

Keyboard Based Password Generation Strategies

Ben Harsha
Purdue University

Jeremiah Blocki
Purdue University

Abstract

A Human Computable Password Generation Scheme is a strategy which allows a user to quickly (re)generate multiple distinct passwords for different web sites by transforming a challenge (e.g., the name of a website like Google) into a password using a small set of secrets (e.g., words, person-action-object stories) that the user has memorized. The goal of these schemes is to help users develop increasingly secure and memorable passwords. The transformation should be simple enough that the user can execute it quickly in their head without assistance, and should produce distinct password for different web sites. In a Keyboard Based Password Generation Scheme the transformation is based on the location(s) of the letters in the challenge on the user's keyboard. This potentially makes the transformation rules easier to learn and apply since we can safely assume that the user will always have a keyboard when (re)generating a password for authentication. We propose several new Keyboard Based Password Generation Schemes and conduct a longitudinal user study (400+ users over 50+ days) to evaluate the usability of each scheme.

1 Introduction

A Human Computable Password is a password that can be generated on the spot by a user for some particular purpose. More importantly, the must be generated in a manner that is not too complex i.e. they should be easy enough to be computed mentally. Such passwords can be based on a smaller number of short secrets (say, 5-20) and may be combined to form a larger number of long chunk-based passwords. This carries several advantages. First, there is less information for

a human to memorize, meaning that larger numbers of unique passwords can be generated without relying on some assistive tool like a password manager. Second, if (or perhaps when) a password breach occurs only a few chunks of the password are leaked, meaning it is more difficult to run an online attack on other accounts using information gained from an offline attack. This helps avoid some of the issues that arise when passwords are reused [8]. Finally, the scheme is naturally rehearsing i.e. users are constantly rehearsing the secrets they memorized as they log in to various sites [1].

We describe a human computable password scheme that is based on the location of letters on a keyboard, the idea being that when a user is logging in they have a device for helping them construct the password literally at their fingertips. To begin, we have a user memorize nine secrets (described in more detail in Section 3). The user is then trained to transform challenge strings (e.g., web site names) into password based using these secrets.

We analyze the security and usability of each of our schemes. To analyze the usability of our schemes we conducted a two-stage longitudinal user study on MTurk (400+ users over 50+ days). Users were first asked to memorize their secrets and trained to use a Keyboard Based Password Generation Scheme and were then periodically asked to return over 50+ day period. Each time a user returned they were asked to (re)generate the passwords for five randomly selected challenge sites (Alexa Top 100) using the scheme. In the shorter (pilot) stage, we tested a few variants of the keyboard scheme along with one additional Human Computable Password strategy and a control group. From this first stage, we selected the best-performing groups and ran a 2nd larger stage to determine which of the groups performed best.

The results of the user study are largely encouraging. Each time users were asked to return over 50+ days most of the users were able to successfully compute all five challenge passwords. Some of the results were a bit surprising e.g., users seemed to perform just as well when asked to memorize 9 random words as opposed to three person-action object stories. The results we obtained for one of our control groups were seemingly contradictory. We discuss the results of the user study in more detail in the Section 6.

2 Previous Constructions

Several previous constructions for human computable passwords have been investigated in the past [1, 2, 9]. Blum and Vempala introduced a work describing several proposed HCP strategies in 2015 [5]. For each of these strategies, they provide estimates for the amount of human work it takes to compute each one. They tested authentication times on a group of 8 friends and family and found that it can take anywhere from 5 to 20 seconds to authenticate using some of these strategies. BBDV17 [2] presented a high effort/high security scheme where users authenticate by memorizing a random image-to-digit mapping that allows them to correctly respond to visual challenges. This method requires a high amount of effort from a user to memorize the mapping (e.g. hours of memorization and 75+ seconds to respond to each challenge), but pays off by giving a high level of security. Specifically, it takes a large number of leaks (e.g. 100+) before an adversary would be able to correctly predict their passwords.

3 Keyboard Generation Schemes

In this section, we describe the various keyboard based password generation strategies we have investigated. A total of 5 variants were tested, with two being selected in the final study. However, all of these variants follow the same core strategy. First, a user does a memorization step to memorize nine chunks divided into three groups, as shown in Table 1.

Table 1: Sample words + chunking structure

group/word	W1	W2	W3
1	apple	vacuum	water
2	laptop	glass	alarm
3	thumb	umbrella	helium

When a user wants to authenticate, the following (general) strategy is used:

1. Take some input string s and select the first three characters. If there are fewer than three characters then wrap back around to the beginning [e.g. "t" becomes "ttt" and "ab" becomes "aba"].
2. Take the first letter of the challenge and look where it falls on a keyboard. Where it falls determines which group/word is selected. Say group/word i is selected.
3. The first word is selected based on i
4. Repeat, selecting the second and third words again based on some specific i .

As an example, say we had the challenge "sample". Our challenge would be "sam". "s" might fall in group 2, "a" in group 2 as well, and "m" in group 3. Thus we pick group 2 word 1, group 2 word 2, and group 3 word 3 to give the password "laptopglasshelium" if using the words and groups from Table 1. Table 2 shows this process visually.

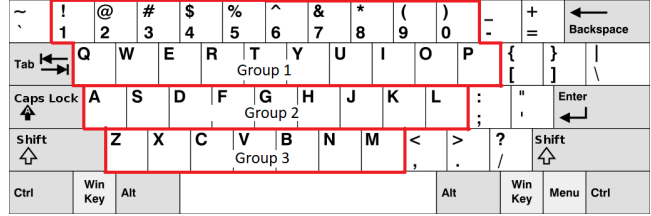


Figure 1: Groupings based on rows

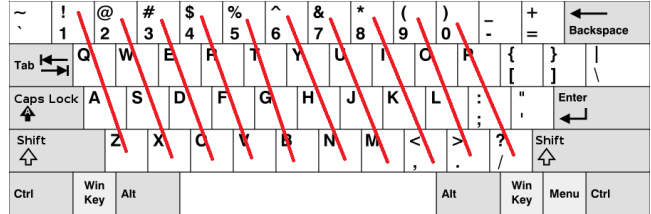


Figure 2: Groupings based on columns

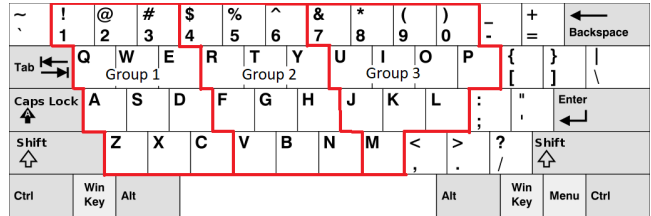


Figure 3: Groupings based on sections

Table 2: Sample words + chunking structure with solution

group/word	W1	W2	W3
1	apple	vacuum	water
2	laptop	glass	alarm
3	thumb	umbrella	helium

The primary difference between our strategies is how the groups and words are selected. We considered three grouping methods and two word generation methods. The grouping methods were the following

- **Row-based groups:** Groupings were based on rows e.g. $G_1 = \{QWERTYUIOP\}$, $G_2 = \{ASDFGHJKL\}$, etc (for a standard US keyboard). Groupings are shown in Figure 1.
- **Column-based groups:** Groupings were based on columns e.g. $G_1 = \{1QAZ\}$. Diagram shown in Figure 2. (Note: This strategy uses a unique indexing strategy where each column corresponds to a single word)
- **Section-based groups:** Groupings were based on sections of a keyboard. Diagram shown in Figure 3.

These were combined with one of two strategies for word generation. In the first strategy, words were selected at random from a list of the 10,000 most common English words. In the second words were generated as a series of three Person-Action-Object stories (e.g. EinsteinKissingPirannah), which have previously been shown to aid users in memorization [1, 4].

The product of these two sets gave us six keyboard based password generation strategies that were run in the first round of the study. We name these based on the keyboard grouping

strategy (rows, columns, or sections) and whether or not they used PAO stories (PAO vs words). For the columns strategy only the random words method was used (as the PAO structure would be destroyed with this method). We refer to the groups using these labels e.g. rows-PAO, sections-words, etc.

In addition to these five groups, two additional groups were run as a comparison. The first group was a control, where users were asked to remember a single randomly generated string that was the correct response to all challenges. The second was a human-computable strategy based on keeping a running sum held mentally that we refer to as the running-sum method.

In the running sum method, the users memorize a single six-digit number e.g. 159643. The number is split into three groups e.g., 15, 36, and 43. Second, each user memorizes a single word e.g. "textbook". When a challenge was given such as "facebook" they repeat the word until it is at least 10 characters long e.g. "facebookfa". To generate the response they begin with their set word "textbook". Next, they look at each group of numbers in turn. They take the first number and use it to index into the 10 character challenge string e.g., $1 \rightarrow \text{"f"}$. The user then locates this character (e.g., "f") on the keyboard and traces up the keyboard (as in Fig 2) to find the number above e.g. "f" $\rightarrow 4$. This number is added to the 2nd number (e.g., 5) in the group and appended to the base word e.g., $5 + 4 = 9$. This is appended to the base word. The process is repeated for the next two groups of digits e.g., 36 and 43. In our example, the correct response to "facebook" would be "textbook996".

4 Security

In this section we introduce a new security definition for human computable passwords schemes and compare it to the definition of Blocki et al. [1] and the security definition of Blum and Vempala [5].

4.1 Security Model

The goal of our security model is to estimate the total number of *additional* user passwords that an attacker might learn after a few breaches¹. If a user's strategy is to pick one (possibly) strong password then the attacker will learn all of the user's passwords after just one breach. We consider the following experiment $\mathcal{PWDS\mathcal{E}C}_k$: (1) The challenger (simulating a human user) selects the secrets for a human computable password scheme and generates passwords for m accounts. (2) The challenger picks a random subset of k accounts (out of m). For each of these accounts the challenger sends the challenge (account name) and the corresponding response (password) to the adversary. (3) Let $0 \leq X \leq m - k$ be the number of additional accounts that the attacker can infer with certainty i.e., accounts for which the attacker can infer all of the necessary secrets to compute the corresponding password. The output of the experiment

¹We assume that if a breach occurs then the attacker automatically learns the corresponding password. This is certainly true if passwords are stored in plaintext. Even if passwords are hashed Blocki et al. [3] found that most breached sites performed insufficient key-stretching and that an attacker would crack *almost all* user passwords.

is $p_{extra} = X / (m - k)$ the fraction of remaining accounts that are directly exposed. We let $\mathbf{E}[\mathcal{PWDS\mathcal{E}C}_k] = \mathbf{E}[X] / (m - k)$ denote the expected value of p_{extra} .

Definition 1 We say that a human computable password scheme is (k, γ) -secure if $\mathbf{E}[\mathcal{PWDS\mathcal{E}C}_k] \leq \gamma$.

Compared to the security model of Blocki et al. [1] definition 1 is slightly weaker. Briefly, in the security model of [1] the attacker is allowed to adaptively select k accounts to breach and then the attacker wins if s/he can breach any of the remaining accounts (say given 3 guesses per account). While this is a strong security model the definition is sometimes too strong that it does not distinguish between less/more secure password strategies. For example, the model of Blocki et al. [1] would not distinguish between the strategies REUSE where the user picks one single password for all accounts and TIERED where the user picks three passwords (weak/medium/strong) which are assigned based on the importance of the website. The TIERED (resp. REUSE) scheme offers stronger (resp. weaker) protection e.g., after a breach the attacker only learns *some* (vs. *all*) of the user's passwords.

Blum and Vempala [5] defined a similar security game which was played in rounds. In each round the challenger selects a random web site (challenge) and the attacker is given several (e.g., 10) guesses to predict the password. If any of these guesses are correct then the experiment halts. Otherwise, we move on to the next round where the challenger shows the attacker the correct password (response) in the last round before selecting a new challenge for the current round. The output of the experiment is the total number of rounds Q and Blum and Vempala use $\mathbf{E}[Q]$ to quantify the security of the password scheme — schemes with a higher $\mathbf{E}[Q]$ are viewed as more secure. One limitation of this model is that the attacker cannot choose which accounts to attack e.g., if attacker infers the user's Amazon password in an early round the game may still continue if the attacker is never challenged to produce this particular password. When challenges are drawn from the Alexa Top 100 web sites we found that $4 \leq \mathbf{E}[Q] \leq 5$ for each of our human computable password schemes. We remark that one should not expect cryptographic level security for a usable human computable password scheme, and note that for commonly used password schemes (e.g., REUSE) we have a much lower value of $\mathbf{E}[Q]$ (e.g., $Q = 1$).

One might notice that in our game (and in [5]) the attacker does not get to adaptively get to select which accounts are breached. Arguably this is a reasonable assumption for our context e.g., the breach at RockYou was due to a failure to protect against SQL injection attacks and was uncorrelated with the location of the letters in the string "RockYou" on the keyboard. A crucial difference between our model and [5] is that we allow the adversary to attack *any* of the remaining user accounts after a breach. Contrast this with the security game of [5]. If an attacker infers the user's Amazon password during an early round the game may still continue as long as the attacker is never challenged to produce this particular password.

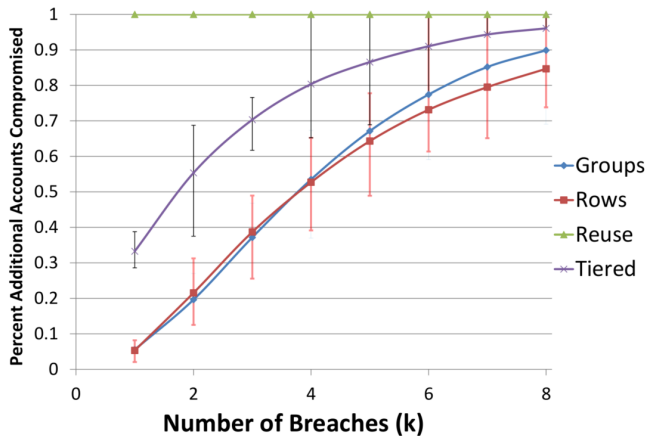


Figure 4: Empirical Security Analysis

4.2 Security Analysis

We analyzed the security of our human computable password schemes using the Alexa top 50 US web sites for our experiments. In particular, for each value of k (number of breaches) we repeated the $\mathcal{PWDS}E C_k$ experiment $n = 10^5$ times to estimate $\mathbf{E}[\mathcal{PWDS}E C_k]$ — the fraction of remaining accounts that are directly exposed after k breaches. Our results are shown in Figure 4. The error bars show the 25% percentile (resp. 75% percentile) over all $n = 10^5$ experiments.

Discussion From a security standpoint it doesn’t make much difference how we partition the letters on the keyboard (e.g., rows vs groups). In both cases we find that after 2 breaches we expect that $\approx 80\%$ of the user’s remaining accounts will be safe. This compares favorably with the popular tiered password strategy (in our empirical experiments each Alexa Top 50 web page is assigned to either the weak/medium/strong tier uniformly at random) where an attacker can predict $> 50\%$ of the user’s remaining passwords in expectation after just 2 breaches. Similarly, as shown in Figure 4 if the user reuses the same password for every account then the attacker will crack 100% of the remaining passwords after just one breach.

5 Study Design

The following two-phase study was conducted to determine the effectiveness of these methods. All portions of the study were reviewed and approved by the Purdue University Institutional Review Board before the study was conducted. Both phases of the study were identical in structure and differed only in the number of return visits and which groups study participants were assigned to.

The first phase of the study was designed as a pilot to limit the total number of groups being studied. We used three keyboard grouping strategies, rows (Figure 1), columns (Figure 2), and sections (Figure 3). We also split based on assigning words randomly (which we call “words”) and based on Person-Action-Object stories (which we label as “PAO”). We refer to the groups based on the keyboard groupings (row,

col, sec) and word selection method (PAO, words) with the exceptions of running sum and control. Upon the conclusion of the first phase we selected the strategies that performed best (based on successful authentication rates) for inclusion in the second, larger phase. Participants were recruited via mechanical Turk in both phases and were paid USD 0.75 for all visits including the introduction and all follow-up visits.

Initial Visit During the first visit the subjects were guided through five steps - informed consent, directions, training/practice, testing, and a debriefing.

1. In the first step the subject’s informed consent was recorded.
2. In the directions stage the user was introduced to the password creation scheme they were assigned and shown their secrets. These instructions included several worked examples using their secrets for them to read.
3. The users were given several practice challenges to complete. During the practice challenges the users were able to see their secrets on screen as an aid. When a challenge was answered correctly it flashed green, and when it was answered incorrectly it flashed red and showed the correct answer.
4. The users were given 5 more challenges, but without their secrets being visible. The users could not advance past this page until a) they answered all 5 challenges correctly or b) they made a total of 10 mistakes over all challenges. We allow the user to move on after 10 misses to prevent them from being unable to complete the study. After a mistake is made they are given a link to view their secrets in case they have forgotten them.
5. A debriefing page with some closing information was shown. This included their followup schedule (with follow up visits x days after the initial visit for $x \in \{0.5, 1.25, 2.4, 4.1, 6.5, 10.4, 16.1, 24.6, 37.4, 56.7\}$) and a request to not write their secrets down.

Follow Up Visits From here we began a series of 6 (resp. 10) follow-ups in the pilot (main) study over a time period 10 (resp. 55) days. Each follow up was identical to stage 4 from the initial visit. The gap between the initial visit and the first follow up was set to 12 hours, with subsequent gaps increasing by a factor of 1.5 i.e., the gap between follow up visits i and $i + 1$ is $0.5 \times 1.5^{i+1}$ corresponding to a ≈ 19 day gap between visits 9 and 10. While we anticipate that an average internet user would rarely go nearly 3 weeks without logging in, we intentionally picked this aggressive schedule to stress test the keyboard based password generation schemes. Prior results of Blocki et al. [4] indicate that users are able to memorize/maintain four person-action-object stories following this same spaced repetition schedule.

NASA TLX Survey Once a user completed all 10 follow-up visits challenges were finished we collected some optional basic demographic data from the users, a question asking if they wrote their secrets down during the study, and a NASA TLX [7] survey.

Study Conditions We ran the pilot study with seven groups - rows-words, rows-PAO, cols-words, sec-words, sec-PAO, running sum, and control. During the pilot study we found that users performed better with the “section” groupings than with row/column groupings. Based on these results we eliminated all conditions based on rows and columns. We also added a new group called sec-PAO-new with modified instructions for PAO memorization², leaving five groups in the main study. These groups were the control, the running sum, sec-words, sec-PAO-old, and sec-PAO-new. A total of $N = 409$ participants reached the end of the first stage (i.e. introduction to their assigned strategy) of the study. We saw 61, 57, 101, 80, and 110 users in the keyboard sections (KS) w/o PAO, KS with PAO (using old instructions), KS with PAO (using a revised set of instructions), the running sum, and control groups respectively.³

6 Preliminary Results

The following represents ongoing analysis of the main phase of the study. As this is still a work in progress not all data has been analyzed completely. In the main phase of the study, we had a total of 409 participants who completed the initial visit. As the study progressed we saw a fairly steady retention rate of between 75 and 90% per visit. The retention rate was relatively stable across different conditions. There was a bug in our data collection script which affected the running sum group. Since we have incomplete data for this group we omit it from Figures 5 and 6 below. However, we remark that the incomplete data we do have shows that users in the running sum condition struggled to respond to challenges and the successful authentication rate was *much* higher in other conditions like sec-PAO and sec-words.

6.1 Numbers of misses

For each visit, we recorded the total number of misses each user made during each follow up visit. Figure 5 shows the median number of misses over time for each group. We remark that a median value below 10 means indicates that most users were eventually able to answer all five challenges correctly without any hints during that particular follow up visit. Observe that for sec-words, sec-PAO-old and sec-PAO-new that the median miss rate is *always* below 10 in *every* follow up visit over 50+ days. This suggests that users are able to continue using our keyboard based human computable strategies over time. We remark that the median miss rates are particularly high for the control group indicating that most users were not able to remember their random system assigned

²The new instruction set was developed after the pilot phase after feedback from the local psychology group who specialize in human memory and password studies.

³Participants were assigned to their groups round-robin style. As many users were doing their initial visit in parallel we assigned groups upon arrival to spread the users out evenly. Some users quit before finishing their initial visit which explains the slightly uneven number for each group. Thus, the fact that some groups had lower participation rates may indicate that user’s found this condition to be more challenging or time consuming.

password until the final round. One potentially surprising observation is that miss rates generally held steady over time (and slightly increased for sec-PAO-old). Based on the prior results of Blocki et al. [4] one might have predicted that the miss rates would decrease over time. On the one hand user’s get more practice as time increases, but on the one hand the gap between follow up rehearsals also increases over time.

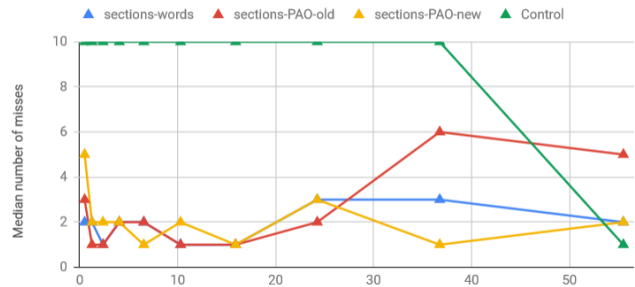


Figure 5: Median miss rates per group

6.2 Timing results

During each follow up visit how long it took each user to successfully respond to each challenge i.e., the time between focusing on the password window and successfully authenticating. Our measure includes wasted time due to mistakes. Figure 6 shows the median time to respond to each individual challenge for each study condition excluding data from user’s who were unsuccessful (≥ 10 mistakes) in that round. Users in the sec-words condition were impressively fast ≈ 5 seconds per challenge. We had anticipated that PAO strategies would be easier for users to navigate in comparison with the random words strategy. Surprisingly, users in the sec-PAO (old/new) conditions were much slower e.g., ≈ 20 seconds per challenge during the last follow up visit.

We also noted that median authentication times tend to dip during the first few follow up visits in all conditions. In the sec-words and control conditions the authentication times hold constant after the initial dip. By contrast, authentication times for sec-PAO (old/new) seem to return back to their starting values at the end of the study. This may be caused by the lengthy gaps between visits at the later stages. We remark that in all conditions authentication times would likely be improved by daily rehearsal, which may also more accurately represent a typical user’s login habits.

6.3 NASA-TLX results

A standard NASA TLX [7] survey was given to subjects at the end of the study. Subjects were asked to answer a question on a scale from 1 to 7, with the results shown in Figure 7. We can see that many of the strategies perform fairly similarly in many categories. One of the main exceptions is with the running sum group, where users clearly rated it as requiring the highest amount of physical work and effort. Interestingly, we also note that users rated the keyboard PAO based strategies as similar or slightly lower temporal effort in comparison to the random

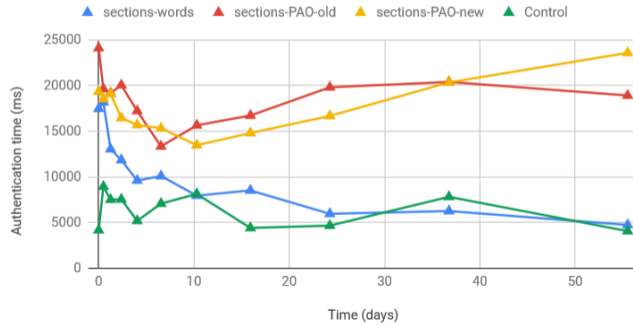


Figure 6: Median time to authenticate per group

words (non-PAO) based strategy. This was surprising given that Figure 6 shows that the authentication times were much higher for the PAO based strategies!

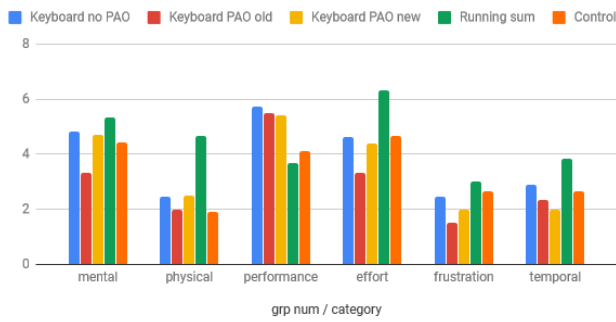


Figure 7: NASA TLX results

7 Future work

The grand challenge when designing HCP strategies is to develop simple schemes that are increasingly usable and/or secure. We make progress towards that goal though we are still in the process of analyzing our data and drawing conclusions. One direction for future work is to draw concrete comparisons with other human computable password schemes e.g., do keyboard based methods offer any statistically significant advantages? Another challenge for the field is to develop strategies to update passwords e.g., in response to expiration policies. We remark that requiring password updates has not been found to be an effective strategy to improve security [6] (except, of course, in the case of a breach), and that some organizations (e.g., Microsoft) have moved away from the practice. Nevertheless, many organizations do still have password expiration policies so it is important to develop usable/secure coping strategies.

Acknowledgments

Our research was supported by the National Science Foundation under award #1704587. Ben Harsha was also supported

References

- [1] Jeremiah Blocki, Manuel Blum, and Anupam Datta. Naturally rehearsing passwords. In Kazuo Sako and by a Rolls-Royce Doctoral Fellowship. The views expressed in this paper are those of the authors and do not necessarily reflect the views of National Science Foundation or Rolls-Royce. Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 361–380, Bangalore, India, December 1–5, 2013. Springer, Heidelberg, Germany.
- [2] Jeremiah Blocki, Manuel Blum, Anupam Datta, and Santosh Vempala. Towards human computable passwords. In Christos H. Papadimitriou, editor, *ITCS 2017: 8th Innovations in Theoretical Computer Science Conference*, volume 4266, pages 10:1–10:47, Berkeley, CA, USA, January 9–11, 2017. LIPIcs.
- [3] Jeremiah Blocki, Benjamin Harsha, and Samson Zhou. On the economics of offline password cracking. In *2018 IEEE Symposium on Security and Privacy*, pages 853–871, San Francisco, CA, USA, May 21–23, 2018. IEEE Computer Society Press.
- [4] Jeremiah Blocki, Saranga Komanduri, Lorrie Faith Cranor, and Anupam Datta. Spaced repetition and mnemonics enable recall of multiple strong passwords. In *ISOC Network and Distributed System Security Symposium – NDSS 2015*, San Diego, CA, USA, February 8–11, 2015. The Internet Society.
- [5] Manuel Blum and Santosh Srinivas Vempala. Publishable humanly usable secure password creation schemas. In *Third AAAI Conference on Human Computation and Crowdsourcing*, 2015.
- [6] Sonia Chiasson and Paul C Van Oorschot. Quantifying the security advantage of password expiration policies. *Designs, Codes and Cryptography*, 77(2-3):401–408, 2015.
- [7] Sandra G Hart and Lowell E Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, volume 52, pages 139–183. Elsevier, 1988.
- [8] Blake Ives, Kenneth R Walsh, and Helmut Schneider. The domino effect of password reuse. *Communications of the ACM*, 47(4):75–78, 2004.
- [9] Samira Samadi, Santosh Vempala, and Adam Tauman Kalai. Usability of humanly computable passwords. In *Sixth AAAI Conference on Human Computation and Crowdsourcing*, 2018.