FastPass: Helping Users Get Better at Password Creation
by

Othman Bichouna Jay Yalamanchili

ABSTRACT
Much research has been done on how to give users visual indicators on password strength. However, these strength indicators don't tell users how to create better passwords. In our study, we sought to use password strength indicators to improve how users create passwords. We created a website that allowed users to enter passwords and gave them a score and suggestion on how to improve their password. At the end of the 5 days of entering passwords, there was a statistically significant difference between the users' score at the beginning and at the end of the study--indicating users did improve their password creating skills over time.

1. INTRODUCTION
Password strength meters have recently become popular on many websites to provide users with a sense on how secure their passwords are. Much research has been done on creating the best ways to indicate password strength. However, these meters do not help users improve their own passwords. Our research extends the idea of password strength meters to create a web application that allows users to enter passwords and receive a password score and suggestions on how to improve their passwords. Our password score was based on features discussed in other papers which were lightweight and easy to calculate client side unlike other password scoring methods that are more robust and done server side [5]. Our general process will be to create a website that participants will go to once a day for the study duration of 5 days, and write 10 unique passwords. For the three middle days, these passwords will be based around a "seed" phrase so that data between subjects is comparable between each other. On the first and last day, there will be no such phrases. Participants will be given password-security increasing suggestions on every password they create. We will test to see on the last day, without suggestions or any seed phrases, if their password score, a measurement of their password strength and security.

2. RELATED WORK
The nature of the password policy itself, past the measurement of complexity, but determining usability and scalability in terms of simplicity for the user in mind, but also achieving rates of security high enough to consider be both usable and secure, is of the utmost priority in our study. Seeing as we will be creating a "one-size-fits-all" policy for our program, as it will try to make a complex password and help guide the user towards a complex password beyond the actual service's own policy—this is a priority. For example, say a user is making an account for a new website, and the password policy is simplistic—there is no policy, just write whatever you desire—this is most clearly a very usable policy for the user, but is also just as insecure. What is particularly interesting about users is that they have notable knowledge of certain steps into making their passwords more secure, but despite this known knowledge, users will still decide to not create secure passwords [3]. Instead users would still partake in bad and insecure password practices, including using all lowercase letters, few numbers, relevant information to themselves (such as birthdays or anniversaries) and would use relevant dictionary terms [3]. What this means for this study is that this sort of hypothesis should be tested again, and then to study the nature of our program's ability to not only give relevant password complexity information, but to also act as a *reminder* for the user to follow secure password practices, and how this affects later user behavior.

What might come as a surprise about user behavior, however, is that despite having some knowledge on secure password policy, it seems like users are still unaware of the insecurity of their own passwords [1]. This seems extremely at odds, but the work of Bees et. Al [1] shows that users will place more priority on certain password practices, in particular, overestimating the strength of adding digits to the password, and underestimating the predictability of certain phrases and substrings [1]. This acts as a clear misunderstanding of password practice, despite the password practice itself being known. Our program should then try to solve this issue by giving not only relevant and secure password practice, but to also rank these practices by studying importance in security, in contrast with the direct misunderstandings of users. For example, the first suggestion given by our program could be “avoid these common phrases and relevant substrings: *example.*”

The actual practices that should be implemented into our program need to try to follow the most relevant and secure password policy. The important and necessary policies that are relevant to the program will be to “avoid using length-only requirements,” “implementing a substring blacklist,” and “avoid using Comp8” as the password composition strength tester as explained by Shay et al[2]. The important elements here are the avoidance of length-only requirements and implementing a substring blacklist, like a key-word dictionary blacklist. An issue we can see arising from this is that these requirements can frustrate users who do not want to spend time working on a complex password (hence why many password managers have the ability to create a generated password). Length-only requirements act as the most usable requirement for users, but in this way also acts as the least secure. Users who had to make their passwords longer, did so in predictable and insecure ways—which was solved by using the substring blacklist [2]. This information seems to agree with the study conducted by Riley [3] that users will take the path of least-resistance in creation of their password, despite knowing notions of secure password practice. Our program will make sure to give the user a generated substring blacklist based on the site they are on, for example, if on the site FaceBook, to avoid substrings in the password like “FaceBook, FB, FBook,” etc.

Another related paper that we took into consideration when building our study was How Does Your Password Measure Up? The Effect of Strength Meters on Password Creation [4]. This paper conducts a user study with different password strength meters that assess the strength of a user’s password to see if different strength formulas or visual indicators would help users create better passwords.The authors tested many different combinations of visual design and password strength assessment formulas on the users and saw how easy their passwords were to crack with existing tools. The authors found that visual and easy to understand meters led users to changing their passwords more frequently and more stringent password assessment meters led to harder to crack passwords. However, if the strength meter was too stringent the users got frustrated and didn't listen to it after a while. This work is similar to our own work in that it gave users a password strength meter which we will also do, and it gave good insight on how to do that effectively. Our work builds on this work because our strength indicator gave actual suggestions to users and also the goal of our research was not just to have users create one better password as it was in this paper, but to teach them the skills necessary to create better passwords wherever they are through a similar password meter tool.

We also took ideas from the paper LPSE: Lightweight password-strength estimation for password meters which gave a method to evaluate passwords client side, which we modified slightly in our own evaluation [5]. This paper presents a lightweight password evaluation algorithm that can be run client-side that is better than the existing state of the art methods. This was tested through a password guessing method which evaluated both strength scores by trying to crack the strongest passwords of each strength algorithm and reporting the compute time necessary. This work is relevant to our own research as we will likely use a modified version of LSPE since we want to score passwords on the client side effectively. Our differentiation with this work is that our password strength algorithm also provided suggestions to the user on how to improve their password. So we used the same scoring system and gave suggestions to the user. This score will form the backbone of our evaluation of user progress over time as well.

In designing our password strength score, we also followed the results laid out in the paper: On the Accuracy of Password Strength Meters [6]. This paper gives a set of metrics on how to assess the quality of various password strength meters. It breaks down the critical design choices and also the algorithmic choices that are important in creating a good measure of password strength. Our work builds on this as we use this paper as a foundation to assess our own metric of password strength as well as its design and how users will perceive it. We used the authors’ work as a foundation for a key component of ours: the strength score. Our end goal was to create an interactive strength score that will teach users better habits and using the methods outlined in this paper to create a good strength score is a good foundation.

3. DESIGN PRINCIPLES

The experiment was done through a website accessible on the internet at any point in time during the study duration. This website was programmed specifically for the study, and was written in CSS, HTML, and Javascript.

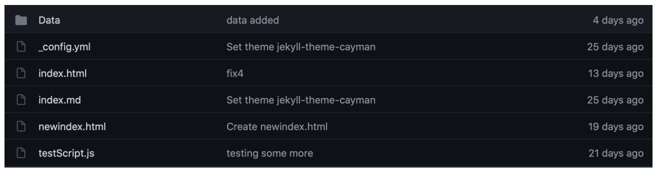
Parts of the website code in HTML and CSS was taken from a free open source template found online This was used to make the website more visually appealing for the participants. The rest of the programming in Javascript--the functionality of the website itself was done by us.

3.1 Github Pages and Privacy Concerns

The website was hosted on Github Pages, a static-webpage implementer designed around Markdown and Jekyll. The choice in using Github Pages was mostly two-fold--ease of use and privacy concerns.

Github Pages is a static-webpage, meaning that it is entirely client-side in design. This means that there would not be any difficult databases to be used in storing user information and data. This is of particular importance in the realm of security and privacy. When working with a study centered around participants' passwords, it can be difficult to ensure that a participant feels secure in writing down passwords, and this is one of our most tangible concerns with the study. There exists the underlying danger for a participant to be giving their possible password information for it to be possibly stored and used. By using a client-side only service like Github Pages, there was no way for us to actually keep any of the passwords that were written down, and Github themselves would not be able to store the password data either.

The second value of Github Pages was that it was a free and simple way to host a website for the entirety of the study. It also did something extremely remarkable--in that it was linked to the Github repository in which our code was held. This meant that at any time, our study was also "open-source" and the inner workings of the actual code was available at any time to a participant and to the world. This provides the benefit that if the participant had doubts about the storage of their passwords and information, they could check the Github repository to see that none of their data was stored--except for their password scores, which will be discussed later. As the study had limited time to be completed, and a prototype of the program was necessary, Github Pages also worked well with its ease of use.

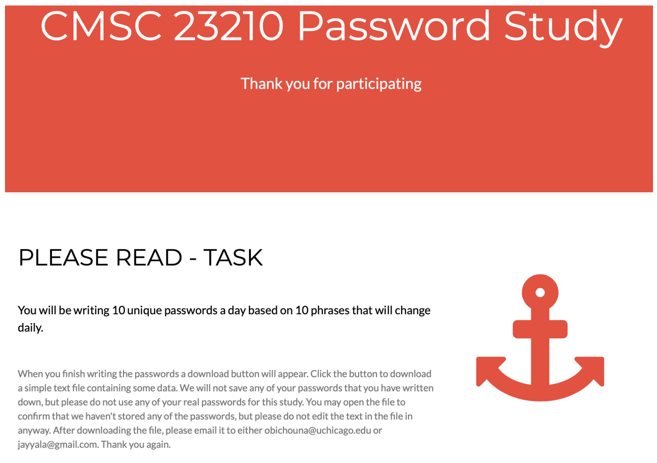


Data	data added	4 days ago
_config.yml	Set theme jekyll-theme-cayman	25 days ago
index.html	fix4	13 days ago
index.md	Set theme jekyll-theme-cayman	25 days ago
newindex.html	Create newindex.html	19 days ago
testScript.js	testing some more	21 days ago

Figure 1. The Github repository containing the open source code for the website.

3.2 The Interface

When a participant connected to the website they were first greeted by the description of their task. The task was given in the most simple terms possible--the website always having the description of the task acting as a necessary reminder in the event that a participant forgets or is unsure of what they are meant to do.



CMSC 23210 Password Study

Thank you for participating

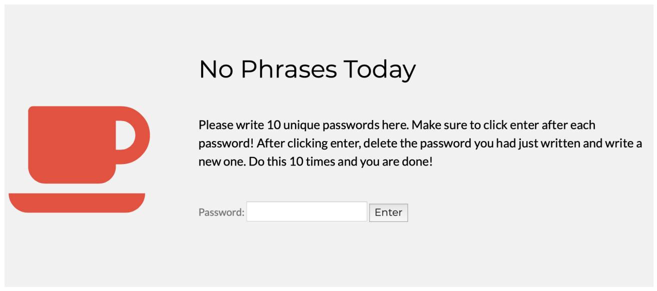
PLEASE READ - TASK

You will be writing 10 unique passwords a day based on 10 phrases that will change daily.

When you finish writing the passwords a download button will appear. Click the button to download a simple text file containing some data. We will not save any of your passwords that you have written down, but please do not use any of your real passwords for this study. You may open the file to confirm that we haven't stored any of the passwords, but please do not edit the text in the file in anyway. After downloading the file, please email it to either obichouna@uchicago.edu or jayyala@gmail.com. Thank you again.

Figure 2. The study website and the task description.

To access the actual interactive part of the website, a participant just had to scroll down to find the next part of the study, which featured the password box that participants would be using. Everyday the title of this section would change, featuring 10 new phrases randomly generated using an online word generator to use as “seeds” for the passwords for that day.


No Phrases Today

Please write 10 unique passwords here. Make sure to click enter after each password! After clicking enter, delete the password you had just written and write a new one. Do this 10 times and you are done!

Password: Enter

Figure 3. The second part of the website featuring the password box and phrases. Note that this figure does not include any “seed” phrases.

It is important to note that as we do not store the data, we cannot accurately determine if participants performed the task correctly or not--instead we just have to assume that they performed the task to the utmost of their ability and did not actively incorrectly follow the instructions.

3.3 Task

The task for the participants was to write 10 unique passwords in this password box. They would do this task everyday for the study duration of 5 days. Everyday, we would add 10 randomly generated “seed” phrases that would be necessary to be used in the 10 passwords for that day. At the end of the day, we would replace the 10 words with a new batch of 10 randomly generated phrases. For example:

- If the first seed phrase was the word “orange,” a participant could write the first password of the day to be “orange123.” Continue for all other 9 phrases.
- The next day, the first seed phrase is the word “ravine,” a participant could write the password “R@vine212.”

After a password is written into the password box, our password scoring algorithm will generate a “password-score” value that will be saved momentarily on the website infrastructure. The task is completed when 10 passwords are written and scored. Once this occurs, the saved password scores are then placed into a text file and downloadable by the subject.

The seed phrases are important for creating a consistent factor in which we could compare subjects’ data against each other. In the example above, with the phrase “orange,” it is possible to compare the first password score of one subject against another because we have some baseline assessment that they at least shared some similarity.

The first day does not have any seed phrases. The three next day contain 10 different seed phrases (for a total of 30 seed phrases over the study duration) and the last day does not have any seed phrases.

3.4 Password Scoring

We wanted to create a password scoring function that was lightweight and could be run on the client side so that user privacy could be preserved. In this process, we drew on ideas from the paper LPSE: Lightweight password-strength estimation for password meters as discussed in section 2. This paper described a method of scoring passwords by pulling features from the passwords and weighing them according to how strong each feature was. In our implementation, the features we chose to weight the passwords on were the number of uppercase letters, the number of lowercase letters, the length, the number of consecutive characters of the same character, the number of special characters and the number of unique characters. Our weighing formula was:

$$score = (3*upper + lower + 2*numbers + length + 4*special - consecutive)*(unique/length)$$

We weighed special characters, then upper, then numbers, then lower case letters in order as that is considered the best ordering of the security of each type of character in the work of Guo and Zhang [5]. We then adjusted for the consecutive characters of the same kind to adjust for the length of the password, and then scaled the entire password by the fraction of unique characters to have the same effect.

3.5 Password Suggestions

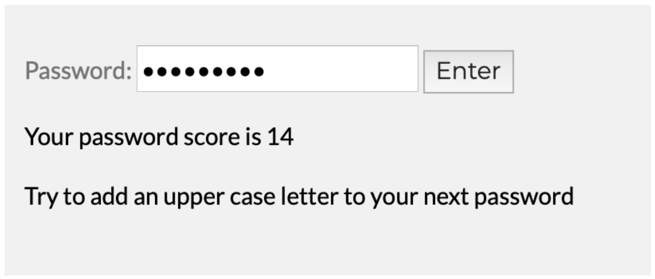
Based on a given password a participant would be given suggestions for their next password. Using the example of “orange” from earlier as the seed phrase--the example password “orange123” would be given a score of “14” based off the formula discussed in section 3.4. They would also be given a suggestion based on the same formula--in this case the suggestion would be to add an upper-case letter in their next password. The order in which suggestions were given were done in order of priority of the formula, and only one suggestion would be given at a time based on the priority (therefore adding an uppercase-letter would be the suggestion given in this example).

One key implementation is the removal of certain "common" password phrases. This includes words like "password" and "iloveyou." This acts as a small password substring blacklist, as described by Shay et Al [2]. Following in the work of Shay et Al is the necessity of not only a length-only requirement.

Notably, the last day of the study did not have any suggestions after a password was scored. This is to test our hypothesis that subjects' password strength will grow over the period, even without suggestions.

Suggestion	**Requirement**
Add uppercase-letter	No uppercase letters
Add special character	No special characters
Add numbers	No numbers
Longer password	Length of password < 6
Avoid common phrase	Use of common phrase
More unique characters	Unique/Length < 0.7
Avoid consecutive characters	Two consecutive characters in a row

Figure 4. Password suggestions and what triggers them



Your password score is 14

Try to add an upper case letter to your next password

Figure 5. Website example for password "orange123" (Notice the suggestion to add an uppercase letter)

4. METHODOLOGY

4.1 Recruitment

The target demographic for this study would be undergraduate students at the University of Chicago. The process involved going to several University of Chicago undergraduate-affiliated groups and making posts asking for study participants. Using this method we were able to find 6 subjects.

4.2 Consent Considerations

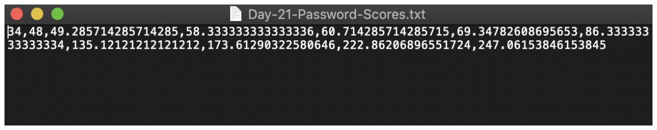
After finding the 6 subjects all the subjects were sent a consent form detailing what they would be doing within the study and their contributions.

- They were not told specifically what the point of the study was. Knowing that the study was trying to test to see if they would make better passwords by the end could change our results.
- They were told what their task was going to be and how their data was going to be collected.
- They were informed that no identifiable information or personal information would be published in the study.
- All data that was collected would also not be identifiable or traceable back to them.
- We would not store any of their password information except for a numerical score.

After signing consent forms, participants were given a link to the study website described in section 3.

4.3 Data Collection

After finishing writing 10 unique passwords for the day either using the seed phrases or not depending on what day of the study it is, a button will appear below the password box telling participants to download their scores. The button will download the 10 password scores of that day as a **.txt* file on the subject's computer.



```
34,48,49.285714285714285,58.333333333333336,60.714285714285715,69.34782608695653,86.333333
33333334,135.12121212121212,173.61290322580646,222.86206896551724,247.06153846153845
```

Figure 6. Example text file and data that is downloaded

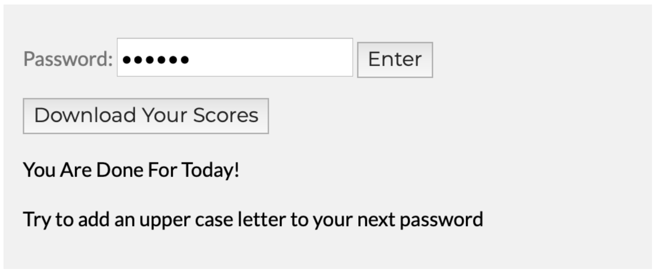
The subject will be notified by their browser that it is a basic text file, and this is also explained in the consent forms described in section 4.2. Once the file is downloaded, the file should be sent to either one of the researchers through their email.

There are several important considerations for why this is the means of data collection.

- The data is in the hands of the subject: this allows the participant to see exactly what data is being sent to us, and so that they know what data is being collected and used in the study.

- As described in section 3.1, Github Pages is a client-side application, and therefore it would have been extremely cumbersome to store the subjects' data without the subject supplying it to us themselves.

Despite Github Pages being client-side, there are ways to have stored the data much like a server-side application would have through a database. We decided to forego this process instead for the first reason explained above. When it comes to password privacy and security, we wanted subjects to feel like they were in control of their data, especially because we had no control over what exactly participants were writing except for the seed phrases. This means that a participant could have used some of their real application passwords, and for this reason we made sure to make it clear to the subjects we would never store their passwords. We also told them, however, that they should not be using their real personal passwords at all during the study, and of course to use the seed phrases when given.


Password: •••••• Enter

Download Your Scores

You Are Done For Today!

Try to add an upper case letter to your next password

Figure 7. The download your scores button after the 10 unique passwords are written

5. FINDINGS

5.1 Data

We had two different groups of data to test in our set. First we had the set of all password scores when the suggestions were available for the first 4 days. Then we also had the scores after the suggestions were turned off and the participants had trained for 4 days. The means and standard deviations of both groups are shown in the table below.

	Training (First 4 Days)	Test (Last Day)
Mean	38.33721566	48.19685152
Standard Deviation	38.81786998	39.07293521
Sample Size	209	55

Figure 8. Table comparing mean and standard deviations between first 4 days and last day of study

5.2 Explanation of Difference of Means Test

To see if there was a statistically significant difference between the training and test scores, we used a *difference of means t-test.* This kind of test is common in drug trials when comparing results of placebo groups to groups that got the drug. In our case, we are comparing the subjects before and after the training which is similar. *The difference of means t-test* works on distributions that are approximately normal, which in our case seems true logically and also experimentally, as our dataset for each day looked like a normal distribution as we can see for Day 1's frequency histogram below:

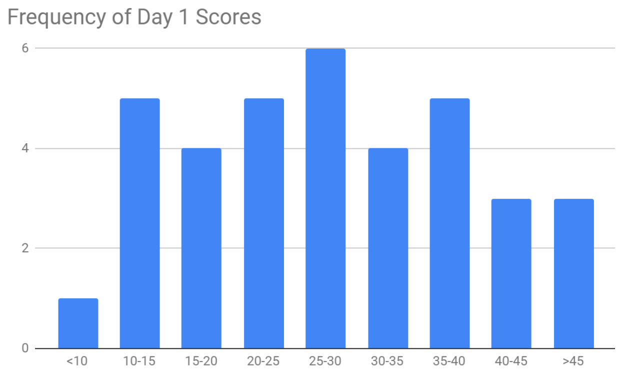


Figure 9. Frequency of scores from Day 1 (Note how it resembles more closely a normal distribution)

The hypothesis in our *difference of means test* was that the mean of the training dataset was at least as high as the testing dataset, and the null hypothesis was that the mean of the training dataset was less than the test dataset. To determine the p-value in a *difference of means test*, a t-score is constructed with the formula:

$$t = \frac{\mu_{train} - \mu_{test}}{\sqrt{\frac{\sigma^2_{train}}{trainsize} + \frac{\sigma^2_{test}}{testsize}}}$$

This is then translated to a p-value from consulting with t-tables and if the p-value is less than 5% then we determine that test mean was greater than the training mean--meaning our tool helped users create better passwords.

5.3 P-Value for T-Test

So if we do a *difference of means t-test* on this data with a null hypothesis that the training mean password score is at least as high as the test, and the alternative hypothesis being the test mean is greater than the training mean, we get a t-score of **-1.67** which corresponds to a p-value of **4.96%** which is significant at the 5% level, so we can reject the null hypothesis and conclude that the users became better at creating passwords after training with the password suggestion tool.

5.4 Graph of Improvement over time

We can also see the change in means of the scores by day to see the day by day progress of the users in Figure 10 below.

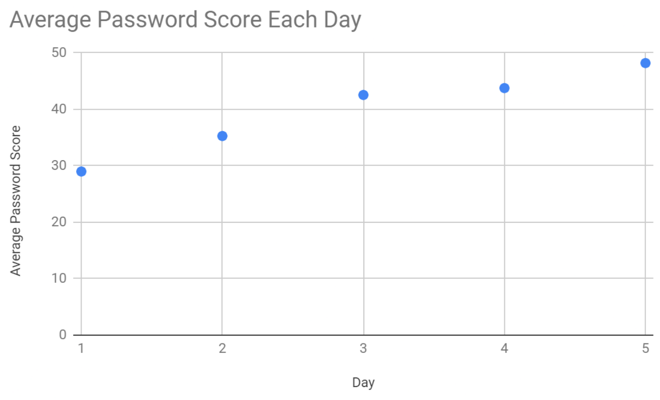


Figure 10. Average password score on each day (Notice the upwards linear pattern)

While this is not a statistical observation as our earlier difference of means test was, we can see users improving their password creating skills over time.

We can also graph the improvement from the first day to the last day as shown below. This shows a significant increase in password scores even when there were no suggestions given on the last day.

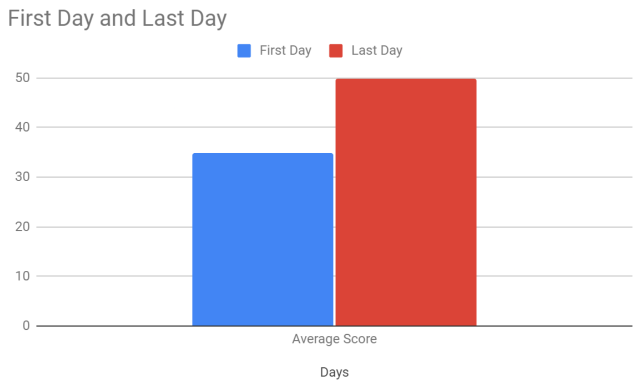


Figure 11. Comparison of the average score from last day and first day

6. DISCUSSION

6.1 Summary of Findings

In our study, we found that after 4 days of training with our password suggestion tool, users became better at creating passwords based on our own evaluation of password strength. This improvement was statistically significant at the 5 percent level. Our study provides a way for existing password managers or password strength bars to help users improve their passwords with suggestions on how to make them stronger. We have shown that users do improve their password creating skills when they are given tangible suggestions on how to strengthen their passwords. These suggestions clearly improve a users' ability to take their password security into their own hands without relying on password managers to make secure passwords as password managers are hard to manage across many different devices, and users who know how to create better passwords will be better prepared to keep themselves secure in environments where password managers can't be used.

6.2 Limitations

There are several notable limitations for this study, especially in its current small scale. The most notable limitation is the time in which the study was conducted. Though the data shows a significant enough increase in subject password scores, it is still difficult to draw conclusions due to the limited time in which the study took place. Because of this limited time frame, it is possible that in a week's time, the subject forgets the ability to make a more complex and stronger password, thereby making the training fruitless.

If the study were to be conducted on a larger scale, there would be several remedies to this problem--one of them being to have participants do the task once again in some sort of time-frame after the conclusion of the main part of the study. This would be to test if there is any “sticking-power” in the repetition of performing the original task.

In the 5-day study, 300 data points were collected, which is a decent amount of data for this small-scale study, but to test if “password training” is a viable method to increase password strength and knowledge retention would require the aforementioned test post-study, and also a much larger and comprehensive data set.

This more comprehensive data set would require a recruitment process that does not limit the study group to a certain common demographic--in this case University of Chicago undergraduate students--but to much more broad groups. We think the findings of this study are particularly interesting in the context of the elderly, who could benefit from increased password strength training and knowledge. Surveys should be conducted getting this demographic information on the subjects on the larger-scale, unlike our study where we did not collect demographic data--except for the fact that we knew all subject members were University of Chicago undergraduates.

On a larger scale, a more rigorous algorithm could be created to test for password-strength, using any of the aforementioned methods in Shay et Al [CITE], like the “2class12” or “3class12” password strength signifiers.

6.3 Lessons Learned

The general hypothesis of “is it possible to train people to make stronger passwords?” seems to have some level of general support from the data. It also seems that including suggestions on a website when creating a password has a tangible impact on increasing a user’s password strength--possibly over time.

As seen in Figure 11, there was a significant increase in the password scores between the first and last day, meaning that in some aspects the password training was a success in the increase of password strength. The much harder implication is to figure out if this was due to sample bias and if the use of password suggestions is effective at increasing password strength over a longer period of time.

7. CONCLUSION

Current methods for password management include the creation of generated “strong passwords” for users, which never gives a user the skills and tools necessary to learn how to create a more secure password themselves. Despite having some knowledge of what creates a secure password, users will seem to take the easiest path forward and create less secure passwords. This is due to a multitude of factors, such as over-valuing certain indicators like password length, rather than the addition of more complex passwords that avoid substrings, have more unique characters, and avoid consecutive repeating characters. In our study, we had participants create 10 unique passwords a day over the course of 5 days and measured the password strength. Based on the results, it seems like repeated exposure to password suggestions allowed participants to create much more secure passwords on the last day than on the first day, displaying some level of password-security information retention.

8. REFERENCES

[1] Blase Ur, Jonathan Bees, Sean M. Segreti, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. 2016. Do Users' Perceptions of Password Security Match Reality? In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16). Association for Computing Machinery, New York, NY, USA, 3748–3760. DOI:https://doi.org/10.1145/2858036.2858546

[2] Richard Shay, Saranga Komanduri, Adam L. Durity, Phillip (Seyoung) Huh, Michelle L. Mazurek, Sean M. Segreti, Blase Ur, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. 2016. Designing Password Policies for Strength and Usability. ACM Trans. Inf. Syst. Secur. 18, 4, Article 13 (May 2016), 34 pages. DOI:https://doi.org/10.1145/2891411

[3] Shannon Riley. 2006. Password Security: What Users Know and What They Actually Do. *Usability News* 8, 1 (February 2006). https://doi.org/10.1.1.597.5846

[4] Ur, et al. How Does Your Password Measure Up? The Effect of Strength Meters on Password Creation. https://www.usenix.org/system/files/conference/usenixsecurity12/sec12-final209.pdf

[5] Guo, Y., & Zhang, Z. (2018). LPSE: Lightweight password-strength estimation for password meters. Computers & Security, 73, 507–518. https://doi.org/10.1016/j.cose.2017.07.012

[6] Golla, M., & Dürmuth, M. (2018). On the Accuracy of Password Strength Meters. Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. Published. https://doi.org/10.1145/3243734.3243769

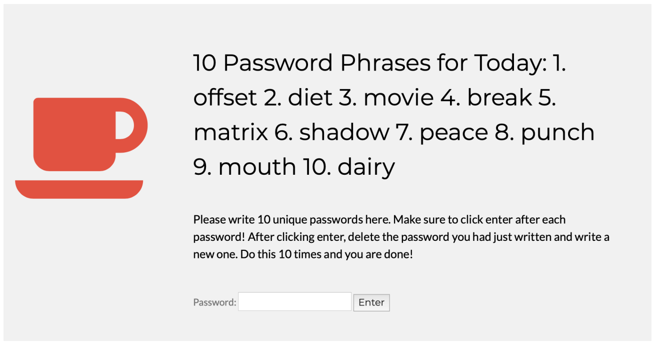
9. APPENDIX

Github Pages link: https://pages.github.com

W3Schools Free Open Source HTML and CSS Template used for the website with modifications: https://www.w3schools.com/w3css/w3css_templates.asp

Link to Study website: https://stmonty.github.io/FastPass

Link to Study Github Repository (all code used in the website prototype): https://github.com/stmonty/FastPass/tree/gh-pages



10 Password Phrases for Today: 1. offset 2. diet 3. movie 4. break 5. matrix 6. shadow 7. peace 8. punch 9. mouth 10. dairy

Please write 10 unique passwords here. Make sure to click enter after each password! After clicking enter, delete the password you had just written and write a new one. Do this 10 times and you are done!

Figure 12. Example of the website with 10 “seed” phrases