY

In this section we explain our method for estimating entropy, then present and discuss our estimates for passwords created under the different conditions. We also interpret these results in light of using heuristics to guess passwords.

Entropy Calculation

We calculated entropy using a variation of Shannon's method [15] from earlier work [17]. More specifically, Shannon's formula for estimating entropy allows the entropy for a distribution of passwords (our final goal) to be calculated by summing the entropy values derived from each element of a password. Hence, we calculate individually the entropy contributed by password length; by number and placement of each class of character (lowercase, uppercase, numbers, symbols); and by the content of each character. The sum of these values is our estimate of the entropy of a password distribution.

Calculating an entropy estimate in this additive fashion gives us improved accuracy with smaller sample sizes than would be possible with traditional trigram methods. Traditional trigram methods, when applied to passwords, require thousands of samples to calculate an accurate estimate [11]. We confirmed empirically that the method we adopt results in useful approximations by applying both methods to multiple random samples (of 500, 1000, 10,000, 50,000, 100,000, and 500,000 passwords) of the RockYou password dataset [19]. The RockYou dataset was used because it is the only dataset large enough to support samples of this size. At the sample size of 1,000 that we use in our study, the numbers we show are significant underestimates of the true entropy, as determined by trigram methods. Hence, it is inappropriate to compare them directly to previously published results (e.g., [4]), and we use them chiefly to compare between conditions. The exception is when we find our *underestimates* to be significantly higher than previously reported estimates.

We show both the cumulative entropy and component entropies for each of the conditions in our study in Table 1.

Entropy Results

Our entropy calculations yield a number of interesting observations. Most notably, the entropy estimates for each condition are statistically significantly different ($p < 0.001$, permutation test, corrected), with the exception of dictionary8 and basic8, which do not differ significantly from each other ($p = 0.36$). This is notable partly because NIST estimates suggest that basic16 and comprehensive8 should result in passwords of the same entropy; we find, however, that basic16 (at 44.67 bits) has significantly more entropy than comprehensive8 (at 34.30 bits). Similarly, NIST estimates suggest that adding a dictionary check increases entropy (by an estimated 6 bits). Our basic8 and dictionary8 conditions were designed to verify that hypothesis. Surprisingly, we find adding the dictionary check actually resulted in a non-statistically-significant decrease in the observed entropy.

Adding numbers to passwords is thought to reduce predictability only slightly, due to the assumption the digits will occur in easily predictable locations within the passwords [4]. Our findings, however, are consistent with recent work [17], and show that a significant amount of the entropy of passwords is contributed by the portions of the passwords composed of numbers. This is a result of several factors. First, we find that the entropy per digit is higher than the entropy per lowercase letter (e.g., 2.82 vs 1.75 in the comprehensive8 condition), indicating that there is less consistency in which digits users choose than which letters they choose, despite the fact that there are 26 lowercase letters and only 10 digits. Second, participants used a larger than expected number of digits in their passwords. In all conditions, including the four that did not require a digit, participants' passwords had a mean of at least 2.20 digits (see Table 1). In particular, basic16 passwords used on average 3.76 numbers, significantly greater than in any other condition (corrected pairwise t-test, $p < 0.05$). Finally, there was greater than expected variability in where the users chose to put digits (see Figure 1).

	comprehensive8	basic16	dictionary8	basic8	basic8survey
Entropy in how many numbers	2.10	3.15	2.76	2.81	2.74
Entropy in where they are	1.61	1.92	1.80	1.84	1.62
Entropy in what they are	6.22	10.56	7.18	6.84	6.94
Total entropy in numbers	**9.94**	**15.63**	**11.74**	**11.49**	**11.30**
Entropy in how many symbols	0.67	0.58	0.56	0.51	0.30
Entropy in where they are	1.55	0.56	0.58	0.56	0.34
Entropy in what they are	3.52	0.37	0.43	0.30	0.13
Total entropy in symbols	**5.75**	**1.51**	**1.56**	**1.37**	**0.78**
Entropy in how many uppercase letters	1.30	0.99	0.90	1.02	0.68
Entropy in where they are	1.32	0.75	0.74	0.85	0.53
Entropy in what they are	2.62	0.89	0.64	0.73	0.48
Total entropy in uppercase letters	**5.24**	**2.63**	**2.29**	**2.60**	**1.69**
Total entropy in lowercase letters	**10.32**	**22.21**	**10.85**	**11.47**	**11.22**
Total entropy in length	**3.05**	**2.69**	**2.55**	**2.50**	**2.21**
Total entropy	**34.30**	**44.67**	**28.99**	**29.43**	**27.19**

Password composition

Numbers	mean (median)	2.20 (2)	3.76 (2)	2.50 (2)	2.38 (2)	2.37 (2)
Symbols	mean (median)	1.13 (1)	0.16 (0)	0.16 (0)	0.12 (0)	0.06 (0)
Uppercase letters	mean (median)	1.50 (1)	0.54 (0)	0.40 (0)	0.43 (0)	0.28 (0)
Lowercase letters	mean (median)	5.90 (6)	13.46 (14)	6.65 (7)	6.74 (7)	6.54 (7)
Length	mean (median)	10.76 (10)	17.98 (17)	9.72 (9)	9.66 (9)	9.27 (9)
Entropy per usage	per number	2.82	2.81	2.87	2.88	2.93
	per symbol	3.11	2.36	2.65	2.59	2.29
	per letter	1.75	1.65	1.63	1.70	1.71

Table 1. Upper table shows entropy calculations across conditions, broken down by the entropy contributed by length and character categories. Lower table characterizes passwords by length and composition, across conditions.

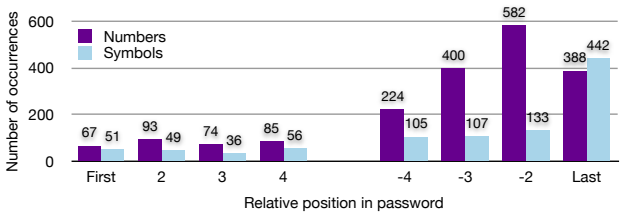

Figure 1. The distribution of digits and symbols in passwords in the comprehensive8 condition, relative to the start (left-most pair of bars) and end (right-most pair of bars) of the password. E.g., the pair of bars at '-3' shows the number of digits and symbols that occur as the third-to-last character of passwords. We omit digits and symbols more than four positions from either end of passwords to avoid duplicates, since many passwords are exactly eight characters long.

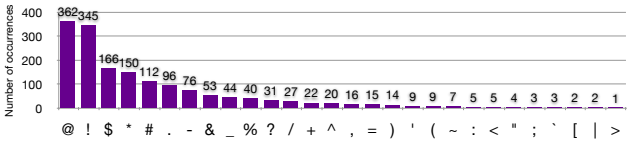

Figure 2. Frequency of occurrence of symbols in passwords created in the comprehensive8 condition.

As with numbers, participants used a variety of symbols; '@' and '!' were the most common, but many others were also used, causing the per-symbol entropy to exceed 2 in all conditions (see Figure 2 for a frequency distribution graph of symbols). However, symbols were used much less frequently than numbers. Only a small fraction of participants made use of symbols when they were not forced to do so by the password-composition policy (means of 0.06–0.16 symbols per password when not required by policy). Even when required by policy to use both symbols and numbers, participants favored numbers; in the comprehensive8 condition, which required both a symbol and a number, participants used a mean of 2.20 numbers but only 1.13 symbols. Hence, the overall entropy contributed by symbols is lower that that contributed by numbers, across all conditions.

Interestingly, median lengths consistently exceeded requirements (see Table 1), adding to entropy by making it harder to guess length. Participants in the comprehensive8 condition exceeded the length requirements by a mean of 2.76 characters, which is statistically significantly greater than the amount by which participants in other conditions exceeded the requirements (corrected pairwise t-test, $p < 0.001$).

	comp8	bas16	d8	bas8
Cracked	0/997	9/971	31/963	188/972
Cracked + passed dict. check	0/997	6/971	26/963	99/972
Cracked + not in large dict.	0/997	1/971	3/963	28/972

Table 2. Number of passwords cracked, cracked passwords that passed our dictionary check, and cracked passwords that would pass a dictionary check even against our largest dictionary; across all study conditions. Several users in each condition used identical passwords, which is why the number of unique passwords reported here is less than 1,000.

Dictionary Checks and Entropy

Including a dictionary check in the password-composition policy did not change the observed entropy significantly, but it still may strengthen passwords. Entropy is a good measure of the information-theoretic predictability of passwords, but it is possible even within a password distribution of high entropy to have a small set of passwords that are "easily" guessable. To evaluate this possibility, we used the popular password cracker John the Ripper[3] across conditions. We used the tool's heuristic mode, which transforms dictionary words into password guesses. We do not consider other types of guessing attacks, such as informed guessing based on knowledge of the user, in this discussion. The results are shown in Table 2. Comparing basic8 and dictionary8, it is notable that John the Ripper guessed 188 of 972 unique passwords in basic8, but only 31 of 963 in dictionary8. This difference is statistically significant (FET, $p < 0.0001$).

If we had performed a dictionary check on basic8, 89 of those 188 guessed passwords would have been rejected. Both these data points illustrate that dictionary checks, even if they do not increase entropy, help significantly in producing passwords more resistant to heuristic guessing.

Also notable is that the comprehensive8 and basic16 conditions produced passwords considerably more resilient to cracking, with 0 and 9 passwords successfully cracked. This appears to indicate that long passwords (as evidenced by basic16) are resistant to cracking even without a dictionary check, and that passwords that contain symbols and numbers in addition to a dictionary check (comprehensive8) are significantly harder to crack that those that contain a dictionary check (dictionary8). Unsurprisingly, the choice of dictionary used for the dictionary check also has an effect on the guessability of passwords. In our regular dictionary check, we used a comprehensive dictionary that contained about 3.9 million entries (words, common phrases, misspelled words, names and titles, etc.). As shown in Table 2, if we had instead used a larger dictionary with 24.7 million entries (also obtained from the Openwall Project), an overwhelming majority of the cracked passwords would have been caught by a dictionary check at password-creation time.

USERS AND THEIR BEHAVIOR

In this section, we discuss how variations in the password-composition policies affect users' behavior when creating and remembering passwords, as well as their impressions of the process. Overall, we find significant differences among the policies, with comprehensive8 proving least user-friendly on several measures, while unsurprisingly basic8 and basic8survey proved generally easiest for users to cope with. Less predictably, basic16 proved better than the comparable-strength comprehensive8 in several respects.

[3]http://www.openwall.com/john

	comp8	bas16	d8	bas8	surv
Average attempts to create a password					
Total	3.35	1.66	1.88	1.13	1.17
Among those who failed at least once	3.85	2.37	3.00	2.11	2.04
Password creation failures					
% failed confirmation	3.1	3.4	2.4	2.4	2.2
% failed policy	57.6	39.4	3.59	8.5	12.7
% failed both	21.6	4.5	5.1	0.5	0.3
Dropouts per condition					
% dropouts	25.0	15.0	18.3	14.7	16.5
Password reuse					
% exact reuse	3.3	2.3	19.7	25.6	34.6
% modified reuse	31.3	25.0	19.2	19.1	17.7
% created new	62.5	66.0	53.3	48.2	39.2

Table 3. Overview of password-creation data, across conditions. The average number of attempts required to successfully create a password, both among all participants and among only those who failed at least once. The percentage of participants who failed to create a valid password at least once; either because their initial and confirmation passwords did not match, because their password did not conform to requirements, or both. The percentage of potential participants who dropped out before completing part one. Finally, the percentage of participants who reused a password exactly, reused with modifications, and created an entirely new password.

Password Creation, Storage, and Memory

We obtained several interesting results on how users create passwords, including how often they fail to make a valid password, how they cope with such failures, and how often they report reusing passwords from other accounts. Most notably, our data demonstrate that successfully creating a password is significantly more difficult under stricter password policies, particularly those involving dictionary checks. Table 3 contains detailed data about these findings.

Failed creation attempts

We measured how many participants failed to create an acceptable password at least once, either because their "enter password" and "confirm password" entries failed to match or because the password they selected did not conform to the policy requirements. The comprehensive8 policy condition proved by far the most difficult, as only 17.7% of users in this condition could create a password in one try. By contrast, 52.7%, 56.6%, 88.6%, and 84.8% of participants in the basic16, dictionary8, basic8, and basic8survey conditions respectively created an acceptable password in one try.

As shown in Table 3, comprehensive8 participants required on average 3.35 attempts to create an acceptable password. This was significantly more attempts than in dictionary8 (mean 1.88), which was significantly more than in basic16 (mean 1.66). Those three, in turn, were significantly more failure-prone than basic8, with an average of 1.13 attempts per participant (all $p < 0.001$, pairwise-corrected t-tests).

Counting only participants with at least one failure, comprehensive8 participants also required the most attempts, averaging 3.85. Participants in the dictionary8 condition required significantly fewer attempts, at 3.00, than comprehensive8, but significantly more than any other condition ($p < 0.001$, pairwise corrected t-tests). These results suggest that the dictionary check was especially difficult for participants to deal with.

Coping with failure

Perhaps the dictionary checks gave our participants so much difficulty because they had trouble determining whether a password would be acceptable without trying it. Because our technique stripped non-alphabetic characters before testing against the dictionary, a participant who tried a password like "p1a1s1s1w1o1r1d" would be rejected, perhaps without understanding why. We have several examples of participants trying and failing to pass the dictionary check that seem to bolster this hypothesis. For example, one user made the following sequence of password attempts, first seemingly not aware that the dictionary check ignored digits, and on subsequent failures changing parts of the password (such as numbers and symbols) that were irrelevant to the dictionary check: "cheese" → "1cheese1" → "12#$asdf" → "12#$qwER" → "43@!reWQ".

This process may have been made more difficult because the error message indicated only that the password was not allowed to contain a dictionary word, rather than explaining the check mechanism in detail or highlighting the problem characters. In addition, our dictionary includes non-standard "words" based on common passwords that users might not expect to find in a dictionary.

Participants' strategies to cope with these apparently confusing failures fall into two broad categories: incrementally modifying the initial password attempt until a successful password is found, and changing directions entirely. Within the dictionary8 condition, 31% of participants who failed password creation used only incremental modifications, while the other 69% changed strategies entirely, including 16% who switched to an all-numeric password. Within comprehensive8, 38% of those who failed used incremental changes, with the other 62% changing direction.

Giving up on creating a password

Another indicator of the difficulty participants had creating passwords is the rate at which they dropped out before completing part one, possibly because they had given up on making an acceptable password. This data indicates that the comprehensive8 condition caused prospective users considerably more trouble than the other conditions.

Considering only users who began the study no earlier and no later than participants included in our final data (6,212 in total), we find that 25% in the comprehensive8 condition dropped out before completing part one, compared to less than 19% in each of the other conditions. Dropout rates for comprehensive8 were significantly higher than all other conditions, none of which differed significantly from each other ($p < 0.001$, pairwise corrected FET).

To test our theory that this greater dropout rate was motivated by difficulty creating acceptable passwords, we looked at how far participants progressed before quitting. Among 1,108 prospective participants who dropped out on the first day, 68% in the comprehensive8 condition dropped out while trying to create a password, along with 32% in dictionary8 and no more than 26% in any other condition. This is significantly higher for comprehensive8 than any other condition, and significantly higher for dictionary8 than either basic8 or basic8survey ($p < 0.002$, pairwise corrected FET).

Reusing passwords

We also examined how likely participants were to report reusing a password from another account, exactly or with modifications, when making a password for our study. Participants in the comprehensive8 and basic16 conditions were significantly less likely to report reusing a password exactly than those in the other conditions. By contrast, these participants were more likely than other participants to report reusing existing passwords with modification, though this difference was not always statistically significant.

Both of these practices can be dangerous; reusing passwords exactly is known to lead to security breaches [3, 8], and Zhang et al. found that modifications tend to be predictable, making the resulting passwords easy to crack [22]. We hypothesize that because comprehensive8 and basic16 are more stringent than the others, our participants were less likely to already have suitable passwords; as a result, they frequently modified an existing password instead. Nonetheless, when considering all participants who reused a password either exactly or with modifications, participants in the comprehensive8 and basic16 conditions still reported significantly less reuse than in other conditions.

Storing passwords

Overall, 31% of participants who returned for part two reported writing down the password they used for the study, either electronically or on paper. (See Figure 3 for details.) We found no significant difference in rates of reported storage on paper between our study passwords and real email passwords; however, reported electronic storage of study passwords is significantly lower than for real email passwords.

A significantly greater proportion (50%) of comprehensive8 participants stored their passwords than in all other conditions; and basic16 participants were significantly more likely to store (33%) than basic8 and basic8survey participants (26% and 17% respectively, $p < 0.03$, pairwise corrected FET).

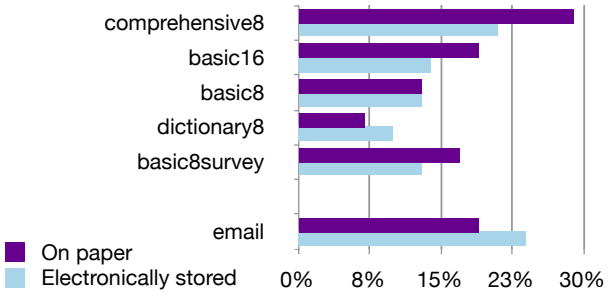

Figure 3. Percentage of users in each condition who stored their password, either on paper or electronically. For reference, we also include the percentage of users storing their actual email password.

Remembering passwords

Overall, 11.1% of participants who returned for part two used the forgotten password email feature, and only 1.6% failed to log in after five attempts and were subsequently shown their password. Participants who did remember their password without being shown required on average 1.22 attempts to log in successfully. We found no significant differences in these rates across conditions, using an ANOVA.

Participants who did not think they would be able to remember their password tended to store it proactively, electronically or on paper. Considering only those participants who did not store their password, 13.0% used the forgotten password feature and 2.1% failed to login in five attempts, compared with 7.1% and 0.4% respectively for those who did store their password. These differences were significant ($p < 0.001$, FET). We found no significant differences in these rates across conditions; however, the fact that users wrote down comprehensive8 and basic16 passwords more frequently than others (as discussed above) suggests that participants perceive these passwords as harder to remember.

This idea is bolstered by results from a Likert question asking participants to respond to the statement "Remembering the password I used for this study was difficult." Thirty-five percent of comprehensive8 respondents agreed or strongly agreed, significantly more than in any other condition ($p < 0.001$, pairwise corrected FET). The next most difficult condition was basic16, with 23% reporting that their password was difficult to remember.

Finally, we measured how many attempts participants made to log in before resorting to the forgotten password e-mail feature. On average, participants made only 1.2 attempts, with 39% making zero attempts, suggesting they quickly determined they would not be able to remember. We found no significant difference in this rate across conditions.

User Sentiment

When we asked users their opinions of the password creation process, the results differed sharply by condition. Unsurprisingly, users generally found conditions with stricter requirements more onerous.

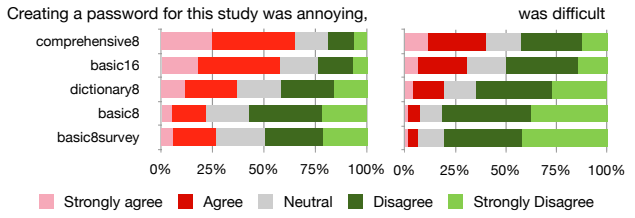

Figure 4. User responses to whether creating a password for this study was annoying or difficult. Answers were reported on a five-point Likert scale, from 1 (strongly agree) to 5 (strongly disagree).

Figure 4 shows user responses to the statement that creating a password for this study was annoying. User annoyance (agree or strongly agree) was significantly different between each condition (all $p < 0.02$, pairwise corrected FET). Responses to the statement "creating a password for this study was difficult" are shown in Figure 4. All pairs of conditions differ significantly in difficulty, except the pair of basic8 and basic8survey ($p < 0.001$, pairwise corrected FET).

We also considered user perceptions of password strength across conditions. Each participant was asked to agree or disagree with the statement "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." Significantly more participants in the comprehensive8 condition (67%) agreed or strongly agreed with this statement than in the basic16 condition (57%); those participants in turn perceived added strength significantly more often than participants in the basic8 and dictionary8 conditions (43% and 49%), whose perceptions were also significantly different (all $p < 0.02$, pairwise corrected FET). Perceptions of password strength may affect users' willingness to "buy in" to new policies that they find annoying.

Ecological Validity

It is difficult to demonstrate ecological validity in any password study where participants are aware they are creating a password for a study, rather than for an account they value and expect to access repeatedly over time. Ideally, password studies would be conducted by collecting data on real passwords created by real users of a deployed system. However, due to the sensitivity of password data and the difficulty of partitioning real users into experimental conditions with different password-composition policies, it is difficult to collect the data we collected in this study from a deployed system.

We tested two approaches to approximate real-life password-composition scenarios. First, recall that in the basic8survey condition, we asked participants to create a password that they would use to complete follow-up surveys. This condition was designed to give us ecologically valid data on the password behavior users exhibit when they create passwords for low-value accounts they expect to use only on a short-term basis. However, this does not necessarily reflect user behavior when dealing with passwords for high-value accounts or accounts used on a long-term basis. Thus, we designed the other conditions to encourage participants to behave in the way they would when creating a password for

one such account (an email account). Two indicators that participants may have answered honestly are that their self-reported password reuse was higher in the basic8survey condition than in the four other conditions, and that the computed entropy of passwords in these four conditions was significantly higher than the entropy of passwords in the basic8survey condition. Both findings are consistent with users picking better passwords to protect a hypothetical email account than to protect a real survey account. Despite this, we cannot conclude that our results completely approximate real-world behavior; because the hypothetical scenario was the same across the four conditions, however, we are confident that our comparisons among conditions are valid.

A further issue is that a significantly larger proportion of comprehensive8 participants than other participants dropped out before completing part one. While this creates a potential for selection bias, we have no evidence to believe one exists. We found no statistically significant difference in demographics between conditions. Furthermore, since entropy is logarithmic and the difference between the entropy of comprehensive8 and the other conditions is large, any bias is unlikely to affect the relative entropy differences we observed between conditions.

DISCUSSION

We have so far presented results that follow immediately from our data. In this section, we weave together results regarding entropy, user behavior, and user sentiment. Although many of these are noteworthy on their own, looking at them together reveals a number of interesting trends. Second, building on that discussion, we summarize the findings that are the main contribution of the paper.

Entropy Tradeoffs

Password-composition policies must strike a balance between maximizing security and minimizing user frustration. Our results indicate that, as might be expected, increases in entropy often correlate with decreases in usability. However, we provide hope for finding a desirable balance; a carefully chosen policy can "buy" as much or more entropy, at a similar or better level of usability, than less optimal policies.

At the low end of the entropy scale, we found that basic8 and dictionary8 do not provide significantly different levels of entropy, although the dictionary check provides extra protection against heuristic cracking. In terms of usability, however, the two conditions have important differences: dictionary8 participants had significantly more difficulty creating a valid password and found the process both more difficult and more annoying. The two conditions are similar, however, with respect to storing and remembering the password, as well as in terms of perceived security. An administrator choosing between these policies will make an explicit tradeoff between added protection from heuristic cracking and increased up-front frustration at creation time.

We also considered two stricter policies: comprehensive8 and basic16. From NIST calculations, we expected these policies to provide similar entropy levels; in reality, however, basic16 provided significantly more entropy. We did not observe a notable difference in resistance to heuristic cracking, but John the Ripper—like other similar tools—is optimized for short passwords. As a result, we cannot say confidently how the two conditions compare for cracking protection. In addition to providing more entropy, however, basic16 also proved as or more usable than comprehensive8 on every measure. We found that basic16 is easier to create, reportedly easier to remember, and less likely to be stored than comprehensive8; the two conditions were similarly annoying to participants. As a result, we consider basic16 clearly superior.

We expected, a priori, that our two strictest policies would provide strictly stronger passwords with strictly less usability than our two less demanding policies. In fact, both basic16 and comprehensive8 provide significantly more entropy than basic8 and dictionary8. But while comprehensive8 is worse on every usability measure than either basic8 or dictionary8, basic16 is not. The basic16 condition proved reasonably comparable to dictionary8 in terms of password creation, storage, and reported memorability, although it did perform worse in user sentiment. Overall, therefore, using basic16 rather than dictionary8 provides a strong gain in resistance to brute force attacks at only a small usability cost.

These results reflect the fact that dictionary checks appear to contribute heavily to reducing usability by making password creation difficult. To some extent, this may be a result of our harsh check that uses a cracking dictionary and matches words of any length, rather than ignoring words of less than three or four letters, as may be more typical. We believe, however, that what makes any dictionary check valuable—preventing the use of common or predictable letter strings in passwords—also makes it inherently more difficult for users to think of a valid password. An interface that more clearly explains to users why their passwords are being rejected and provides suggestions for avoiding future rejections might help to reduce frustration.

Except in basic8survey, we also noted differences between participants who claimed to have written down their password, and those who claimed they had not: writing down passwords produced, on average, an extra 1.9 bits of entropy (statistically significant across 10 pairs of random samples of 1,224 participants each, Bonferroni corrected $p < 0.015$). This seems to suggest that, when using memorability aids, users produce stronger passwords; and could make the case for encouraging the use of password managers.

All these tradeoffs should be considered within the scope of the conditions we tested; future work testing other minimum lengths and combinations of required character classes might find more subtle variations.

Summary of Major Findings

Among conditions we tested, a 16-character minimum with no additional requirements provides the most entropy while proving more usable on many measures than the strongest alternative. Dictionary checks rule out the majority of pass-

words that can be easily cracked using heuristics, but should use large cracking dictionaries if they are to eliminate almost all vulnerable passwords. Unfortunately, dictionary checks also significantly increase user frustration.

Most participants write down or otherwise store their passwords. Interestingly, however, we find that storage is correlated with use of higher-entropy passwords.

We identified several common misperceptions in how password entropy is thought to be affected by users' password-composition strategies. 1) Adding numbers to passwords is thought to add little entropy; we found, by contrast, a lot of entropy in numbers. 2) Dictionary checks, although otherwise useful, add much less entropy than expected. 3) Unexpectedly, users typically create passwords that exceed minimum requirements, thus increasing password entropy.

We believe our results are immediately actionable for system administrators considering how to balance security and usability in their password-composition policies. Future work collecting and analyzing additional empirical data could provide even more detailed information about these tradeoffs.

ACKNOWLEDGMENTS

This research was supported by NSF grants DGE-0903659 and CCF-0424422, 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. Serge Egelman was affiliated with Brown University when this research began.

REFERENCES

1. A. Adams, M. A. Sasse, and P. Lunt. Making passwords secure and usable. In *HCI 97*, 1997.
2. M. Bishop and D. V. Klein. Improving system security via proactive password checking. *Computers & Security*, 14(3):233–249, 1995.
3. J. Bonneau and S. Preibusch. The password thicket: technical and market failures in human authentication on the web. In *Proc. (online) of WEIS'10*, June 2010.
4. W. E. Burr, D. F. Dodson, and W. T. Polk. Electronic authentication guideline. Technical report, NIST, 2006.
5. D. Florêncio and C. Herley. A large-scale study of web password habits. In *Proc. WWW'07*, 2007.
6. D. Florêncio and C. Herley. Where do security policies come from? In *Proc. SOUPS '10*, 2010.
7. P. Inglesant and M. A. Sasse. The true cost of unusable password policies: password use in the wild. In *Proc. ACM CHI'10*, pages 383–392, 2010.
8. B. Ives, K. R. Walsh, and H. Schneider. The domino effect of password reuse. *C. ACM*, 47(4):75–78, 2004.
9. J. L. Massey. Guessing and entropy. In *Proc. IEEE ISIT'94*, page 204, 1994.
10. G. Miller. Note on the bias of information estimates. *Info. Th. Psych.: Problems and Methods*, 1955.
11. L. Paninski. Estimation of entropy and mutual information. *Neural Comp.*, 15(6):1191–1253, 2003.
12. R. W. Proctor, M.-C. Lien, K.-P. L. Vu, E. E. Schultz, and G. Salvendy. Improving computer security for authentication of users: Influence of proactive password restrictions. *Behavior Res. Methods, Instruments, & Computers*, 34(2):163–169, 2002.
13. S. Schechter, C. Herley, and M. Mitzenmacher. Popularity is everything: A new approach to protecting passwords from statistical-guessing attacks. In *Proc. HotSec'10*, 2010.
14. C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Comp. Comm. Rev.*, 5(1), 1949.
15. C. E. Shannon. Prediction and entropy of printed english. *Bell Systems Tech. J.*, 30:50–64, 1951.
16. R. Shay and E. Bertino. A comprehensive simulation tool for the analysis of password policies. *Int. J. Info. Sec.*, 8(4):275–289, 2009.
17. R. Shay, S. Komanduri, P. Kelley, P. Leon, M. Mazurek, L. Bauer, N. Christin, and L. Cranor. Encountering stronger password requirements: user attitudes and behaviors. In *Proc. SOUPS'10*, 2010.
18. J. M. Stanton, K. R. Stam, P. Mastrangelo, and J. Jolton. Analysis of end user security behaviors. *Comp. & Security*, 24(2):124 – 133, 2005.
19. A. Vance. If your password is 123456, just make it hackme. New York Times, `http://www.nytimes.com/2010/01/21/technology/21password.html`, January 2010, retrieved September 2010.
20. K.-P. L. Vu, R. W. Proctor, A. Bhargav-Spantzel, B.-L. B. Tai, and J. Cook. Improving password security and memorability to protect personal and organizational information. *Int. J. of Human-Comp. Studies*, 65(8):744–757, 2007.
21. M. Weir, S. Aggarwal, M. Collins, and H. Stern. Testing metrics for password creation policies by attacking large sets of revealed passwords. In *Proceedings of the 17th ACM conference on Computer and communications security*, CCS '10, pages 162–175, New York, NY, USA, 2010. ACM.
22. Y. Zhang, F. Monrose, and M. K. Reiter. The security of modern password expiration: An algorithmic framework and empirical analysis. In *Proc. ACM CCS'10*, 2010.
23. M. Zviran and W. J. Haga. Password security: an empirical study. *J. Mgt. Info. Sys.*, 15(4):161–185, 1999.