Diversify to Survive: Making Passwords Stronger with Adaptive Policies

Sean M. Segreti, William Melicher, Saranga Komanduri, Darya Melicher, Richard Shay, Blase Ur[†], Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, Michelle L. Mazurek[‡] Carnegie Mellon University [†]University of Chicago [‡]University of Maryland
{ssegreti, billy, sarangak, rshay, lbauer, nicolasc, lorrie}@cmu.edu, darya@cs.cmu.edu, [†]blase@uchicago.edu, [‡]mmazurek@umd.edu

ABSTRACT

Password-composition policies are intended to increase resistance to guessing attacks by requiring certain features (e.g., a minimum length and the inclusion of a digit). Sadly, they often result in users' passwords exhibiting new, yet still predictable, patterns. In this paper, we investigate the usability and security of *adaptive* password-composition policies, which dynamically change password requirements over time as users create new passwords. We conduct a 2,619-participant between-subjects online experiment to evaluate the strength and usability of passwords created with two adaptive password policies. We also design and test a feedback system that guides users to successfully create a password conforming to these policies. We find that a well-configured, structure-based adaptive password policy can significantly increase password strength with little to no decrease in usability. We discuss how system administrators can use these results to improve password diversity.

1. INTRODUCTION

Reports of compromised password databases have become increasingly common in recent years [4, 12, 26, 35, 43, 50]. Such breaches can have far-reaching implications as they allow attackers to perform offline hash cracking attacks with virtually unlimited time. Because people commonly reuse passwords across accounts [11, 14], a breach of one account can compromise other accounts [14, 21]. While computationally expensive password-hashing functions are available, they are not always practical to implement or may be implemented ineffectively, and do not completely remove the existence of easy to exploit patterns. For high-value accounts, it remains imperative that users choose passwords that are hard for attackers to guess.

Password-composition policies, such as requiring a minimum length and inclusion of special characters, are commonly used to discourage users from choosing weak passwords. The usability and security of password-composition policies has been studied in depth [32, 45]; however, even under strict requirements, passwords still often have predictable patterns [23, 24, 45].

To increase a password set's resistance to guessing attacks, rather than focusing on the strength of individual passwords, researchers have proposed *adaptive* password-composition policies, which automatically evolve over time to encourage password diversity [34, 42]. For example, once some number of users of a given system have created passwords fitting a specific pattern, that pattern is banned and subsequent users may not create passwords fitting that pattern [34, 42]. While these proposed adaptive password systems may have strong potential benefits for security, their impact on password strength and usability has yet to be empirically tested.

In this work, we evaluate the security and usability impact of making password-composition policies adaptive. We focus on two implementations of this approach that do not require storing a copy of the plaintext (or reversibly encrypted plaintext) passwords, and which can operate with a traditional (non-adaptive) password policy. Adaptive policy systems that store plaintext or reversibly encrypted passwords are insecure in real-world situations, where one must assume attackers may gain access to the password store.

Our primary focus, Leininger et al.'s PathWell [33, 34], prohibits users from creating passwords with the same character-class *structure* (pattern of symbols, digits, and letters) as another user's password. When a new password is created, its structure is deemed "in use" and is not allowed during future password creation attempts. To increase the usability of the PathWell structure-based approach, we designed and tested a feedback system that guides users to choose a password with a permitted structure. The second approach, introduced by Schechter et al. [42], instead uses a specialized Bloom filter to probabilistically prevent users from creating passwords that are deemed too popular.

To evaluate the security and usability of these approaches, we conducted a two-part, between-subjects online study. 2,619 participants created a password under one of twelve conditions, designed to study: how adding adaptive requirements to traditional password policies affect security and usability; whether participants are confused by the extra requirements; how security and usability change as the stringency of adaptive policies increase; and the effect of graphical feedback.

We found that the passwords created under structure-based adaptive password-composition policies can be several or-

Symposium on Usable Privacy and Security (SOUPS) 2017, July 12–14, 2017, Santa Clara, California.

ders of magnitude more secure than those created without an adaptive password-composition policy. More surprisingly, we found that, structure-based adaptive policies can be applied without a significant usability cost, according to numerous usability metrics. We observed no statistically significant differences in creation time, password recall, or password storage (how often passwords were written down) between pairs of conditions that differed only in whether an adaptive policy was used. The only noteworthy usability downside of applying structure-based policies was that participants needed (on average 0.58–1.58) more attempts to create their passwords; however, this neither significantly impacted the overall time to create a password (of which a single attempt is a small fraction) nor affected user sentiment, except for the condition which simulated the most extreme numbers of disallowed structures. Our attempts to provide additional feedback to overcome the expected usability penalty of structure-based adaptive policies were largely superfluous; little usability had been lost to begin with.

2. BACKGROUND AND RELATED WORK

We first discuss the types of password-guessing attacks that adaptive policies aim to mitigate. We then detail the manner in which password-guessing approaches exploit common patterns in the absence of adaptive policies. Finally, we discuss related work on password-composition policies.

2.1 Password-Guessing Attacks

Password-guessing attacks fall broadly into one of two categories: those for which guessing is limited to a relatively small number of attempts, and those for which large-scale guessing is possible. An example of the former category is an online attack, in which an attacker submits password guesses to a running system. Because a well-configured system will have a policy that rate-limits guessing or locks accounts following a small number of incorrect authentication attempts, attackers are limited to making some of the most likely guesses. Measurements of fraudulent SSH login attempts revealed that some of the most common passwords that attackers guess are passwords often found in data breaches, as well as passwords related to system administration (e.g., variants of "root") [1]. If the adaptive system is bootstrapped, these types of common passwords could be banned initially by an adaptive policy. If not, the threat of password guessing is minimized to only the small number of accounts permitted to pick such a password before that password is banned.

Large-scale guessing attacks also present a major threat in a number of different circumstances. One such situation is an offline attack aimed at discovering credentials reused from other sites. If an attacker obtains a store of hashed passwords, which has become unfortunately common in recent years [4, 12, 26, 35, 43, 50], the attacker can perform offline hash cracking, limited only by his or her time and resources. Because users often reuse passwords across accounts [11,14], attackers can use the credentials obtained in an offline attack to compromise other accounts [14, 21].

Password-specific hash functions are designed to be computationally expensive in order to limit the number of guesses an attacker can make. Unfortunately, their deployment has proven error-prone in practice [25], is difficult to implement on some popular platforms that support backwards compatibility with legacy systems [38], and does not remove the existence of some easily exploitable patterns. Researchers have proposed systems to prevent offline cracking attacks [29], though these systems have yet to be deployed in practice and rely on having accurate models of generating artificial, yet plausibly human-chosen, passwords.

Even if system administrators were to follow all best practices to prevent offline password cracking for web accounts, other situations in which passwords are used would still be vulnerable to offline guessing. Encrypted file containers and full-disk encryption, as well as password stores from password managers (encrypted with a key derived from a master password), would remain vulnerable to offline guessing if an attacker gains access to the relevant file or device. Because adaptive schemes would not adapt over time in these single-user systems, these schemes could be bootstrapped with likely password patterns.

Despite the ability for system administrators to rate-limit online attacks and employ some technical mechanisms to minimize, but not completely eliminate, the threat of offline attacks, a user concerned about his or her high-value accounts is incentivized to practice defense in depth. Rather than relying exclusively on a system administrator to follow all best practices perfectly, which is far from guaranteed in practice, a user should choose unique passwords that are hard for attackers to guess. As we show in this paper, adaptive policies better enable users to do so.

2.2 Guessing Common Patterns

The types of common password characteristics adaptive policies aim to avoid can be exploited by password-cracking approaches. For example, Weir et al. proposed a probabilistic context-free grammar (PCFG) to model passwords [53]. Based on training data of previously observed passwords, PCFG assigns probabilities to both password structure (e.g., *princess111* has the structure {*8 letters*}{*3 digits*}) and constituent strings (e.g., "111"). Kelley et al. proposed improvements to this method [30], e.g., to treat uppercase and lowercase letters independently. Other researchers have advocated using grammatical structures and semantic tokens as non-terminals [40, 51]. Komanduri recently offered several PCFG improvements, including string tokenization and assigning probabilities to terminal strings not seen in training data [31]. The PCFG and its variants have been used in a number of prior studies to gauge password strength [10, 13, 16, 30, 36, 37, 45, 48]. Structure-based adaptive policies [34] make the PCFG approach less effective because the PCFG relies on the commonality of password structures to guess likely passwords.

Markov models also effectively model human-chosen passwords. Narayanan and Shmatikov first proposed using a Markov model of letters in natural language with finite automata representing password structures [39]. Castelluccia et al. used a similar method as part of their password meters [8]. Recently, Dürmuth et al. [15] and Ma et al. [36] evaluated the effectiveness of multiple variations of Markov models for cracking passwords, finding that Markov models were more accurate than PCFG at guessing passwords under certain circumstances. Popular password-cracking software

packages, such as John the Ripper[11] and Hashcat,[22] offer variants of a Markov model.

Adaptive policies have two conceptual advantages that aid in resisting guessing by Markov models. Structure-based adaptive policies encourage passwords with unpredictable structures, which are likely to foster character-level unpredictability that in turn may be hard to capture in a Markov model. Furthermore, string-based adaptive policies forbid the predictable passwords that a Markov model would easily guess.

Adaptive policies also provide conceptual difficulties for the guessing approaches of common password-cracking software tools. For example, the Hashcat toolkit implements a "mask attack," in which password guesses are generated by progressively exhausting the keyspace of each structure in an attacker-defined ordered list.[33] Despite its brute-force component, this approach can be effective in real-world cracking because many users craft passwords matching popular structures [41, 46].

2.3 Password-Composition Policies

Human-chosen secrets frequently share predictable, and thus exploitable, characteristics [5]. To discourage such patterns, organizations like the National Institute of Standards and Technology (NIST) recommend that system administrators employ password-composition policies, such as mandating a minimum length and the inclusion of a digit [7]. However, passwords created under these guidelines still frequently have exploitable patterns, such as consisting of a dictionary word followed by a number and symbol [37]. Furthermore, while some policies are better than others at balancing the tradeoff between leading users to create passwords that are harder to guess and improving the usability of password creation [45], particularly onerous password-composition requirements can unduly burden and annoy users [32].

Beyond requiring that passwords be at least a particular length and contain particular classes of characters, policies can also prohibit (blacklist) the most popular or predictable passwords. A judiciously chosen blacklist can lead users to pick passwords that are far harder to guess than those created without a blacklist [27, 32, 52]. String-based adaptive policies essentially build a blacklist that expands over time to reflect new password patterns.

Common patterns can make passwords easy to guess, yet they also can make them easy to remember. However, a recent study by Bonneau and Schechter showed that people are capable of remembering a large set of random characters if they are presented using spaced repetition [6]. Adaptive policies strive to capitalize on this discovery and, by introducing more complexity, to find a way to prompt users to create passwords with fewer exploitable patterns and thus higher resistance to guessing attacks.

3. METHODOLOGY

We conducted a two-part online study to examine how participants create and use passwords under two adaptive password policies in multiple configurations. In the first part of the study, we asked participants to create a password under a specific policy, take a survey, and then recall their password. Two days later, we asked participants to return and recall their password, in addition to completing a second survey. We recruited participants through Amazon's Mechanical Turk crowdsourcing service (MTurk). We required that participants be at least 18 years old and located in the United States. Our overall methodology is based on techniques used to compare password-composition policies in prior work [30, 32, 44, 45, 47, 48]. Our protocol was approved by our institution's IRB.

In *part one* of our study, we asked participants to imagine their main email account had been compromised, and they must create a new password. Prior work suggests that asking participants to imagine creating a password for their email account leads to stronger passwords than simply creating passwords for a study [30, 32]. We informed them that they would be asked to re-enter their password in a few days, and instructed them to do whatever they would normally do to remember and protect a new password.

We then showed participants one of twelve sets of password-creation instructions, depending on their assigned condition, described in Section 3.4. After creating a password, participants completed a survey on the password creation experience, as well as how they chose their password. We then asked participants to recall their password. Participants who typed their password incorrectly five times were then shown their password.

Two days later, we invited participants via email to return for *part two* of the study in which we asked participants to recall their password. After five incorrect attempts, participants were shown their password. Participants could follow a "Forgot Password" link to be emailed a link to their password. Next, we administered another survey about the steps the participant took to remember their password, including whether and how participants stored their passwords (e.g., writing it down or saving it electronically).

Our data-collection method enables us to measure several quantitative usability metrics during password creation and recall. We collect timing information and the number of password creation/recall failures. We use electronic copy-paste/autofill detection during the recall phase to augment the self-reported survey data. We also ask participants sentiment questions about the ease of both creating and recalling a password.

3.1 Adaptive Policies

We evaluated two adaptive password-policy systems. Both of these systems have two characteristics that we consider essential for an adaptive system. First, a secure system must not store passwords in plaintext or reversibly-encrypted ciphertext, as this creates a new avenue of attack. Second, an effective system must integrate with traditional password-composition policies for ease of deployment.

The first adaptive policy we evaluate operates on password structures, the password's sequence of character classes (uppercase, lowercase, digit, or symbol). We implement KoreLogic's Password Topology Histogram Wear-Leveling (PathWell) [33], which is designed to enforce password structure diversity, and refer to this system as the *structure-based ap-*

[11] 1 `www.openwall.com/john/`

[22] 2 `hashcat.net`

[33] 3 `hashcat.net/wiki/doku.php?id=mask_attack`

proach. In its simplest form, PathWell would require that all passwords in a set must have a unique structure. For example, if the password 'passWord11!' is in the set, then 'asdfQwer99#' would not be allowed for future passwords, because they both have the same character-class structure.

However, blacklisting a character-class structure after a single use could potentially help attackers by letting them know that, once they have successfully found a hash preimage, no other passwords in the set have the same character-class structure, obviating additional guesses within that structure. Additionally, it can decrease the usability of the system as more passwords are created. Therefore, PathWell also enables a structure to be blacklisted after some number of passwords use that structure, and this is the approach we use. PathWell's structure blacklist is designed to be preloaded with commonly used structures, and to grow over time as users are added to a system.

The second adaptive policy operates on passwords as a whole, rather than their character-class structure. In particular, we evaluate Microsoft Research's "Popularity is Everything" system, which prevents a given password from being used too many times in a system [42]. To do so, this approach uses a specialized database based on a Bloom filter [3] to record how many times a password is used without storing the password itself. As with PathWell, this database grows over time. We refer to this as the *string-based approach*.

3.2 Password-Composition Policies

To prevent users from creating passwords that are very weak or especially short (and vulnerable to brute-force attacks), adaptive policies should be used together with password composition requirements. Because early adopters of adaptive password policies in the real world are most likely organizations with high security needs, we chose to focus on stronger-than-average password-composition policies commonly employed in organizational and government settings, rather than for run-of-the-mill online accounts.

Historically, password-composition policies for higher security settings have mandated many different character classes. For example, in what we term the *4class8* (*4c8* for short) policy, passwords must contain at least eight characters, including all four character classes (lowercase letters, uppercase letters, digits, and symbols). This policy was recommended by NIST guidelines in 2011 [7] and once represented the *de facto* industry best practice. Such a policy is still popular, leading us to study it. Recently, however, password-composition policy guidance has begun to emphasize password length, rather than including character classes. Such a shift is evident both in the academic research literature [19, 45], as well as in the mass media [2]. U.S. government accounts have begun to deploy policies that emphasize password length [9].

We focus most of our experiments on a password-composition policy which we term *3class12* (*3c12* for short). This policy requires that passwords contain at least twelve characters, as well as three of the four character classes. This particular 3class12 policy has been found in prior work to better balance the security-usability tradeoff than policies like 4class8 [45] and is similar to policies in use on U.S. government systems [17].

Size	PCFG	# Structures 3c12	# Structures 4c8	# Passwords 3c12	# Passwords 4c8
M	10^5	2,141	2,236	1.62E56	4.64E28
L	10^6	8,940	—	1.65E56	—
XL	10^7	48,199	—	4.39E59	—

Table 1: Description of the blacklists we used. The first two columns show the how many password guesses were modeled in generating that blacklist. The remaining columns describe how many structures and passwords each blacklist disallows for the 3c12 and the 4c8 polices.

3.3 Systematically Testing Adaptive Policies

Evaluating an adaptive policy experimentally poses a unique challenge: if participants are working with an adaptive policy on a real-world system, each participant's password will modify the blacklist, thereby creating a unique environment for each participant. Instead, we opted to use pre-calculated blacklists of different sizes, allowing us to collect results from hundreds of participants exposed to the exact same situation. As such, we effectively compare passwords created at different points in the adaptive process, and those without an adaptive policy.

For this evaluation to succeed, it is critical to build blacklists that capture the most popular passwords. We gathered a total of 32,965,921 passwords from public leaks [22, 43, 50]. However, relatively few passwords from these sets meet the requirements of our stringent baseline policies (Section 3.4), limiting the size of the blacklists we could generate. To compensate, we trained a PCFG guesser [30] with these leaks and used it to enumerate the most probable guesses that conform to the minimum requirements. Using these guesses, we computed blacklists of the most common structures and passwords. This process simulates initial users in an adaptive system choosing highly probable passwords, with the corresponding structures subsequently being blacklisted. In Table 1, we summarize the blacklists we evaluated. Note that the M blacklist serves two purposes: Its corresponding 2,141 unique character-class structures are used to configure the structure-based adaptive approach; the 10^5 passwords used to generate the blacklist are the passwords that are banned in the string-based approach.

3.4 Conditions and Research Questions

We assigned participants to one of twelve conditions, each with different requirements, instructions, and feedback. Because of the large number of possible factors, it was not feasible to test all combinations of factors in isolation. Instead, we chose to run a set of conditions spanning five research questions (RQs) detailed below.

RQ1: Impact of Structure-Based Adaptive Policy *How are the usability and security of passwords affected when using a structure-based adaptive policy in addition to a traditional policy?* To answer this question, we evaluate two traditional password-composition policies: 3c12 and 4c8. We test each policy both with and without a medium-sized blacklist of character-class structures. The following conditions address our first research question:

- **3c12**: Passwords must contain at least 12 characters and include at least three character classes. This policy has been recommended in the academic literature [45]

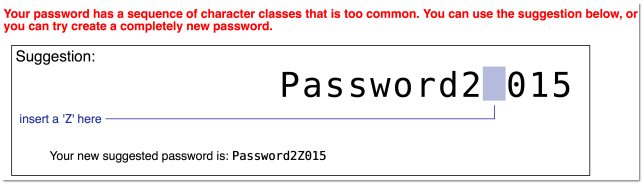


Figure 1: The interface shown to participants in standard structure-based blacklist conditions (Struct_M, Struct_L, Struct_XL, Struct_MIns, Struct_MSub, and Struct_M4c8) when their password was rejected by the structure blacklist. The suggested modification in this example was randomly chosen as insert a "Z" character. The Struct_M3Hint interface is similar, but instead shows three different suggested modifications.

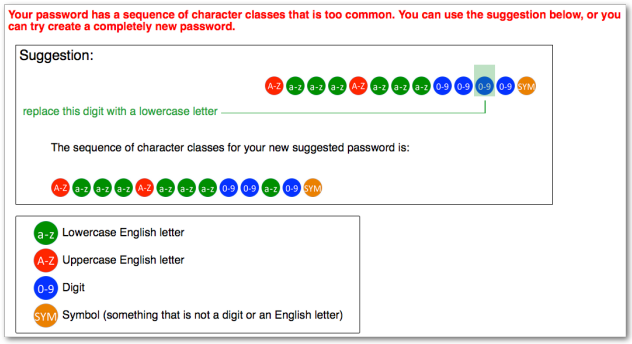


Figure 2: The interface shown to participants in Struct_MS when their password was rejected by the structure blacklist. The interface shows the character-class structure of the attempted password, rather than the password itself. The Struct_MSV interface was similar, except the structure was displayed in real time as it was typed. The Struct_MHyb interface was also similar, except only the inserted or substituted characters were obfuscated.

and is similar to policies for many U.S. government accounts [9] (e.g., [17]).

- **Struct_M**: The 3c12 policy plus a structure-based adaptive policy with 2,141 banned structures (corresponding to the first 10^5 PCFG guesses, as shown in Table 1).
- **4c8**: Passwords must contain at least 8 characters and include all four character classes. This policy was recommended by NIST guidelines in 2011 [7] and once represented best practice. Note that this condition is unique in that data for it was collected as part of an earlier study that used the same data-collection methodology. The condition does not contribute to any of our main results, but we include it here as an additional, informative baseline.
- **Struct_M4c8**: The 4c8 policy plus a structure-based adaptive policy with 2,236 banned structures, as shown in Table 1.

As part of this work, we designed and implemented a feedback system for the structure-based adaptive policies. If a participant's attempted password used a banned structure, the feedback system displayed a randomly selected modification (either insertion or substitution of a random character or character class) at a randomly selected location to their initial password such that the new password is guaranteed to have a legal structure. We informed participants that using the feedback was optional. For most conditions, we used a configuration of the structure-based adaptive policy feedback system that we expected to provide the most usability benefits, as shown in Figure 1.

RQ2: Structure-Based vs. String-Based Adaptiveness *What security and usability impact do structure-based and string-based adaptive policies have relative to each other, as well as relative to a non-adaptive policy?* To analyze this question, we studied only the higher security 3c12 policy in three versions: no adaptiveness; structure-based adaptiveness; and string-based adaptiveness [42]. However, the string-based adaptive policy does not include a feedback system. In contrast to a structure-based system, how to craft a minimally different password securely in a string-based system is non-obvious. The following three conditions address this research question:

- **3c12**: Previously introduced.
- **Struct_M**: Previously introduced.
- **String_M**: 3c12 with 10^5 passwords banned.

RQ3: Varying Blacklist Sizes *What are the usability and security consequences of different size blacklists?* As users create accounts on a structure-based adaptive system, more structures will be banned, potentially making password creation more frustrating for users whose desired structures are banned. As detailed in Section 3.3, we created blacklists of varying sizes to simulate different points in time in the adoption of an adaptive system, as the number of banned structures increases.

To measure the increasing difficulty of creating valid passwords in the presence of larger blacklists, as well as the theoretical security benefits of larger blacklists, we compared the following conditions. All are based on 3c12. Blacklist details are shown in Table 1.

- **3c12**: Previously introduced.
- **Struct_M**: Previously introduced.
- **Struct_L**: 3c12 with 8,940 banned structures (corresponding to banning 1,000,000 passwords).
- **Struct_XL**: 3c12 with 48,199 banned structures (corresponding to banning 10,000,000 passwords).

The remaining research questions involve different modes of feedback to the user when password creation fails.

RQ4: Number of Suggested Modifications *What are the usability and security consequences of presenting the user with more or fewer suggested modifications to their rejected password?* Whereas Struct_M shows one suggested modification, we also tested a condition that shows (at the same time) three suggested modifications. A participant could choose among the three hints, or choose a completely different password. We also evaluated a condition with hints disabled.

- **Struct_M**: Previously introduced (one hint).
- **Struct_M3Hint**: Identical to Struct_M, but with the feedback system showing three examples of possible modifications to the rejected password that lead to a permitted structure.
- **Struct_MNoHint**: Identical to Struct_M, but with no suggested modifications shown to the user.

RQ5: Insertion vs. Substitution Feedback Our standard method of suggesting modifications to a banned structure would either propose inserting a character or substitut-

ing a character. The standard implementation chose among those two possibilities with equal probability. *Does suggesting just insertions or just substitutions affect usability?*

- **Struct$_{\mathbf{M}}$**: Previously introduced.
- **Struct$_{\mathbf{M}}$Ins**: Identical to Struct$_{\mathrm{M}}$, but only offering feedback with suggestions of character insertions.
- **Struct$_{\mathbf{M}}$Sub**: Identical to Struct$_{\mathrm{M}}$, but only offering feedback with suggestions of character substitutions.

RQ6: Preventing Shoulder Surfing of Suggestions We speculate that suggesting modifications to banned structures could potentially aid in shoulder-surfing or screen-scraping attacks because both the rejected password itself and the suggested character to be inserted/substituted are displayed (see Figure 1). *Can the usability of the feedback system be preserved while limiting the potentially sensitive information shown on screen?* To answer this, we compare four conditions with different amounts of potentially sensitive information shown in the feedback interface.

- **Struct$_{\mathbf{M}}$**: Previously introduced.
- **Struct$_{\mathbf{M}}$Hyb**: Like Struct$_{\mathrm{M}}$ in that the rejected password is still shown on screen. However, suggestions instead relate to inserting or substituting a particular class of characters (e.g., a digit) rather than a specific character.
- **Struct$_{\mathbf{M}}$S**: Like Struct$_{\mathrm{M}}$Hyb, except all characters (rejected password and suggested modification) are replaced with a representation of their character class. See Figure 2.
- **Struct$_{\mathbf{M}}$SV**: Identical to Struct$_{\mathrm{M}}$S, except the interface displays the password's structure in real time as the participant types it, rather than only when a password is rejected. The intention was to help users understand the concept of character classes via a real-time example during creation.

3.5 Measuring Password Strength

To evaluate the passwords created in each condition, we analyze general password-composition characteristics, such as average length, inclusion of a variety of character classes, as well as password guessability [49], which models how many guesses a simulated attacker would make to guess a given fraction of a password set. To compute password guessability, we use the Password Guessability Service (PGS) in its recommended configuration (including the cracking methods: Probabilistic Context Free Grammars, Markov Models, Neural Networks, John the Ripper, and Hashcat), which combines several guessing attacks. This approach has previously been shown to be a conservative estimate of an expert in password forensics [49].

Modeling how an attacker would optimally attack a set of passwords created under an adaptive policy raises a number of subtle issues. In a structure-based adaptive policy configured such that a single usage causes a structure to be banned, successfully guessing a password with a particular structure implies that an attacker should avoid making additional guesses with the same structure. Similarly, if an attacker could somehow learn the list of used/banned structures, an attack could be refined by only attempting guesses with those structures. In our tests, we assume that blacklisted structures are unknown to the attacker, and that a structure is banned only after multiple passwords with the same structure are created, making it difficult for an attacker to determine that all passwords with a given structure were guessed and thus benefit from ceasing to make guesses with that structure. With this, we assume that sufficient rate-limiting and/or CAPTCHA solutions are implemented to prevent an attacker from abusing the password creation process to learn details about the adaptive policy's blacklist.

Because an attacker will not know at what point during the adaptive process a particular password was created, they will not be able to exclude potential guesses with particular character-class structures. Thus, we intentionally did *not* modify the computed PGS results to account for different blacklist sizes. Similarly, PCFG results that were used to create the blacklists may still be valid guesses for passwords created early in the adaptive process, yet the attacker does not know which passwords those are.

Hence, our guessability results compare the strength of passwords created earlier during the use of an adaptive policy (e.g., in Struct$_{\mathrm{M}}$) to those created later during the use of an adaptive policy (e.g., in Struct$_{\mathrm{L}}$) to those created without an adaptive policy (e.g., in 3c12).

3.6 Statistical Testing

For our usability metrics, we first performed omnibus statistical tests across all conditions. For omnibus comparisons, we use Kruskal-Wallis (KW) tests for quantitative data and Pearson's Chi-squared tests for categorical data. If the omnibus test was significant, we performed pairwise tests of pre-selected contrasts that correspond with each of our research questions. For pairwise comparisons, we use the Mann-Whitney U tests for quantitative data and Chi-squared tests (Fisher's Exact test when there are small bins) for categorical data. We use non-parametric statistical tests to avoid making assumptions about our data's distribution.

In particular, we made pairwise comparisons between each of the following groups of conditions: varying blacklist sizes (3c12, String$_{\mathrm{M}}$, Struct$_{\mathrm{M}}$, Struct$_{\mathrm{L}}$, Struct$_{\mathrm{XL}}$); the number of hints (Struct$_{\mathrm{M}}$NoHint, Struct$_{\mathrm{M}}$, Struct$_{\mathrm{M}}$3Hint); the type of suggestion (3c12, Struct$_{\mathrm{M}}$Ins, Struct$_{\mathrm{M}}$Sub, Struct$_{\mathrm{M}}$); stopping shoulder-surfing and screen-scraping (Struct$_{\mathrm{M}}$NoHint, Struct$_{\mathrm{M}}$, Struct$_{\mathrm{M}}$SV, Struct$_{\mathrm{M}}$S, Struct$_{\mathrm{M}}$Hyb). In each set, we compared each condition to all other conditions in that group. For all set of pairwise contrasts, we corrected for multiple testing using Holm-Bonferroni correction (HC).

For comparing the results of our simulated cracking attacks, we used a Log-Rank test, a statistical method used in survival analysis [28]. This test compares two guessing curves and takes into account whether a password was guessed, as well as at what point guessing stops. In this way, we can use all the data in our guessing curves for the statistical tests. All statistical tests use a significance level of $\alpha = .05$.

3.7 Limitations

Our methodology, which is similar to that employed by prior password research [30, 32, 45, 47, 48], has a number of limitations. By testing password recall once after a few minutes and once again after a few days, our study investigated password use that lies in between frequent and rare use. As such, we are not able to make strong statements about participants' ability to remember passwords in our study over long periods of time.

Condition	# Participants	Length	Uppercase	Lowercase	Digits	Symbols	Guessed (%)	Create difficult (%)	Create attempts	Create time (s)	Create confusing (%)	Recall attempts	Recall time (s)
3c12	163	13.9	1.6	7.9	3.1	1.3	49.1	28.2	1.50	47.9	9.82	1.80	26.2
String_{M}	216	13.6	1.7	7.4	3.2	1.3	46.8	26.4	1.50	48.5	6.94	1.82	23.3
Struct_{M}	213	14.1	1.8	7.5	3.5	1.3	25.4	31.5	2.08	50.4	9.39	1.57	25.3
Struct_{L}	159	14.1	1.6	7.3	3.8	1.4	17.6	43.4	2.51	48.4	19.5	1.86	26.6
$\text{Struct}_{\text{XL}}$	247	14.0	1.8	7.2	3.6	1.4	14.1	36.8	2.58	50.8	21.5	1.64	30.3
Struct_{M}3Hint	216	14.2	2.0	7.6	3.3	1.3	25.0	32.4	1.96	49.2	12.0	1.73	26.5
Struct_{M}NoHint	163	13.9	2.0	7.3	3.3	1.3	19.6	40.5	2.12	48.0	17.8	1.67	24.9
Struct_{M}Ins	207	14.4	1.9	7.9	3.3	1.4	23.1	32.9	1.97	54.3	13.5	1.78	27.0
Struct_{M}Sub	202	14.0	1.8	7.7	3.2	1.3	23.3	37.1	2.06	56.0	11.4	1.76	33.6
Struct_{M}Hyb	206	14.0	2.0	7.6	3.1	1.2	29.1	45.1	2.17	51.7	14.1	1.68	30.1
Struct_{M}S	209	13.8	2.1	7.0	3.6	1.3	31.1	33.0	2.00	54.3	12.9	1.90	28.2
Struct_{M}SV	204	14.0	1.9	6.9	3.7	1.5	25.5	32.4	1.88	58.5	14.7	1.98	28.3
Struct_{M}4c8	214	11.1	1.6	5.4	2.8	1.3	60.3	26.2	2.39	42.5	12.6	1.84	23.7

Table 2: Properties of passwords and study measurements, by condition. The second column shows participants who finished part two within three days.

Across our conditions, a relatively high number of participants did not return for part two. We excluded them from our analyses, except for analyzing the dropout rate as an indicator of dissatisfaction with a condition. Users who drop out of a study may behave differently than those who do not, potentially biasing our results.

All of the passwords in our study were collected for this study and were not used to protect real accounts, limiting ecological validity. In contrast to real-world, high-value passwords, study participants would not suffer consequences if they chose a weak password or forgot their password, nor were they incentivized to adopt their normal password behavior beyond our request that they do so. Two recent studies investigated the degree to which passwords collected for research studies resemble real, high-value accounts, and both concluded that passwords created during studies can resemble real, high-value passwords, yet are not a perfect proxy [18, 37].

While password-guessing approaches are most successful at modeling passwords given closely matched training data [30, 36], no major leaks of passwords contain passwords created under 3c12 and 4c8 policies. To compensate, we trained a probabilistic context-free grammar on the subset of passwords from large-scale leaks that fit those policies. We also used this grammar to model large numbers of likely passwords to create the blacklists. While having very large sets of real 3c12 and 4c8 passwords would have been strictly more accurate, no such sets are currently available to researchers.

For the reasons described in Section 3.5, we believe PGS models a reasonable attacker even for adaptive policies. Conceivably, however, some other strategy for ordering guesses against adaptive policies could prove to be more effective. That said, we are not currently aware of any such attack.

4. RESULTS

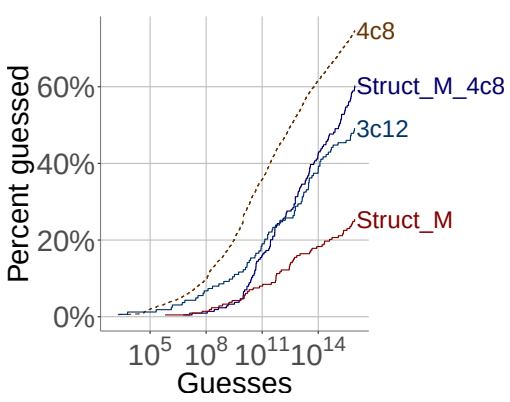
We find that an adaptive policy with a large blacklist dramatically increased the security of passwords. Surprisingly, this large increase in security is accompanied by only a small impact on usability. We tested numerous interface modifications to mitigate the decrease we expected in usability. In the absence of substantial usability decreases, however, these interface modifications have minimal impact on either security or usability. We detail general password characteristics by condition in Table 2; guessability in Figure 3; and usability in Table 3.

Participants received 55 cents for the first part of our study and 70 cents for the second. Of the 3,391 participants who began our study, 2,619 finished part one, 1,975 returned for part two within three days of receiving our invitation to return, and 1,799 finished part two of the study within three days of receiving that invitation. Other than the discussion of dropout rates, our analysis focuses only on the 1,799 participants who finished the entire study. Participants for whom we detect electronic copy-pasting from keystroke timing data almost without exception report that they wrote down their password in the survey, which suggests that participants truthfully disclosed rates of password storage. The number of participants per condition is shown in Table 2. 53% of participants reported being male, 46% female, and the remaining 1% declined to answer. Participants' mean age was 29 years (median 29).

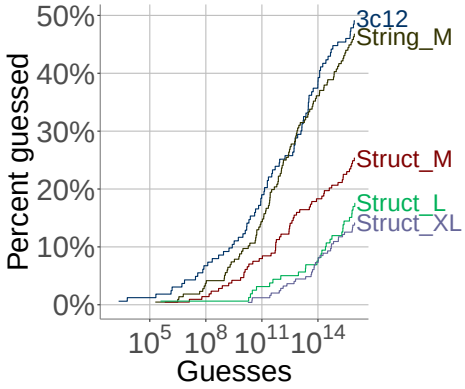
4.1 Impact of Structure-Based Adaptation

To examine the effect of implementing an adaptive policy, we compared 3c12 to Struct_{M} and 4c8 to Struct_{M}4c8. These two pairs each compare a password-composition policy with a structure-based adaptive blacklist to one without.

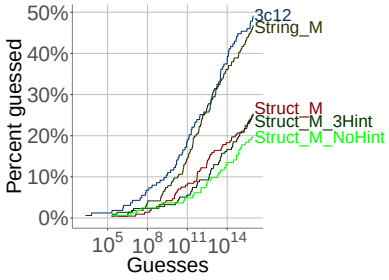
The inclusion of structure-based blacklists had a profound effect on security for both the 3c12 and 4c8 policies. As shown in Figure 3a, after 10^{16} guesses, PGS had correctly guessed roughly half as many passwords in Struct_{M} and Struct_{M}4c8 (with the adaptive policy) compared to 3c12 and 4c8 (without the adaptive policy), respectively. The difference between 3c12 and Struct_{M} is statistically significant (Log-Rank test, $X^2(1) = 23.9$, $p < 0.001$). Because the data for 4c8 was collected for a prior study, we did not perform statistical testing on that comparison.



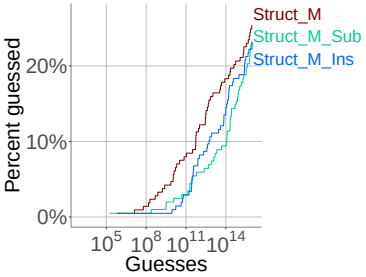
(a) Traditional policies and adaptive policies.



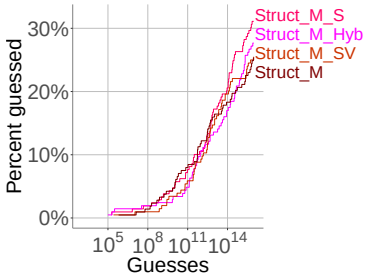
(b) Structure-based adaptive policy conditions with different blacklist sizes.



(c) Varying number of hints for the structure-based adaptive policy conditions.



(d) Varying suggestion type with structure-based adaptive policy.



(e) Conditions with interface changes to prevent shoulder-surfing.

Figure 3: The guessability of each password set. The x-axis shows the guess number (logarithmic scale). The y-axis shows the percent guessed at that guess number. Lines that are lower represent passwords that are more resistant against guessing attacks.

Along with our hypothesis that adaptive policies would result in more secure passwords, which was supported by our data, we also hypothesized that adaptive policies would result in decreased usability. Surprisingly, the structure-based approaches with medium-sized structure blacklists in Struct_M and Struct_M4c8 had only minimal impact on usability over 3c12 and 4c8.

We found no significant differences in our omnibus comparisons across all 12 conditions for these usability metrics. For instance, we did not find the inclusion of an adaptive policy to cause participants to perceive password creation as significantly more difficult or confusing. Similarly, we did not find the inclusion of an adaptive policy to make participants significantly more likely to store their passwords on paper or electronically. Nor did we find the inclusion of an adaptive policy to significantly impact the proportion of participants who were able to recall their password or how many attempts it took them to do so.

The only usability decrease that resulted from the structure-based adaptive policy with a medium-sized blacklist was increasing the number of attempts required to create a password. Specifically, participants required significantly more attempts to create compliant Struct_M passwords than 3c12 passwords (KW, H(1) = 22.9, $p < 0.001$) and Struct_M4c8 (KW, H(1) = 46.8, $p < 0.001$). That adaptive policies cause users to require more attempts to create a compliant password is unsurprising, though. By design, adaptive policies must reject candidate passwords to have any effect. Between the adaptive policies, we again observed differences in the number of attempts participants required to create a compliant password. For participants to create a compliant password, Struct_M4c8 required significantly more attempts than Struct_M (KW, H(1) = 5.48, $p < 0.019$). Struct_M4c8 also required significantly more time to submit a first attempt, whether compliant or not, than Struct_M (KW, H(1) = 7.34, $p < 0.020$).

4.2 Structure-Based vs. String-Based

Our experimental design allows for some limited comparison between the Struct_M and String_M approaches. Because optimal configurations for the Leininger et al. [33] and Schecter et al. [42] approaches have not yet been established, results of these comparisons should not be generalized beyond our particular configurations.

Under the configurations we tested in Struct_M and String_M, whose blacklists were built using the same source passwords, we find that Struct_M produces passwords roughly twice as difficult to guess as String_M (46.8% vs 25.4% cracked at cutoff, LogRank, $X^2(1) = 20.9$, p < 0.001), with similar usability results. In fact, the guessability of the string-based String_M did not differ significantly from 3c12 (49.1% vs 46.8% cracked at cutoff, LogRank, $X^2(1) = 0.431$, p < 0.735), which did not have an adaptive component.

Intuitively, the security improvement occurs because blacklisting a structure eliminates many potentially common passwords at once, whereas blacklisting a string eliminates only one. In terms of usability, a key factor is that it is trivial to quickly and automatically suggest a modified password

Agree password creation confusing *Omni.* χ^2_{13}=43.3, p<.001

cond.1	%	cond.2	%	χ^2_1	p-value
String$_{\text{M}}$	6.94	Struct$_{\text{L}}$	19.5	12.3	.003
		Struct$_{\text{XL}}$	21.5	18.2	<.001
Struct$_{\text{M}}$	9.39	Struct$_{\text{L}}$	19.5	7.03	.040
		Struct$_{\text{XL}}$	21.5	11.6	.005

Agree password creation difficult *Omni.* χ^2_{13}=39.1, p<.001

cond.1	%	cond.2	%	χ^2_1	p-value
String$_{\text{M}}$	26.4	Struct$_{\text{L}}$	43.4	11.1	.009
Struct$_{\text{M}}$	31.5	Struct$_{\text{M}}$Hyb	45.1	7.75	.032

Password creation attempts *Omni.* KW χ^2_{13}=143, p<.001

cond.1	mean	cond.2	mean	χ^2_1	p-value
3c12	1.50	Struct$_{\text{M}}$	2.08	22.9	<.001
		Struct$_{\text{M}}$4c8	2.39	46.8	<.001
String$_{\text{M}}$	1.50	Struct$_{\text{L}}$	2.51	33.3	<.001
		Struct$_{\text{XL}}$	2.58	68.0	<.001
Struct$_{\text{M}}$	2.08	3c12	1.50	22.9	<.001
		Struct$_{\text{M}}$4c8	2.39	5.48	.019
		Struct$_{\text{XL}}$	2.58	11.2	.004
Struct$_{\text{M}}$SV	1.88	Struct$_{\text{M}}$Hyb	2.17	6.96	.05

Password entry time during creation (s) *Omni.* KW χ^2_{13} =33.9, p=.001

cond.1	median	cond.2	median	χ^2_1	p-value
Struct$_{\text{M}}$	50.4	Struct$_{\text{M}}$4c8	42.5	7.34	.020

Password entry time during recall (s) *Omni.* χ^2_{13}=37.8, p<.001

cond.1	median	cond.2	median	χ^2_1	p-value
Struct$_{\text{XL}}$	30.3	String$_{\text{M}}$	23.3	11.5	.006

% Cracked (Log-Rank test)

cond.1	%	cond.2	%	χ^2_1	p-value
3c12	49.1	Struct$_{\text{M}}$	25.4	23.9	<.001
		Struct$_{\text{L}}$	17.6	42.7	<.001
		Struct$_{\text{XL}}$	14.1	73.3	<.001
		Struct$_{\text{M}}$3Hint	25.0	23.9	<.001
		Struct$_{\text{M}}$NoHint	19.6	27.9	<.001
String$_{\text{M}}$	46.8	Struct$_{\text{L}}$	17.6	38.9	<.001
		Struct$_{\text{M}}$	25.4	20.9	<.001
		Struct$_{\text{XL}}$	14.1	68.6	<.001
Struct$_{\text{M}}$	25.4	Struct$_{\text{M}}$4c8	28.5	16.2	<.001
		Struct$_{\text{XL}}$	14.1	11.2	<.003

Table 3: The statistically significant pairwise differences among our metrics.

that is close to the user's original attempt but still guaranteed to pass the structure check. Because any string-based password that is rejected is itself already a popular password, how one might automatically generate a minimally different, yet secure, password is non-obvious.

Although passwords created under Struct$_{\text{M}}$ were significantly more secure than those created under String$_{\text{M}}$, we did not observe significant differences between these two conditions for any of our usability metrics. As we describe later, however, we did find String$_{\text{M}}$ to have significant usability advantages over the structure-based policies configured with larger blacklists.

While more research comparing these approaches is necessary, our results suggest that a system administrator with access to a limited list of passwords with which to generate an initial blacklist should use a structure-based, rather than string-based, approach.

4.3 Varying Blacklist Sizes

Having found that implementing an adaptive system led to far more secure passwords while incurring minimal usability cost, we also explored how varying the size of the blacklists would impact security and usability. As we detailed in Section 3.4, these blacklists of different sizes should primarily be interpreted as proxies for different points in time during the life cycle of an adaptive policy, rather than configuration options. We also evaluated how these structure-based blacklists of different sizes compared to the medium size string-based blacklist. To do so, we compare the following four conditions: 3c12, String$_{\text{M}}$, Struct$_{\text{M}}$, Struct$_{\text{L}}$, Struct$_{\text{XL}}$.

The security of the passwords generally increased with the size of the blacklist, as shown in Figure 3b. Compared to 3c12, significantly fewer Struct$_{\text{L}}$ and Struct$_{\text{XL}}$ passwords were guessed (Log-Rank test, 3c12 vs Struct$_{\text{L}}$, $X^2(1) = 42.7$, $p < 0.001$; 3c12 vs Struct$_{\text{XL}}$, $X^2 = 73.3$, $p < 0.001$). Surprisingly, the guessability of Struct$_{\text{L}}$ and Struct$_{\text{XL}}$ did not differ significantly, suggesting that at structure blacklists of those sizes, the probability of a user creating a password with the next most common structure over any other permitted structure is very small.

Unsurprisingly, password creation generally required less effort in conditions with smaller blacklists. In essence, password creation becomes harder over time in an adaptive system. Struct$_{\text{M}}$ required significantly fewer creation attempts than Struct$_{\text{XL}}$ (KW, H(1) = 11.2, $p < 0.004$). In contrast, the time to create passwords on the first attempt did not differ significantly across conditions. This finding makes sense because participants in all conditions were shown the same text and interface during the first creation attempt.

Participants in conditions with smaller blacklists found password creation less difficult than those in conditions with larger blacklists. Participants in Struct$_{\text{M}}$ rated password creation as less confusing than participants in Struct$_{\text{L}}$ (Chi-squared, $X^2(1) = 7.03$, $p < 0.040$) or in Struct$_{\text{XL}}$ (Chi-squared, $X^2(1) = 11.6$ $p < 0.004$).

Despite these differences during password creation, we observed few differences across conditions in terms of password recall, suggesting that password memorability does not decrease significantly for users who create passwords later in the adaptive process. More precisely, the rate at which participants stored their passwords did not differ significantly across conditions (omnibus $X^2(13) = 16.3$, $p = 0.233$). The number of attempts participants required to recall their password also did not differ significantly across conditions (omnibus KW, H(13) = 9.21, $p = 0.757$).

All structure-based blacklists we tested resulted in more secure passwords than the string-based blacklist we tested.

That is, compared to $String_M$, fewer $Struct_M$, $Struct_L$, and $Struct_{XL}$ passwords were guessed (Log-Rank test, $String_M$ vs $Struct_M$, $X^2(1) = 20.9$, $p < 0.001$; $String_M$ vs $Struct_L$, $X^2(1) = 38.9$, $p < 0.001$; $String_M$ vs $Struct_{XL}$, $X^2(1) = 68.6$, $p < 0.001$).

Although it was less secure than the structure-based adaptive conditions, the string-based adaptive condition was generally more usable, requiring significantly fewer creation attempts than condition $Struct_L$ (KW, $H(1) = 44.0$, $p < 0.001$) or $Struct_{XL}$ (KW, $H(1) = 68.0$, $p < 0.001$). Participants in $String_M$ rated password creation as significantly less difficult (Chi-squared, $X^2(1) = 11.1$, $p < 0.008$) and less confusing (Chi-squared, $X^2(1) = 12.3$, $p < 0.003$) than participants in $Struct_L$. Participants in $String_M$ required less time to recall passwords than $Struct_{XL}$ participants (Chi-squared, $X^2(1) = 11.5$), $p = 0.006$) even though, as stated earlier, we did not observe significant differences in the memorability of those passwords.

4.4 Number of Suggested Modifications

We initially hypothesized that structure-based adaptive policies would cause a profound loss in usability. Therefore, we focused a number of conditions on the feedback given to users when their password was rejected. In those cases, the system would suggest modifications to the user's rejected password to make it compliant.

We tried varying the number of suggested modifications (one suggested modification versus three), as well as not suggesting any modifications. However, varying the number of suggested modifications did not have an impact on either security or usability.

In particular, we made pairwise comparisons across conditions $Struct_M$NoHint, $Struct_M$, and $Struct_M$3Hint. We did not observe statistically significant differences in the relative guessability of any of the following three condition pairs (Log-Rank test, $Struct_M$3Hint vs. $Struct_M$ $X^2(1) = 0.06$, p = 1.0; $Struct_M$NoHint vs. $Struct_M$ $X^2(1) = 1.887$, p = 0.678; $Struct_M$NoHint vs. $Struct_M$3Hint $X^2(1) = 1.415$, p = 0.703). Similarly, we did not find any pairwise comparisons for our usability metrics to have statistically significant differences.

We also calculated how many participants saw a hint, as well as how many accepted the hint's advice. In condition $Struct_M$, 75 of 213 participants saw at least one hint generated by the adaptive password policy during password creation, similar to the hint shown in Figure 1. Of those 75, slightly less than half (34) did not accept the advice shown in the hint and attempted to create an entirely new password, while the remaining 41 participants followed the guidance provided by the feedback.

4.5 Insertion vs. Substitution Feedback

We also examined the type of suggestions the adaptive system makes for rejected passwords. We compared $Struct_M$Ins, $Struct_M$Sub, and $Struct_M$, which respectively gave participants feedback that suggested either inserting a character, substituting a character, or one of the two (with equal probability). The locations of the character insertion/substitution suggestions were chosen randomly. We did not observe any significant differences in usability across these conditions.

4.6 Shoulder Surfing of Suggestions

As detailed in Section 3.4, we varied the suggested modifications in ways designed to minimize the information shown on screen, experimenting with showing structures instead of the actual password in either the suggestions, and as the user types their password. We expected that minimizing this information would decrease usability, yet would minimize the advantage to a shoulder-surfing adversary. To evalute this, we compared $Struct_M$, $Struct_M$Hyb, $Struct_M$S, and $Struct_M$SV. Because the extra information gleaned from shoulder surfing is not modeled in our guessability analyses, we did not expect to observe differences in guessability.

As expected, these conditions did not differ significantly in guessability. Surprisingly, though, we also observed minimal impact on usability. Although 45% of participants in $Struct_M$Hyb said creating a password was difficult, which was marginally higher than the proportion of participants $Struct_M$ (32%) who shared the same sentiment (Chi-squared, $X^2(1) = 7.75$, $p < 0.032$), we did not observe any other significant differences in usability.

5. DISCUSSION

Overall, we found that applying a structure-based adaptive policy to 3c12 was beneficial, substantially increasing security with a comparatively mild negative effect on usability. The effect on security was dramatic; about half as many passwords were cracked in condition $Struct_M$ as in 3c12. Surprisingly, although participants on average required more attempts to create compliant passwords in condition $Struct_M$ than 3c12, participants did not rate password creation as significantly more difficult. More importantly, the number of attempts required to recall their password, password entry time, and the fraction of participants who stored their password did not differ between conditions, suggesting that the structure-based adaptive policy does not negatively affect password memorability.

Varying the structure blacklist size, our proxy for an increase in the number of users of a given adaptive system, had profound effects on the security of passwords. As more users join the system and more structures are banned, new users are creating far more secure passwords than the initial users of the system. As expected, larger blacklists caused participants to require more creation attempts, yet this mostly did not increase participants' perceived difficulty of the task, in contrast to prior experiments (e.g., [45]). Interestingly, $Struct_{XL}$ had no security benefits over $Struct_L$, suggesting that the security benefits may have diminishing returns as the structure blacklist grows. Taking into account these diminishing returns, as well as the security disadvantages of blacklisting a structure after a single use (Section 3.5), we recommend blacklisting a structure only after multiple uses. Based on the diminishing returns of blacklisting structures, it could be beneficial to increase the number of uses before a structure is blacklisted as the number of passwords in the system increases. For systems with huge user bases (e.g. Google, Facebook, Twitter) this concept may become more important. We also suggest bootstrapping this system with the few thousand most common structures and letting the blacklist grow over time; this significantly increased resistance to guessing attacks with minimal usability sacrifices.

Neither varying the number nor removing hints altogether had a significant impact. Only about half of participants

who saw a hint (34 of 75) in condition Struct_M used the suggested password. This could be because participants felt they could make passwords that were more memorable, yet would still satisfy the requirements, or felt it would be more secure to use their own changes.

Based on prior work [20], we expected participants to find insertion suggestions more usable than substitution suggestions. However, we did not find this to be the case. At the same time, we found no significant differences with respect to the resistance of such passwords to guessing attacks.

A drawback of any password-creation feedback interface is that it could risk revealing information to attackers about the password through shoulder-surfing attacks. We hoped to minimize the impact of shoulder surfing by providing somewhat obfuscated feedback to participants. With minor exceptions, we found no significant differences according to our strength and usability metrics. As a result, we recommend the techniques used in Struct_MS, or Struct_MSV if real-time feedback is desired.

6. CONCLUSION

We evaluated string- and structure-based adaptive password policies, finding that adaptive policies provide significant security benefit with seemingly little usability cost, and should be considered for use in environments with large numbers of users. To balance usability and security, we recommend augmenting a strong password-composition policy with a structure-based adaptive system.

Surprisingly, the feedback system we tested did not improve usability as we had expected. Regardless of the type of feedback provided, participants made significantly stronger passwords with structure-based blacklists than without them, leading us to speculate that simply instructing participants who attempted to create blacklisted passwords to try to create a password with an uncommon sequence of character classes was sufficient; this should be investigated in future work. We find that obfuscating suggested passwords by their character-class representations, or not giving feedback at all, to be as usable as feedback approaches that are more vulnerable to shoulder surfing.

7. ACKNOWLEDGMENTS

This research was supported in part by PNC Financial Services Group, and by a grant from NATO through Carnegie Mellon CyLab. The authors acknowledge KoreLogic for technical assistance with PathWell.

8. REFERENCES

[1] Abdou, A., Barrera, D., and van Oorschot, P. What lies beneath? Analyzing automated SSH bruteforce attacks. In *Proc. Passwords* (2015).

[2] Barrett, B. 7 password experts on how to lock down your online security. *Wired*, 2016. https://www.wired.com/2016/05/password-tips-experts/.

[3] Bloom, B. H. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM 13*, 7 (July 1970), 422–426.

[4] Bonneau, J. The Gawker hack: How a million passwords were lost, 2010. https://www.lightbluetouchpaper.org/2010/12/15/the-gawker-hack-how-a-million-passwords-were-lost/.

[5] Bonneau, J. *Guessing human-chosen secrets*. PhD thesis, University of Cambridge, 2012.

[6] Bonneau, J., and Schechter, S. Towards reliable storage of 56-bit secrets in human memory. In *Proc. USENIX Security* (2014).

[7] Burr, W. E., Dodson, D. F., Newton, E. M., Perlner, R. A., Polk, W. T., Gupta, S., and Nabbus, E. A. Electronic authentication guideline. Tech. rep., NIST, 2011.

[8] Castelluccia, C., Durmuth, M., and Perito, D. Adaptive password-strength meters from Markov models. In *Proc. NDSS* (2012).

[9] Choong, Y.-Y., Theofanos, M., and Hung-Kung, L. United States Federal Employees' Password Management Behaviors. Tech. rep., NIST, 2014.

[10] Chou, H.-C., Lee, H.-C., Yu, H.-J., Lai, F.-P., Huang, K.-H., and Hsueh, C.-W. Password cracking based on learned patterns from disclosed passwords. *IJICIC* (2013).

[11] Das, A., Bonneau, J., Caesar, M., Borisov, N., and Wang, X. The Tangled Web of Password Reuse. In *Proc. NDSS* (2014).

[12] Davies, C. Millions of eHarmony passwords leaked. *Slash Gear*, 2012. https://www.slashgear.com/millions-of-eharmony-passwords-leaked-07232691/.

[13] Dell'Amico, M., Michiardi, P., and Roudier, Y. Password strength: An empirical analysis. In *Proc. INFOCOM* (2010).

[14] Duckett, C. Login duplication allows 20m Alibaba accounts to be attacked. *ZDNet*, 2016. http://www.zdnet.com/article/login-duplication-allows-20m-alibaba-accounts-to-be-attacked/.

[15] Dürmuth, M., Angelstorf, F., Castelluccia, C., Perito, D., and Chaabane, A. OMEN: Faster password guessing using an ordered markov enumerator. In *ESSoS*. 2015.

[16] Dürmuth, M., Chaabane, A., Perito, D., and Castelluccia, C. When privacy meets security: Leveraging personal information for password cracking. *CoRR* (2013).

[17] E2 Solutions. CW Government Travel, 2016.

[18] Fahl, S., Harbach, M., Acar, Y., and Smith, M. On the ecological validity of a password study. In *Proc. SOUPS* (2013).

[19] Florêncio, D., and Herley, C. Where do security policies come from? In *Proc. SOUPS* (2010).

[20] Forget, A., Chiasson, S., van Oorschot, P. C., and Biddle, R. Improving text passwords through persuasion. In *Proc. SOUPS* (2008).

[21] Geuss, M. Mozilla: Data stolen from hacked bug database was used to attack Firefox. *Ars Technica*, 2015. https://arstechnica.com/security/2015/09/mozilla-data-stolen-from-hacked-bug-database-was-used-to-attack-firefox/.

[22] Goodin, D. Hackers expose 453,000 credentials allegedly taken from Yahoo service. *Ars Technica*, 2012. http://arstechnica.com/security/2012/07/yahoo-service-hacked/.

[23] Goodin, D. Why passwords have never been weaker-and crackers have never been stronger. *Ars*

Technica, 2012. https://arstechnica.com/security/2012/08/passwords-under-assault/.

[24] Goodin, D. "thereisnofatebutwhatwemake"-turbo-charged cracking comes to long passwords. *Ars Technica*, 2013. https://arstechnica.com/security/2013/08/thereisnofatebutwhatwemake-turbo-charged-cracking-comes-to-long-passwords/.

[25] Goodin, D. Once seen as bulletproof, 11+ million Ashley Madison passwords already cracked. *Ars Technica*, 2015. https://arstechnica.com/security/2015/09/once-seen-as-bulletproof-11-million-ashley-madison-passwords-already-cracked/.

[26] Graham, R. Notes on the Ashley-Madison dump, 2015. http://blog.erratasec.com/2015/08/notes-on-ashley-madison-dump.html.

[27] Habib, H., Colnago, J., Melicher, W., Ur, B., Segreti, S., Bauer, L., Christin, N., and Cranor, L. F. Password creation in the presence of blacklists. In *Proc. USEC* (2017).

[28] Harrington, D. P., and Fleming, T. R. A class of rank test procedures for censored survival data. *Biometrika 69*, 3 (1982), 553–566.

[29] Juels, A., and Rivest, R. L. Honeywords: Making password-cracking detectable. In *Proc. CCS* (2013).

[30] 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 *Proc. IEEE SP* (2012).

[31] Komanduri, S. *Modeling the adversary to evaluate password strengh with limited samples*. PhD thesis, CMU, 2015.

[32] Komanduri, S., Shay, R., Kelley, P. G., Mazurek, M. L., Bauer, L., Christin, N., Cranor, L. F., and Egelman, S. Of passwords and people: measuring the effect of password-composition policies. In *Proc. CHI* (2011).

[33] KoreLogic. libpathwell: Pam module and library for auditing/enforcing password topology histogram wear-leveling, 2015. https://github.com/KoreLogicSecurity/libpathwell.

[34] Leininger, H. LibPathWell 0.6.1 Released, 2015. https://blog.korelogic.com/blog/2015/07/31/libpathwell-0_6_1.

[35] Love, D. Apple on iCloud breach: It's not our fault hackers guessed celebrity passwords. *International Business Times*, 2014. http://www.ibtimes.com/apple-icloud-breach-its-not-our-fault-hackers-guessed-celebrity-passwords-1676268.

[36] Ma, J., Yang, W., Luo, M., and Li, N. A study of probabilistic password models. In *Proc. IEEE SP* (2014).

[37] 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 *Proc. CCS* (2013).

[38] Microsoft. Microsoft SMB Protocol and CIFS Protocol Overview.

[39] Narayanan, A., and Shmatikov, V. Fast dictionary attacks on passwords using time-space tradeoff. In *Proc. CCS* (2005).

[40] Rao, A., Jha, B., and Kini, G. Effect of grammar on security of long passwords. In *Proc. CODASPY* (2013).

[41] Redman, R. PathWell Topologies, 2014. https://blog.korelogic.com/blog/2014/04/04/pathwell_topologies.

[42] Schechter, S., Herley, C., and Mitzenmacher, M. Popularity is everything: A new approach to protecting passwords from statistical-guessing attacks. In *Proc. HotSec* (2010).

[43] Schneier, B. Myspace passwords aren't so dumb. *Wired*, 2006. http://archive.wired.com/politics/security/commentary/securitymatters/2006/12/72300.

[44] Shay, R., Kelley, P. G., Komanduri, S., Mazurek, M. L., Ur, B., Vidas, T., Bauer, L., Christin, N., and Cranor, L. F. Correct horse battery staple: Exploring the usability of system-assigned passphrases. In *Proc. SOUPS* (2012).

[45] 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 *Proc. CHI* (2014).

[46] Trustwave. 2014 global security report. https://www.trustwave.com/Resources/Library/Documents/2014-Trustwave-Global-Security-Report.

[47] Ur, B., Alfieri, F., Aung, M., Bauer, L., Christin, N., Colnago, J., Cranor, L. F., Dixon, H., Naeini, P. E., Habib, H., Johnson, N., and Melicher, W. Design and evaluation of a data-driven password meter. In *Proc. CHI* (2017).

[48] 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 *Proc. USENIX Security* (2012).

[49] Ur, B., Segreti, S. M., Bauer, L., Christin, N., Cranor, L. F., Komanduri, S., Kurilova, D., Mazurek, M. L., Melicher, W., and Shay, R. Measuring real-world accuracies and biases in modeling password guessability. In *Proc. USENIX Security* (2015).

[50] Vance, A. If your password is 123456, just make it HackMe. *The New York Times*, 2010. http://www.nytimes.com/2010/01/21/technology/21password.html?mcubz=1.

[51] Veras, R., Collins, C., and Thorpe, J. On the semantic patterns of passwords and their security impact. In *Proc. NDSS* (2014).

[52] Weir, M., Aggarwal, S., Collins, M., and Stern, H. Testing metrics for password creation policies by attacking large sets of revealed passwords. In *Proc. CCS* (2010).

[53] Weir, M., Aggarwal, S., de Medeiros, B., and Glodek, B. Password cracking using probabilistic context-free grammars. In *Proc. IEEE SP* (2009).