Modeling Aggregate Security with User Agents that Employ Password Memorization Techniques

Christopher Novak
Department of Computer Science
Dartmouth College
novak.chris.m@gmail.com

Jim Blythe
Information Sciences Institute
University of Southern California
blythe@isi.edu

Ross Koppel
Department of Sociology
University of Pennsylvania
rkoppel@sas.penn.edu

Vijay Kothari
Department of Computer Science
Dartmouth College
vijayk@cs.dartmouth.edu

Sean Smith
Department of Computer Science
Dartmouth College
sws@cs.dartmouth.edu

ABSTRACT

We discuss our ongoing work with an agent-based password simulation which models how site-enforced password requirements affect aggregate security when people interact with multiple authentication systems. We model two password memorization techniques: passphrase generation and spaced repetition. Our simulation suggests system-generated passphrases lead to lower aggregate security across services that enforce even moderate password requirements. Furthermore, allowing users to expand their password length over time via spaced repetition increases aggregate security.

1. INTRODUCTION

Password authentication is often a critical, if not the sole, authentication mechanism in existing systems. Unfortunately, it places significant burden on users to perform the often overwhelming task of memorizing and recalling username and password strings. Indeed, the mantra that "practice makes perfect" affirms that most of us cannot instantly memorize everything. In the same vein, users of authentication systems are only willing to expend so much effort to comply with security and privacy rules (e.g. [2, 11]). When users circumvent policies, security suffers. In one of many examples we found, an organization required employees to change passwords every 90 days, which led some employees to reset their password to the previous one on the 91st day [5].

To provide a framework to analyze these effects quantitatively, in both this work and prior work, our team has been using the agent-based modeling platform DASH (Deter Agents Simulating Humans) to build models of how users create and manage passwords for multiple online accounts. *Cognitive burden* can be a useful tool for representing mental workload in computational models of human behavior; in our most recent prior work, we used heavy cognitive burden as a means of explaining why users often engage in risky behaviors, namely writing down and reusing passwords [19].

Committing unique, strong passwords to memory is generally a task that involves heavy cognitive burden. However, some security analysts have recently pointed out that high password strength and memorability may not be mutually exclusive [6]. By employing specific password coping and memorization techniques, users might be able to increase the number of unique, strong passwords they can remember.

In this paper, we examine how two techniques for effectively memorizing secure passwords, *spaced repetition* and *passphrase generation*, affect *aggregate security*, a security measure that accounts for the fact that people use multiple authentication systems—and not in isolation.

Spaced repetition refers to a learning technique that involves exposure to previously learned material over increasing intervals of time. We apply this technique to an existing password simulation to model how an authentication system might allow users to learn a shorter initial password and lengthen it over time. As has been shown by several security researchers, committing passwords to memory significantly increases the user's cognitive burden [16, 22]. Spaced repetition can be combined with *password lengthening*, which refers to the process of extending a base password over time after successful recalls. By limiting the original password length to something short, the user can theoretically expend far less effort than if she had originally attempted to memorize a longer password with higher initial strength. As the user strengthens her knowledge of the password, she can gradually increase its length and/or complexity with comparatively little cognitive burden. Joseph Bonneau and Stuart Schechter tested this technique with a group of 223 participants on Amazon's Mechanical Turk, finding that 94% were able to successfully commit a 56-bit password to memory over the course of two weeks [6]. We note that spaced repetition can be applied to any scenario that requires memorization; here, we examine spaced repetition as it applies to the process of lengthening a base password. (Bonneau and

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

Schechter refer to this process as "spaced repetition" alone.)

Passphrase generation refers to the creation of long passwords composed of several valid English words. By concatenating words together to form a long password, the strength of the password is relatively high, but the words might be easier to memorize than a password of similar strength with a variety of character types and few full words. In theory, this technique could be used to generate "secure" passwords with significantly less cognitive burden to the user.

Modeling these two techniques with DASH enables us to quickly approximate real-world behavior and measure aggregate security without incurring the cost of implementing policy changes in practice. This could benefit security practitioners attempting to design authentication and security policies to improve aggregate security.

2. RELATED WORK

Many papers have documented strategies that users employ to cope with cumbersome password policies, including recording passwords on physical and digital media, sharing passwords with family members, and reusing passwords across services; estimated the fraction of users who use these strategies; and studied the security repercussions of non-compliance, e.g. [14, 17, 25, 10, 5, 7]. Since non-compliance makes password policies ineffectual, these observations have spurred interest in understanding the factors that drive users toward circumvention, e.g. [30, 20].

Both new and existing models have been used to explain why and when users engage in password coping strategies, e.g. [2, 11]. Simulations have been created to measure the efficacy of different password composition policies in light user circumvention, e.g. [27, 1, 24]. Indeed, this paper, which is a continuation of earlier work [21, 19], falls under this category. Our ultimate aim is to accurately predict the aggregate security afforded by a set of password policies by simulating user agents who create and cope with passwords and password policies. We believe our simulation distinguishes itself by the depth of the user agent, which we hope will allow for greater productive power, but may require more extensive validation. This paper expands on our earlier work by comparing two password memorization techniques in simulation.

3. AGENT MODELS OF PASSWORD BEHAVIOR

Agent-based simulation can be used as a complement to empirical studies of human behavior in exploring the consequences of password policies and human responses to them. While empirical studies are essential to understanding the range of individual behaviors around passwords, they are costly and have natural limits in scope and size. Simulations cannot replace empirical work but can indicate the consequences of those behaviors in larger groups and in a wider range of circumstances. For example, given a distribution of coping strategies such as resetting passwords and sharing passwords between sites, one can examine how the prevalence of these strategies changes in response to stricter policies. Simulations used in this way are subject to the same issues of generalizability as other methods, and care must be taken in making decisions based on their results. This includes (1) making use of recent applicable empirical work and understanding its generalizability, (2) analyzing the sensitivity of the predictions to any assumptions that are made in the simulation and (3) verifying that the emergent properties of the simulation align with available observed values.

In our work we build on the DASH model for cognitive agents [4] and on previous simulations of password behavior built with DASH [19]. Agents built using DASH have access to a dual-process model of reasoning, goal-directed behavior that is responsive to changes in the agent's world and recall affected by spreading activation. These capabilities are grounded in empirical work in cognitive science and help provide an account of cognitive burden and password recall for our agent model of password use. The work described here builds on a previous password agent model that included an explicit model of cognitive burden, a finite amount of memory for passwords, and fixed coping strategies including reusing passwords between sites and writing passwords down. Agents employed these strategies when memorization of a set of passwords exceeded the agent's memory limit. We showed that this simulation provided behavior consistent with observations including those on password reuse and the proportion of reset attempts from Florencio and Herley [10]. In this paper we extend this work to consider passphrase generation and memorizing passwords through spaced repetition.

4. SYSTEM-GENERATED PASSPHRASES

In 2011, xkcd published its now well-known "Correct Horse Battery Staple" cartoon [23], prompting public interest and research about passphrases, which are composed of concatenated valid English words. Shay et al. found that users forgot system-generated passphrases and system-assigned passwords at similar rates and that although entropy was similar for both, users had to spend more time typing in passphrases [28]. However, these types of studies on password memorability and usability are often problematic in that they isolate the experiment from the everyday lives of the participants. On average, users are tasked with memorizing approximately 25 passwords [10], and the cognitive burden of doing so drives users to write them down or reuse them [5]. Agent-based simulations may thus be a more realistic approximation for how users might actually benefit from system-generated passphrases than typical studies might suggest.

Implementation

To simulate the use of system-generated passphrases over traditional user-defined passwords, we extended our password simulation (built on the DASH agent modeling platform [19, 21]). In our original model, we modeled human cognitive burden as the minimal cost of a tree that spans the set of known passwords plus the empty string, where the edge weight is given by Levenshtein distance (i.e. "edit distance."). Such a model for cognitive burden is useful in accounting for small permutations of passwords for different sites. Take, for example, the passwords `apxxqwoinwe$a1` and `apxxqwoinwe$a2`. Certainly, the cost of remembering both passwords is only slightly more than remembering one, and the Levenshtein distance captures this fact. However, consider too the passwords `feltcoldchair1` and `feltcoldchair2`. Certainly, the latter two passwords would be easier to memorize than the former ones, but both password tuples

have the same Levenshtein distance.

To handle system-generated passphrases, we created a new cognitive burden cost function to examine the composition of the password rather than the relationships between passwords, as the passwords themselves are unlikely to be related with a sufficiently large dictionary. We model the *word count* cost of memorizing a set of passwords $P = \{p_1, p_2, \ldots, p_n\}$ as $WC(P) = \sum_{p \in P} WC(p)$, where $WC(p)$ is the number of valid English-language words in password p. Special characters and digits are characterized as individual words, though our system-generated passphrases do not contain either type of character. To determine if a substring in a password constitutes an English-language word, we use the NLTK English-word corpus [3] to validate the longest substrings that compose individual words of the system-generated passphrase. We use the Viterbi algorithm [13] to search for maximal length valid word substrings in $O(n)$ time rather than the $O(n^2)$ time of a brute-force search. We tested a new set of passphrases against our original password set using Levenshtein distance, word count, and an average of the two to examine how passphrases affect aggregate security.

5. SPACED REPETITION OF PASSWORDS

Memorization via spaced repetition involves increasing time intervals between satisfactory memorization of content. On the other hand, *massed repetition* involves repeated memorization of content over a short period of time. There are many theories surrounding the effectiveness of spaced repetition over massed repetition. One acknowledges the importance of active knowledge retrieval in strengthening memory, and another posits that the learner's contextual surroundings are more variable than with massed repetition, increasing the number of opportunities for recall cues [9]. In the previously cited Bonneau and Schechter study, nearly all participants were able to remember a 56.4-bit secret after a median of 36 logins by employing spaced repetition [6]. Much of the research that surrounds passwords and authentication fails to acknowledge the importance of time and repetition in knowledge recall, and so one of our primary aims here is to better model these factors.

Implementation

Existing agent-based models for authentication often incorporate the effect of repetition on memory. In our previous password simulation, users who successfully log in to a system with a given password increase their belief in the correctness of that password while also reducing their beliefs in the correctness of all other passwords [21, 19]. However, we wanted to model how an authentication system might be able to combine this technique with password lengthening, which allows users to memorize an n-length password and slowly expand its length over time. In the simulation, the user may lengthen her password only a set number of times (a parameter whose default is three), and each expansion involves increasing the length of the password by four.

To account for this type of authentication system, we incorporated an *early attack risk* into our model to account for the possibility that an attacker can successfully learn a password before it has reached its longest length—and correspondingly, highest possible strength.

6. EXPERIMENTAL DESIGN

To test the effectiveness of passphrase generation and spaced repetition in our simulation, we defined a security measure M to be the probability that a given service is safe from attack, where an attack can be a brute-force attack (here, an offline dictionary attack), an attacker's discovery of a written password, or an indirect reuse attack wherein an attacker uses an agent's username and password from one site to log in to another site. The probability of a brute-force attack on a password p is $bf(p)$, where

$$bf(p) = \begin{cases} \frac{t \times g}{\mathrm{k(p)}}, & \text{if } t \times \mathrm{g} < k(p) \\ 1, & \text{otherwise} \end{cases}$$

Here, t denotes the attack time in seconds, g denotes the number of guesses per second, and $k(p)$ denotes the size of the keyspace for password p.

We assume that the attacker has the capability to guess 100 million times per second, which is achievable for even a remotely knowledgeable attacker without expensive hardware (e.g. [15]). Further, we assume that the attacker spends no longer than an hour attempting to crack the password. In other words, we do not model a strong attack. To approximate the size of the keyspace for a given password p, we use Passfault, an open-source tool for measuring password complexity. Passfault computes the smallest search space as the complexity of the password, using factors like dictionary matches, misspellings, substitutions, keyboard patterns, and string repetitions to make its determination [26].

Given a specific user and service, the probability of an attacker's compromise of the user's written-down password is set to the *stolen attack risk* parameter if the password has been written down, and zero otherwise. We performed sensitivity analysis of this parameter (among others) in [21]. Finally, the probability of a reuse attack on a specific service is equal to one minus the probability of the service's safety from a reuse attack, where the probability of safety from a reuse attack for a service S is a product of probabilities, where each probability is the probability that S is safe from a reuse attack that stems from service S'. The probability that S is safe from a reuse attack that stems from S' is equal to the *reuse attack risk* (also explored in [21]) times the probability that S' is safe from a direct attack, where the direct attack can be a brute force or written password attack. We then averaged M across all services.

Our last simulation used a set of approximately thirty passwords of increasing complexity. These passwords were not totally realistic but provide useful guides for analysis of aggregate security. As an example, we used `P@SsWoRd12` and `MyPaSsW0Rd!234?`. The passwords were all in some way related; we used insertions and substitutions to make passwords slightly more complex. These passwords made sense for our initial iteration of the simulation, as they captured the fact that people often use slightly permuted forms of a *base password* across sites [29], resulting in a lower total cognitive burden than if they had used totally distinct passwords for each site. However, we wanted to increase the diversity of passwords to account for different password memorization strategies that users might employ.

To test passphrase generation, we used a newer version of the

password simulation built with Python. We tested two distinct password sets from which the user agents may choose passwords: the original, permuted password set as described in the previous paragraph, and a new password set composed of system-generated passphrases. For each password set used, we tested three different cognitive burden functions: one that uses Levenshtein distance alone, another that uses word count alone (as described in Section 2), and a third that uses both Levenshtein distance and word count.

Spaced repetition relies on the fact that as people successfully commit concepts or words to memory, they become easier to recall over time. The previous iteration of our password simulation did account for the fact that users are more likely to successfully recall passwords with each successful login, but the cognitive burden function used did not take into account the fact that eventually the cognitive cost of remembering a relatively well-known password decreases. We changed our password simulation to allow agents to practice spaced repetition of passwords as described in a real-word study [6]. After an agent's belief in a given password reaches the *expansion threshold* (a new model parameter that we define), she can extend the length of her password by a preset number of randomly generated characters. To reflect the fact that the cognitive burden of remembering a password approaches zero after a long period of successful recalls, we include only the cost of the new characters in the cognitive burden function. However, the likelihood of a brute-force attack is higher than if the agent had opted to use the expanded-length password from the outset, as an attacker could more easily learn the shorter password; if the user lengthens the shorter password, it is nearly trivial to compute the new password (given that the lengthening adds only a few characters, as is the case in our simulation and Bonneau and Schechter's study). For the cognitive burden function, we used the average of Levenshtein distance and word count. We opted to use the original password set.

7. PRELIMINARY RESULTS

For each experiment run, we tested 9 service *hardnesses*, where each each hardness is an additive factor to a base minimum password length enforced by the service. Each service can be either "weak", "average", or "strong", and there is an equal distribution of service types. (Here, we use three of each respective type.) "Weak", "average", and "strong" services enforce minimum password lengths of $h+1$, $h+5$, and $h+8$, respectively, where h is the hardness variable. Password length is defined as the number of characters in the password.

7.1 System-Generated Passphrases

In this section, we explore the efficacy of system-generated passphrases over standard passwords, where we define a system-generated passphrase to be a string composed of concatenated valid English words, and a standard password to be a password drawn from our original password set, which is composed of passwords like `MyS3cUReP@SsW0rd!2345` and `P@SsWord12`. Although password length is the only requirement that services enforce, our simulation considers password and passphrase strength in the computation of direct attack risk.

Levenshtein Distance Cognitive Burden Function

We first used Levenshtein distance to model the user's cognitive burden, which is the same metric that we used in earlier iterations of the simulation. While system-generated passphrases offer greater aggregate security when password requirements are relatively lax, their relative utility declines with stricter password requirements. Figure 1 displays the relationship.

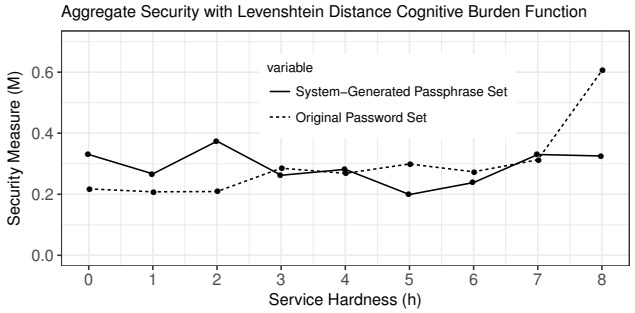

Figure 1: Levenshtein Distance Cognitive Burden Function vs. Aggregate Security

Word Count Cognitive Burden Function

As a proxy for cognitive burden, Levenshtein distance does not take into account the fact that words are easier to remember than strings of random characters. We modified our simulation to compute cognitive burden by summing over the word count for each password across the password set. Again, we found that system-generated passphrases offer greater aggregate security when password requirements are relatively lax, but decline in utility as password requirements increase. We found this result somewhat surprising, as the modified cognitive burden function assigns a lower cost to passwords composed of concatenated words than those of random characters.

We believe that this result can be explained by the fact that as password length requirements increase, system-generated passphrases become exponentially easier to crack than standard passwords, outweighing the benefit to cognitive burden that results from using passphrases over standard passwords. Figure 2 displays the relationship.

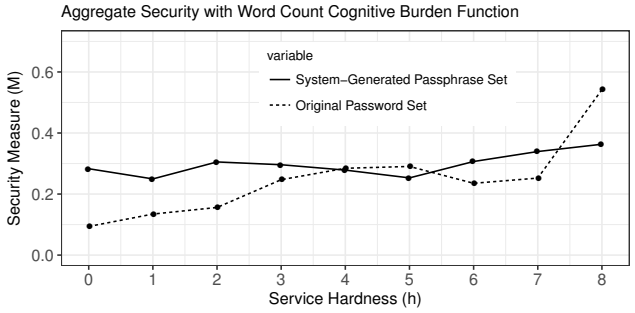

Figure 2: Word Count Cognitive Burden Function vs. Aggregate Security

Averaged Levenshtein Distance and Word Count Cognitive Burden Function

For our last experiment on system-generated passphrases, we modified our cognitive burden function to take both Levenshtein distance and word count into account. Specifically, we computed cognitive burden to be the average of the two. Our results show that system-generated passphrases perform significantly worse than the original password set that we used. Figure 3 displays the relationship.

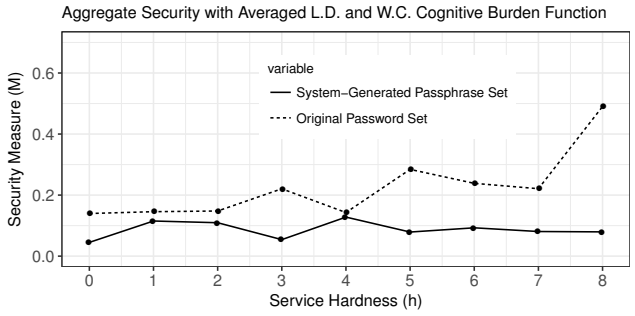

Figure 3: Averaged Levenshtein Distance and Word Count Cognitive Burden Function vs. Aggregate Security

7.2 Spaced Repetition

To model the spaced repetition approach to password memorization that Bonneau and Schechter describe [6], we ran the simulation using our original password set and a cognitive burden function that averaged Levenshtein distance and word count of the password set. Figure 4 shows the results of our trials.

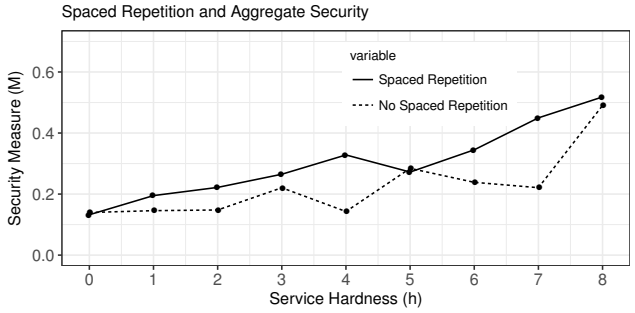

Figure 4: Spaced Repetition and Aggregate Security

In general, our trials showed that incorporating spaced repetition into the authentication scheme generally improved aggregate security. While it may seem self-evident that lengthening passwords generally increases their resistance to brute force attacks, the risk of an early attack must be accounted for. If a password p is discovered by an attacker before the user lengthens it to p' (where $p' = p$ plus a short string of random characters), then it is nearly trivial for the attacker to crack p' with knowledge of p.

8. FUTURE WORK

While computing direct attack risk, we assumed that the attacker in question was unskilled and did not have expensive hardware. We believe that it would be useful to further extend the simulation to account for the greater computational power of organizations and governments. Furthermore, we did not take the cognitive burden of memorizing usernames into account; different services have varying username requirements, and often times a user's first choice has already been taken. Finally, we assumed that users would be willing to comply with these memorization techniques. It would be useful to account for the possibility that users may simply be unwilling to comply with either technique in a real-world scenario. (However, even if we can't achieve full compliance in practice, running simulations that assume full compliance can still be useful. If running a reliable simulation tells us that even if users fully comply with a policy, we'll achieve bad security, then we shouldn't adopt that policy. That is, we can weed out hopeless policies.) Modeling cognitive burden accurately remains a challenge, but it is essential to simulating how users interact with authentication systems.

9. CONCLUSIONS

Complying with password requirements is a burdensome task that leads users to write down passwords and reuse them across services. As a result, accurately modeling cognitive burden is critical to agent-based simulations of password use. Although strategies for creating memorable passwords have been proposed many times (e.g. [8, 12, 31, 18]), it is critical to evaluate their effectiveness on more than just risk of brute-force cracking.

In our agent-based model of aggregate security, we found that system-generated passphrases generally led to lower aggregate security than standard passwords when sites enforce even moderate password length requirements. However, passphrases performed relatively well when sites enforced lax password length requirements. As mentioned earlier, we believe that this result can be explained by the fact that as password length requirements increase, system-generated passphrases become exponentially easier to crack than standard passwords, outweighing the benefit to cognitive burden that results from using passphrases over standard passwords. Further, system-generated passphrases offer users less flexibility than user-defined passwords, and our model assumed that users cannot generate their own passphrases. When users are allowed to create their own passphrases, they often choose ones that are considerably easier to guess than system-generated ones. Our initial simulation has not shown clear benefits from system-generated passphrases. We suspect that new types of service-enforced constraints on passphrases might increase aggregate security (e.g. rejecting passphrases that parse as sentence fragments, since that greatly reduces strength). More modeling work is needed to understand the consequences of this approach.

Spaced repetition is a memorization technique that can be applied across nearly any domain of knowledge, and our simulation supports its effectiveness. Shorter passwords are generally less secure but far easier to memorize initially. By allowing users to memorize a short password initially and expand its length and complexity over time, the initial password memorization task becomes easier. In our simulation, spaced repetition of passwords increases aggregate security, even while taking into account the fact that an attacker can crack the initial password far more easily. However, it is important for the time interval between lengthenings to be unknown to the attacker. In theory, an authentication system could use a combination of entropy and the user's login success rate to ensure this assumption holds.

10. REFERENCES

[1] S. Arnell, A. Beautement, P. Inglesant, B. Monahan, D. Pym, and A. Sasse. Systematic decision making in security management modelling password usage and support. In *International Workshop on Quantitative Aspects in Security Assurance. Pisa, Italy*, 2012.

[2] A. Beautement, M. A. Sasse, and M. Wonham. The Compliance Budget: Managing Security Behaviour in Organisations. In *Proc. New Security Paradigms Workshop*, NSPW '08. ACM, 2008.

[3] S. Bird. NLTK: The Natural Language Toolkit. In *Proc. COLING/ACL*, COLING-ACL '06. Association for Computational Linguistics, 2006.

[4] J. Blythe. A dual-process cognitive model for testing resilient control systems. In *International Symposium on Resilient Control Systems (ISRCS)*. IEEE, 2012.

[5] J. Blythe, R. Koppel, and S. W. Smith. Circumvention of security: Good users do bad things. *IEEE Security & Privacy*, 11(5):80–83, 2013.

[6] J. Bonneau and S. Schechter. Towards reliable storage of 56-bit secrets in human memory. In *Proc. 23rd USENIX Security Symposium*, SEC'14, 2014.

[7] A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang. The tangled web of password reuse. In *NDSS*, volume 14, pages 23–26, 2014.

[8] V. Fiorentini, M. Shao, and J. Medero. Generating Memorable Mnemonic Encodings of Numbers. *arXiv:1705.02700 [cs]*, May 2017. arXiv: 1705.02700.

[9] S. T. Fiske and S. H. Kang. Spaced repetition promotes efficient and effective learning: Policy implications for instruction. *Policy Insights from the Behavioral and Brain Sciences*, 3(1):12–19, 2016.

[10] D. Florêncio and C. Herley. A large-scale study of web password habits. In *Proc. International Conference on World Wide Web*, WWW '07. ACM, 2007.

[11] D. Florêncio, C. Herley, and P. C. Van Oorschot. Password Portfolios and the Finite-Effort User: Sustainably Managing Large Numbers of Accounts. In *Usenix Security*, pages 575–590, 2014.

[12] A. Forget, S. Chiasson, P. C. van Oorschot, and R. Biddle. Improving Text Passwords Through Persuasion. In *Proc. Symposium on Usable Privacy and Security*, SOUPS '08. ACM, 2008.

[13] G. D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, Mar. 1973.

[14] S. Gaw and E. W. Felten. Password management strategies for online accounts. In *Proceedings of the second symposium on Usable privacy and security*, pages 44–55. ACM, 2006.

[15] D. Goodin. 25-GPU cluster cracks every standard Windows password in <6 hours, Dec. 2012.

[16] A.-M. Horcher and G. P. Tejay. Building a better password: The role of cognitive load in information security training. In *Proc. IEEE Int. Conference on Intelligence and Security Informatics*, ISI'09, 2009.

[17] B. Ives, K. R. Walsh, and H. Schneider. The domino effect of password reuse. *Communications of the ACM*, 47(4):75–78, 2004.

[18] K. A. Juang, S. Ranganayakulu, and J. S. Greenstein. Using System-Generated Mnemonics to Improve the Usability and Security of Password Authentication. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 56(1):506–510, Sept. 2012.

[19] B. Korbar, J. Blythe, R. Koppel, V. Kothari, and S. W. Smith. Validating an agent-based model of human password behavior. In *Workshops at the AAAI Conference on Artificial Intelligence*, 2016.

[20] V. Kothari, J. Blythe, R. Koppel, and S. Smith. Password logbooks and what their amazon reviews reveal about their users' motivations, beliefs, and behaviors. 2017.

[21] V. Kothari, J. Blythe, S. W. Smith, and R. Koppel. Measuring the security impacts of password policies using cognitive behavioral agent-based modeling. In *Proc. Symposium and Bootcamp on the Science of Security*, HotSoS '15. ACM, 2015.

[22] S. Mujeye and Y. Levy. Complex passwords: How far is too far? the role of cognitive load on employee productivity. *Online Journal of Applied Knowledge Management*, 1(1):122–132, 2013.

[23] R. Munroe. Password Strength, 2012.

[24] K. Renaud and L. Mackenzie. Simpass: Quantifying the impact of password behaviours and policy directives on an organisation's systems. *Journal of Artificial Societies and Social Simulation*, 16(3):3, 2013.

[25] S. Riley. Password security: What users know and what they actually do. *Usability News*, 8(1):2833–2836, 2006.

[26] B. A. Rodrigues, J. R. B. Paiva, V. M. Gomes, C. Morris, and W. P. Calixto. Passfault: an Open Source Tool for Measuring Password Complexity and Strength. Orlando, Florida, Mar. 2017.

[27] R. Shay, A. Bhargav-Spantzel, and E. Bertino. Password policy simulation and analysis. In *Proceedings of the 2007 ACM workshop on Digital identity management*, pages 1–10. ACM, 2007.

[28] R. Shay, P. G. Kelley, S. Komanduri, M. L. Mazurek, B. Ur, T. Vidas, L. Bauer, N. Christin, and L. F. Cranor. Correct horse battery staple: Exploring the usability of system-assigned passphrases. In *Proc. Eighth Symposium on Usable Privacy and Security*, SOUPS '12. ACM, 2012.

[29] R. Shay, S. Komanduri, P. G. Kelley, P. G. Leon, M. L. Mazurek, L. Bauer, N. Christin, and L. F. Cranor. Encountering Stronger Password Requirements: User Attitudes and Behaviors. In *Proc. Symposium on Usable Privacy and Security*, SOUPS '10. ACM, 2010.

[30] E. Stobert and R. Biddle. The password life cycle: user behaviour in managing passwords. In *Proc. SOUPS*, 2014.

[31] U. Topkara, M. J. Atallah, and M. Topkara. Passwords Decay, Words Endure: Secure and Re-usable Multiple Password Mnemonics. In *Proc. ACM Symposium on Applied Computing*, SAC '07. ACM, 2007.