Estimating Password Strength Using *zxcvbn*

Group 12

Branko Bajic *10134885*

Ana Carrocci *30003047*

David Keizer *10175086*

Edison Mendoza *10160301*

Jaedwin Montal *30012238*

***Abstract*—Password meters use varying password policies to estimate password strength. Due to this it is difficult to end-users to choose good passwords. The zxcvbn password strength meter uses dictionaries and rules to estimate how long it would take an attacker to break the password. However, it does not include n-gram frequency in its estimation. By creating an extension of zxcvbn that supports n-grams, end-users have the option of using a third party to more accurately estimate password strength. This paper details the research behind the creation of zxcvbN-grams.**

***Index Terms*—password, entropy, strength, meter, n-grams**

I. Introduction

Text-based passwords have become the most prominent method of authentication in the digital era. More often than not, there are individuals who either have no knowledge of what constitutes a reliable password, or they may simply have no motivation to create a good one, and are perhaps too ignorant to care if the information they are storing stays secure. It then becomes a form of responsibility for these services to guide and ensure that their users are creating strong and effective passwords. These services predominantly use password strength estimators (or password meters) to do so. These password meters are normally colored-bars that indicate if a given password is strong or weak when creating an account. Most of these password estimators involve very basic policies that either guide or restrict the user through a set of rules or standards, in order to create their password.

However, according to Dan Wheelers findings in his article [1], these password meters may not be as reliable or effective as we think. For example, he found that the password *qwER43@!* (a seemingly convoluted and complex password) is weak according to PayPals password meter, while eBays own meter judged it as strong. Moreover, in the research of de Carnavalet and Mannan [2], they stated that meters from several high-profile web services (e.g., Google, Yahoo!, PayPal) are quite simplistic in nature and bear no indication of any serious efforts from these service providers, to actually protect their users sensitive information.

This is an alarming precedent, since this creates the possibility of an user being led into believing that a password they created is strong, when in reality, this may not be the case. This is a concerning security risk, especially when we consider Mark Burnetts findings [3] that, while online attacks are difficult, there are enough people with enough weak passwords that they will always yield results. Thus, ineffective password meters could mean potentially easier and/or a higher number of attacks from potential attackers. As such, having a strong and secure password has become as important as ever in our digital age.

In an attempt to partially address the weaknesses of password meters, we examined several existing password estimators, in order to identify which one was the most comprehensive. From there, we were able to create an expanded version of this tool, which retained the lightweight design from the existing tool, while adding crucial functionality in estimating password strength effectively.

II. Problem

With this precedent in mind, our group's evident first steps were to delve further into this issue of weak and inconsistent password meters, policies and/or guidelines that have been occurring among many popular websites and online services. We have found three research papers spanning from 2011 to 2015 that shed light on how grave of a predicament this has been throughout the years. The contents of these papers and their findings on this problem of password meter weakness and inconsistency are briefly highlighted below:

- Furnell (2011). Furnell looked at 10 websites from Alexas Global Top Sites list, including Google, Facebook and Wikipedia, as based on their traffic ranking, and assessed their password meters, policies or guidelines (e.g., if they provided guidance or imposed restrictions, means of password recovery, etc.) and other pertinent characteristics that they may possess. Upon assessment, he concludes that, since 2007, there has been no clear improvement from these websites password practices, despite the increase in their number of users, stating that, aside from a few, most of the sites offered nothing beyond standard password protection and that they are not implementing it as well as they could do [4].

- de Carnavalet and Mannan (2014). They evaluated the password meters of 11 prominent web services, ranging from financial, email and messaging services. They analyzed their password meters and extracted code to study the algorithms they may use, and even plugged them into a custom dictionary-attack algorithm. Then, they reported the prominent characteristics that these meters deployed. Upon analysis, they found that the password meters they analyzed are highly inconsistent, fail to provide coherent feedback . . . and sometimes provide strength measurements that are blatantly misleading. As a final note, they comment on the implementation of most of these password meters, stating that they bear no indication of any serious efforts from these service providers [2].

- Wang and Wang (2015). They decided to perform an empirical analysis of password creation policies imposed on high-profile websites, including 20 from the United States and 30 from China. They analyzed each of the policies through a systematic, evidence-supported approach and they tested them through custom guessing attacks. They concluded that these policies among leading websites: (1) are highly diversified, citing that no two sites enforce the same password creation policy (i.e., among the 50 sites, there were 50 distinct policies), and (2) are unable to serve their purpose of securing user information, and are therefore vulnerable to targeted online guessing attacks [5].

Therefore, as we can see, there is incontrovertible evidence that there has been a history of inconsistency and a slew of weaknesses among the password meters of popular websites through the years. Thus, from these studies, we can highlight a major problem, in that the password meters, policies or guidelines of popular, high-profile, high-traffic websites are highly inconsistent, give incoherent feedback and/or are not well-implemented. This poses a great security risk for users of these web services since:

1) this can lead to users creating weak, insecure passwords, or
2) it may give users the wrong perception that a password they are creating is strong or secure enough, and, most importantly,
3) it becomes easier for attackers to guess and crack these users passwords. It would therefore be pertinent to re-analyze and scrutinize the efficacy of password meters deployed by a few selected popular websites, as there has been noteworthy growth in the online world, with the massive growth of social media and online businesses.

III. Proposed Work

As it is clear that there is a problem of inconsistent on password meters among numerous popular websites, our group has worked in three phases to perform our work for our project, which are detailed below:

Phase One: Research

Using the above three studies as a reference, we studied and analyzed the prominent policies, characteristics and guidelines of the password meters utilized by a selection of popular websites. We then tested these password meters and policies by using common, unreliable passwords as input. From these, we compared results among the websites and we tried to analyze and see if there are still inconsistencies among these popular, high-profile websites. Then, we compared our findings to the aforementioned research papers and noted if there are any inconsistencies and weakness that are still currently occurring and should be highlighted, and use these as a reference to our possible improvements by adding n-grams that we can suggest to these websites in their implementation of their password strength calculation metrics.

Phase Two: Implementation

To alleviate the problems of inconsistency and weaknesses among password meters that we have found in our first phase, we then wanted to find a reliable password strength estimator that can be used as a standard that these popular web services can emulate or deploy themselves or, at least, encourage them to use this tool as a future point of reference. Thus, in analyzing this tool, we wanted to:

- Fully understand the tool and the algorithm it uses.
- Use the tool ourselves, and modify it by adding n-grams.
- Find a noteworthy and real-world list of leaked passwords that can be used to achieve the tools efficacy and efficiency.

Phase Three: Comparison

Finally, once we had made our proposed changes to the chosen password meter, we wanted to see if our changes actually made a difference or not by comparing our modified tool with existing password meters, in particular the one we chose to make changes to. To do so we would have to come up with metrics for how to evaluate the strength of password meters in addition to coming up with a list of passwords to test each password meter with. Finally, we would take our comparison results and try to come up with useful conclusions based on them.

Therefore, the overall goal of this research is to propose possible ways of improving the existing meters that currently popular websites use. In consequence, we expect that password meters will become actual effective tools for measuring a passwords strength, and help those who may be unaware of the proper techniques for formulating strong passwords. So that, users are able to actually protect their sensitive data from possible attacks.

IV. Findings, Experiments and Results

Phase 1: Research

1) High-traffic Websites

To begin our work, we first looked at five prominent and popular websites whose password meters we want it to analyze. We choose them based on the latest Alexa rankings [6] as based on each websites traffic (i.e., users daily time on the site, daily visitor page views, other websites that link to it). Also, we decided to choose web services that vary in their specific use, so we consider social media, business, and email or messaging services. With these in mind, we have chosen to work with the following websites presented in [1, Tab. 1]

We have chosen the five websites as they are among the most popular and dominant websites that many web users are utilizing today. We decided a conservative number of websites firstly due to time constraints, as we felt it may be unfeasible to collect and analyze data for a higher number of sites. We feel, however, that the websites we selected were diverse enough, so that we chose a range of websites for different types of online services, and they are too representative enough for the purposes our project. We believe that looking at these five prominent websites will satisfy our motive of showing if there are still inconsistencies or weaknesses among their password meters, policies or guidelines.

2) Comparing Password Meters and their Policies

Now, we take a closer look at the characteristics of the password meters policies and/or guidelines that these five high-profile websites use. We investigate how each of these websites guide or restrict users as they create their passwords and list the attributes they have. Since, our goal is to see if there are any main changes in these websites password meters in terms of guiding users through password creation, as compared to the findings found by the three research papers we mentioned before. We want to see if they have either improved their policies, or if they remained the same. Also, we want to note if there are any peculiarities that they may possess, which can help us in pinpointing inconsistencies or weaknesses that they may have. We will observe the following common characteristics or requirements:

- Strength meter or scale: We want to see if these websites provide any visual or textual meter or scale, that illustrates users password strength.
- Length limits: We want to check if there are policies or guideline that restrict the minimum or maximum length of a users password.
- Character set requirements: As above, in this case we want target if the policies or guidelines require or force the user to use at least one of the following: digits, uppercase or lowercase letters and/or special symbols.
- Allows user information: We want to observe if it is possible for the user to enter personal information as their password, such as the users name or birth date.
- Feedback for user: If the password is accepted/rejected, we want to see if there is any form of feedback about the password that the user is currently creating.

Analysis of results. Upon observations of the password meters, policies and guidelines in [2, Tab. 2], we have found the following:

- All of the websites do not use some form of scale to let the users visually see the rating of their password.
- Most of them employ a minimum limit of 6 characters. Similarly, most of them do not enforce a maximum length. Coincidentally, both limits in minimum and maximum length that differed are from Google and Amazon. Perhaps, as this two companies encompasses a wider range of services, thus, they restrict the maximum length of their users passwords, which may decrease security, considering an attacker knows at least the maximum length of the password.
- None of the above websites request the user to include at least one special character in a their password. However, from our testing, it seems that for Facebook special characters are required in some way, but there was no blatant messages anywhere indicating the user that these are needed for their password.
- Three of the five websites allowed user information to be used as a users password. This is quite worrisome since an attacker who can easily gain a users private data, by fishing for instance, can easily use this to guess a users password. It is even more concerning when, according to NIST findings [7][8], users tend to use personal information as their password for better memorization. However, users personal information should not be allowed to be used as their password.
- Two out of the five websites did not give any form of feedback. Appropriate and coherent feedback is important to users as this can lead to less user frustration as they are properly guided in creating a secure password.

Therefore, from the above findings, we can clearly see that there are still significant inconsistencies in the way that password meters, policies and guidelines are implemented even for five popular websites. From the set of characteristics, rules or requirements that we observed and tested, none of the websites

unanimously agreed upon a specific way to enforce password strength. Moreover, from the studies of Furnell [4] and de Carnavalet and Mannan [2], who have concluded that there does not exist a greater level of consistency between the practices of leading sites and that commonly-used meters are highly inconsistent, despite the changes in technology during the last couple of years, big companies like Facebook or Google has not improve their password meters, guidelines and/or policies. Consequently, the findings related to these password meters and policies are still highly inconsistent and there is no standard method of enforcing or improving them.

3) Testing Password Meters

NIST guidelines [8] recommend that a blacklist of leaked or common passwords (of the appropriate size) should be used by password meters and password creation policies in order to prevent users from using such common and easy to guess passwords, which are very vulnerable and ineffective against attacks. As a straightforward experiment, we want to test if our five websites, during account creation, will succeed in preventing a user from using any of the top 25 passwords found in the annual lists of the 100 worst passwords, as published by SplashData, Inc., a provider of security applications and services [9], which releases a list of the top worst 100 passwords yearly based on data they have examined from millions of passwords leaked in data breaches. As a experiment, we used the latest published list from 2017 and the top 25 passwords on the list. Our assumption is that all five websites should unanimously reject these passwords, as these passwords, by security standards, are the most ineffective in protecting user information. The results are listed in [3, Tab. 3]

Analysis of results. Upon analyzing the results in [3, Tab. 3], one glaring finding is that Amazon, one of biggest online retailers today, which stores multitudinous amounts of personal and financial information of its users, on account creation, allows 22 out of the 25 top worst passwords of 2017 list to be used as a users valid login password. Even if we put into perspective the supposition that Amazon is greatly securing their servers from potential user information breaches, it is still a bad precedent that one of the highest-profile websites is allowing the most common of passwords to be accepted. It is also crucial to note that the 3 out of the 25, Amazons password meter/policy rejected was merely based on the fact that they were less than 6 characters long. This is truly a worrisome case, and serves as a bad model of password security. However, for the other four, aside from a few considerable slips, most of the 25 passwords were rejected. Nonetheless, it should again be noted that, as per our assumption, since these 25 passwords are the worst and most commonly used passwords and if NIST standards are followed [8], the results should have instead been that all these websites must reject any of these passwords during account creation.

4) Using N-grams to Estimate Password Strength

From our previous findings, we observe that very little work has been performed on password meters policies to enforce better password strength among users. This is mostly because people tend to choose passwords that are easier to remember, than complex and more secure passwords compose of random combinations of characters. Thus, the most simple and worst passwords keep giving attackers access to users sensitive information, and users keep using the worst passwords because they are words, numbers, and/or symbols that are easier to remember.

Following that order of ideas, we have observed that some password meters on high traffic websites do not enforce more controls through their policies, which guide users to believe that the passwords they are creating are secure against attackers. Also, this policies that are currently in place are not enough, since, users surpass them by adding a symbol, a number, and an uppercase/lowercase letter at the beginning or end of their passwords, but they keep simple, common sequences of words in place. Consequently, passwords remain insecure and weak against many forms of online attacks.

Hence, we chose to develop our work around word n-grams, considering users prefer to use memorable and common sequences of words when creating their passwords. We aim to show how the commonality of these frequencies can affect passwords strength. To be more precise, we will consider word n-grams has any logical sequences of words from a given string [12], which probability can be computed following the assumptions of the Markov models. Basically, this model predicts the probability that the next word, from a sequence of words, might be present in a users password [13]. Thus, we can approximate a future event by considering all recent ones. In this case, all possible events will be the words contained in the password, which sequence follows the consistency of the human speech [14].

D. Jurafsky and J. Martin in their book Speech and Language Processing state that the Markov model

pursues the following probability:

$$P(W_1^n) \approx \prod_{k=1}^{n} P(W_k|W_{k-1})$$

The above formula, assumes that based on the probability of one word, we can compute the probability of a complete word sequence. Then, we evaluate n-gram probabilities by applying maximum likelihood estimation (MLE) to estimate the parameters of an n-gram model by getting counts from the sentence, and normalizing the counts so that they lie between 0 and 1.

$$P(W_n|W_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

The MLE estimation divides the observed frequency of sequence by inspecting the frequency of a prefix [15]. Jurafsky and Martin called this rate, relative frequency. Essentially, we can use the Markov Model and the MLE normalization to predict n-grams as a defensive tool to improve password meters estimation, and give users a more accurate estimation of their password strength. Therefore, including an improvement of this type on password strength evaluators might increase accuracy on password meters to estimate password strength.

5) Summary of Findings

Therefore, even from our study on a moderate yet diverse number of popular websites based on rankings by highest user traffic today, we have found the following:

- There are still striking inconsistencies in guidelines and policies during password creation.
- The basic requirements for a users password, are mostly similar among all the websites we tested, despite a few differences, we have observed that there is still disparities in what is considered a weak/strong password.
- There are still popular websites today, which have not implemented any form of protection against the simplest of attacks, by preventing users to use, at least, any password from the top 100 worst passwords list.
- There are still highly inconsistent outcomes under the same password testing procedures, and consequently they have weak implementations.
- There exists the possibility of adding more accuracy to password meters policies and guidelines, by simply adding word n-grams checking on passwords estimators.

Therefore, there is a real and urgent necessity for a standard or model that can or should be used in the implementation of these websites password meters, policies or guidelines.

Phase 2: Implementation

1) Choosing a Reliable Tool

As per our analysis and testing from the first part of our project, we have observed that there are still prevalent inconsistencies and weaknesses among the password meters, policies or guidelines of highly popular websites. As such, since there is high chance that many users visit and use these same websites simultaneously on a daily basis, a lack of standardization in terms of their password creation policies is an unsettling problem since this can lead users into falsely believing that the passwords they are creating are strong, and thus, become more susceptible against online attacks. With this in mind, there should be some feasible, standard password policy, meter, or set of algorithms that should be used consistently among all websites today, or at the very least, serve as a point of reference that other password strength meters and estimators should strive to emulate, so that users get consistent and logical feedback on their choices when creating passwords for the many existing web services today.

For this project, we analyzed an open-source password meter, the zxcvbn tool created by Dan Wheeler, a password strength estimator, which decomposes an input password into various patterns, checks various dictionaries, and assigns and computes various entropies for each of them, and then computes a final total entropy for the given password. In de Carnavalet and Mannans paper, they deemed that zxcvbn yields more accurate strength evaluations [10] than the other meters they studied. Thus, as we were able to find actual source code of zxcvbn, we have chosen to proceed with looking further and analyzing zxcvbn, and see if it could be a possible candidate for this standard/model password meter that we are seeking.

2) Understanding zxcvbn and it's Advantages

For over 30 years, most password requirements have been a ramification of 4 particular common flawed guidelines: Lower and Uppercase characters, Digits and symbols, or LUDS. However, despite being obsolete and futile in terms of its security implementations, LUDS are still pervasively used today [11]. Thus, D. Wheeler implemented zxcvbn, as an alternative, open-source password strength estimator, which is no more burdensome to implement than LUDS. zxcvbn employs many leaked passwords and dictionaries to simulate guessing attacks to compute the strength of passwords.

How does zxcvbn works? To estimate a passwords strength, zxcvbn goes through the following three phases:

- Match: The first phase is responsible for finding any kind of patterns given a particular password. It incorporates other commonly used password patterns such as tokens, sequences, repeats, keyboard patterns, dates, and brute-force matches during the matchmaking.
- Estimate: The second phase calculates the strength of entropy of each of the matched patterns found from the first matching phase. During this phase, zxcvbn estimates the attempts needed to guess other pattern types.
- Search: The third and final phase is responsible for finding the lowest and simplest entropy of all given matched patterns. By having both a string password and a set of overlapping matches S, the last step is to search for the non-overlapping adjacent match sequence S that covers the password and minimizes expression by using several algorithms.

Within this threefold process, zxcvbn breaks down a given password into several patterns, and assigns each matched pattern an estimated entropy. Then, the total entropy is calculated as the sum of the calculated entropies for each of the given inputs matched patterns. However, the algorithm only keeps the lowest among all possible entropy summations, so that the strength of a given password is underestimated. The highlight of zxcvbn is its ability of pattern detection, which is performed by considering the following patterns, which are listed below:

- Repeat entropy: Logarithm base 2 calculations of any repeating patterns (e.g. aaaaa);
- Year and Date entropy: Determines if a series of numbers corresponds to some date or year;
- Spatial entropy: Checks for combinations of shifts and turns on a keyboard (e.g. qwerty)
- Uppercase entropy: Checks how many capital letters exist in a password (e.g., Football)
- L33t entropy: Similar to uppercase entropy; checks if any l33t substitutions are made in a password (e.g. p@ssw0rd)
- Dictionary entropy: Parts of a password are checked against and is ranked based on several dictionaries of common/leaked passwords, common English words and common English names.

Then, the summed entropy is matched to a score by supposing that a guess would take 0.01 second, and that an attacker can distribute the load on 100 computers. That is, it assigns a password a score based on computations of an attackers possible guess/crack time (as guided by its entropy calculations). It gives a password a score from 0 to 4: '0' if crack time is less than 102 seconds, '1' if crack time is between 102 and 104 seconds, '2' if crack time is between 104 and 106 seconds, '3' if crack time is between 106 and 108 seconds, and '4' if crack time is beyond 108 seconds. The harder it is to guess the password, the higher its score will be. Therefore, we can see how straightforward yet complex the calculations are that zxcvbn goes through to measure a passwords strength.

Moreover, aside from the actual complex and intricate process that zxcvbn goes through to estimate password strength, the following advantages that it possesses also gives this tool even more credibility and support towards being a password meter that many, if not all, websites can deploy or emulate:

- Easy to adopt: zxcvbn is readily available through various Github repositories, available in 16 different programming languages and their variations. Also, the file sizes of these implementations only range from 2 to 3MB. Thus, it is evident that development and modification of zxcvbn is highly encouraged.
- Requires minimal storage: It requires about 1.5MB of compressed storage data to estimate some of the best-known guessing attacks for a maximum of 105guesses, about 245kB for 104 guesses, and 29kB for 103 guesses.
- Runs fast: Despite the intricacies and processes it goes through to estimate a passwords strength, zxcvbn does so within milliseconds.
- Constantly updated: zxcvbn is occasionally being improved upon and updated, with bug fixes and updates which can be easily fetched with a few lines of code.

3) zxcvbN-grams and Summary of Findings

One of the key weaknesses that zxcvbn does have, was stated in the drop-box blog article by D. Wheeler. He explained that the tool has some main limitations when estimating password strength due to its lack of support of n-grams [1]. Therefore, our group worked with zxcvbns code and was able to respond to this limitation. To improve the performance of zxcvbn we extended its codebase to create zxcvbN-grams, which is a version of zxcvbn that includes n-gram support. To do this, we sourced n-grams data files [16], which contained bi-grams, tri-grams, and further n-grams, the longest containing n-grams of five words. Each n-gram also contained a relative frequency. After combining the various n-gram files into a master n-gram list, we parsed the file, sorted the n-grams by frequency, and constructed another dictionary that zxcvbn could then use during the match phase.

Consequently, due to the large number of n-grams included in this dictionary, zxcvbn assigns each n-gram a high entropy. This is important because

during the search phase, as previously stated zxcvbn will choose the match that has the lowest entropy, which is effectively taking the min-entropy of the match results. As a result, the shorter n-grams will often not be used by zxcvbn as it will build up the n-grams using the small existing dictionaries, which provides a lower entropy score. This means that the new version of zxcvbn, zxcvbN-grams, will have the greatest improvement when working on longer n-grams. Thus, our modified version of the code now catches passwords that contain word frequencies n-grams, such as *Ababyinthehouse* and *Hundr3dsofmillionsofdollars$*. By adding a word n-gram dictionary to the code, we were able to improve password estimation and reduce the entropy of any password containing complete and grammatically correct sentences.

Therefore, zxcvbN-grams new support for n-grams, it is a good candidate that can be used as a standard among password meter, as it provides all the benefits of using zxcvbn. It provides more security, when estimating password strength through a complex set of calculations. It is more flexible, considering that it allows many password styles to flourish, given enough complexity; and it is more usable, as it is implemented to be easily adopted to simple, rule-free interfaces that provide instant feedback to users creating a password. The addition of n-gram support is crucial in effectively and accurately estimate the strength of passwords.

Phase 3: Comparison

1) Performance of zxcvbn vs. zxcvbN-grams

To evaluate zxcvbN-grams we compared it to the original zxcvbn tool, to identify if our new tool will show that n-grams are weaker. Table [4, Tab.4] shows a small selection of passwords that were manually chosen to demonstrate the improvements between the two tools.

thetop0fthetow3r, *Adayatthe0ffice!* or *Hundr3dsofmillionsofdollars$* are examples of a longer passwords that zxcvbn indicates are strong passwords, but zxcvbN-grams indicates are weaker, as they shows up in the n-gram dictionary when combined with other rules.

Ababyinthehouse is a far more common n-gram than *housetheinbabyA*, but zxcvbn rates them identically. Our tool zxcvbN-grams correctly identifies that *Ababyinthehouse* is far more likely a password than *housetheinbabyA*, and correctly suggests that the *Ababyinthehouse* is far weaker a password.

GoodPassword123 is an example of a password that neither tool uses n-grams on, due to zxcvbns usage of min-entropy.

magniFicent6doorkob is an example of a strong password that both tools estimate as being difficult to guess. It Uses symbols and numbers in non-conventional ways, as well as an n-gram that is highly uncommon.

To evaluate the performance of zxcvbN-grams, we used the *testzxcvbngrams.py* script to test 100 randomly selected n-grams from our n-gram data, and evaluated those n-gram passwords using both zxcvbn and zxcvbN-grams. By comparing the scores between the two tools for a given password, we can examine which passwords zxcvbN-grams show as weaker when compared to zxcvbn. In addition to comparing the score, *testzxcvbngrams.py* also outputs the guesses log10, as the number of guesses is what zxcvbn uses to estimate how long it would take an adversary to crack the a given password.

Of the 100 randomly selected N-Grams shown in tables [5, Tab.5] and [6, Tab.6], zxcvbN-grams rated 19 of the N-Grams as a less secure password than zxcvbn. Of these 19, only two were bi-grams and N-grams, the rest were n-grams consisting of four to five words. This is expected due to zxcvbns above stated use on min-entropy.

Readability of each cell in tables [5, Tab.5] and [6, Tab.6] is as follows: *Password, zxcvbn score, zxcvbn entropy*

2) Performance of zxcvbN-grams vs. Popular Password Meters

In order to test our program even more, we compared it to other popular passwords meters that are free to use online. To do so, we chose *Hundr3dsofmillionsofdollars$* as the password to be tested throughout these different password meters. This is because *Hundr3dsofmillionsofdollars$* was heavily affected by our version of the zxcvbn as it is a common 5-gram phrase. *Howsecureismypassword.net*, a popular password meter tool online evaluated the password as very strong and estimated that it will take 53 decillion years to crack the password. As well, *passwordmeter.com* and *my1login.com* evaluated the password as very strong with *my1login.com* estimating that it will take 225 years to crack the password. *Kaspersky*, a *cybersecurity* and anti-virus company was the only one to correctly identify the password as weak. Comparisons of these results can be seen in table [7, Tab.7]

V. Changes Between Proposal and Evaluation

Our original proposed experiment included the creation of a password cracker that tested the strength of a particular password system. The system would take in, as input, a

dictionary D of possible values (i.e. English letters, digits, ascii characters, etc.) from that password system, a rule set R which consists of the rules for generating a password in that system, and a reference password file F. The password cracker would then generate a list of passwords L (under a time constraint) using D and R. Finally, it would go through L and see if any of the generated passwords match a password in our reference file F. The output would be the number of matches found (or possibly a security rating based on the number of matches found).

This system was unfortunately deemed improbable to achieve in the limited time frame we had, thus we proposed an alternative experiment and re-evaluated our measurement of success/failure.

Our focus changed to password strength meters, and how the meters, policies or guidelines are highly inconsistent, give incoherent feedback and are not well-implemented. For our new experiment we decided to implement it in three different phases: a research phase, an implementation phase, and a comparison phase. Our new goals included the sequential success of each phase - finding accurate and meaningful data in the research phase, successfully implementing/modifying an existing password meter, and getting comparably better results than the original password strength meter (as well better resulting in general vs. various other meters shown in table [7, Tab. 7]).

These goals were achieved as we successfully implemented an extension of the original password strength meter chosen (zxcvbn) to better accurately show results. These improvements are shown in tables [4, Tab.4], [5, Tab.5], [6, Tab.6].

With the redirection of our project came some challenges. Time constraint was a big one as each member had various other classes and projects while devoting the remaining time to the project. The original source code for zxcvbn was not clear to restructure and took some time to understand how the code worked and how we could potentially implement a solution that would result in an improvement. N-grams as well posed to be difficult as there are various definitions of n-grams (n-gram character frequencies vs. n-gram word frequencies, for our project we chose to use n-gram word frequencies). As well as choosing a limited number of n-grams and deciding what number that would be (in our case, up to 5-grams) to have accurately measurable data in the limited amount of time we had.

VI. Conclusion

Passwords are the most common way to secure our information and data. Therefore it is essential to create strong passwords for different online accounts that we create. Password meters and password guidelines are helpful tools that aid users to create strong passwords. However, the problem with password meters and guidelines is its inconsistency among different websites and weak implementations for some. This leads users to believe that their passwords are strong when they are not, giving them a false sense of security. After examining several password strength meters that are readily available online, we found zxcvbn to be a great realistic strength meter tool. While a great tool, it does have some limitations. We decided to improve zxcvbn with the addition of n-grams. Our version, zxcvbN-grams, improves the tool by detecting common phrases within the password. We were able to make the tool even more powerful while still keeping it lightweight. We came across some issues but we were able to finish the implementation and found the results to be very promising. Throughout this project, we have gained a deeper appreciation for n-grams and its importance in choosing a good password.

While zxcvbN-grams is an improvement on the existing zxcvbn, it can still be further developed. As it stands, zxcvbN-grams only supports the english language. Expanded support to other languages is important if this tool is to be used on a global audience. Furthermore, the tool currently does not tolerate misspellings of a word[1]. The development of this would greatly increase the strength of the tool as it would be able to detect passwords such as *assword* into a misspelling of *password* instead of evaluating it as two words, *ass* and *word*.

Finally, from our observations and analysis, we found that zxcvbN-grams, is a good open-source alternative as a password strength estimator, since it can truly and accurately estimate a passwords strength by predicting an attackers ability to guess passwords through the complex methods it performs. We therefore, recommend the use of zxcvbN-grams as a starting point towards this goal of having a standard and consistent manner of accurately assessing and estimating a passwords strength, and thus, aid the users of these popular websites in creating better, more secure passwords in this day and age where attacks on personal information are prevalent.

VII. Distribution of Work

Name	Contributions
Branko Bajic	Code Development Proposal/Final Report Three Questions Presentation
Ana Carrocci	Proposal/Final Report Three Questions Presentation Latex
David Keizer	Code Development Proposal/Final Report Three Questions Presentation Latex
Edison Mendoza	Proposal/Final Report Three Questions Presentation Latex
Jaedwin Montal	Proposal/Final Report Three Questions Presentation Latex

Weekly discussions contributed to the success of our project. Each member was always present and contributed to the discussions that were conducted either in-person, using slack, or using VoIP application called discord.

VIII. References

[1] D. Wheeler, zxcvbn: Realistic Password Strength Estimation, April, 2012. [Online]. Available: https://blogs.dropbox.com/tech/2012/04/zxcvbn-realistic-password-strength-estimation/ [Accessed Oct. 1, 2017].

[2] X. de Carn de Carnavalet and M. Mannan, "From Very Weak to Very Strong: Analyzing Password-Strength Meters," presented at Network and Distributed System Security Symposium, San Diego, California, 2014.

[3] M. Burnett, Perfect Passwords: Selection, Protection, Authentication. Rockland, MA: Syngress Publishing, Inc., 2006.

[4] S. Furnell, Assessing password guidance and enforcement on leading websites, Computer Fraud Security, 2011.

[5] D. Wang and P. Wang, The Emperors New Password Creation Policies, presented at 20th European Symposium on Research in Computer Security, Vienna, Austria, 2015.

[6] Alexa.com. Alexa Top 500 Global Sites. [Online]. Available: https://www.alexa.com/ topsites [Accessed Nov. 15, 2018].

[7] K. Scarfone and M. Souppaya, Guide to enterprise password management, NIST SP800-118, 2013.

[8] W. Burr, D. Dodson, R. Perlner, W. Polk, S. Gupta and E. Nabbus, Electronic authentication guideline, NIST SP800-63, 2006.

[9] M. Slain, 100 Worst Passwords of 2017! The Full List teamsid.com. [Online]. Available: https://www.teamsid.com/worst-passwords-2017-full-list/ .[Accessed Nov. 16, 2018].

[10] X. de Carne de Carnavalet and M. Mannan, A large-scale evaluation of high-impact password strength meters, ACM Transactions on Information and System Security, 2015.

[11] D. Wheeler, zxcvbn: Low-Budget Password Strength Estimation, presented at USENIX Security Symposium, Austin, Texas, 2016.

[12] Yulong Yang, Janne Lindqvist, and Antti Oulasvirta, Text Entry Method Affects Password Security, presented at USENIX Security Symposium, San Diego, California 2014

[13] Claude Castelluccia, Markus Durmuth, and Daniele Perito, Adaptive Password-Strength Meters from Markov Models, presented at NDSS Symposium, San Diego, California, 2012

[14] Jay Destories Probabilistic Password Modeling: Predicting Passwords with Machine Learning [Online]. Available: http://www.cs.tufts.edu/comp/116/archive/fall2015/jdestories.pdf. [Accessed Dec. 04, 2018].

[15] Daniel Jurafsky and James H. Martin Speech and Language Processing. Pearson, NJ, 2018

[16] "N-grams data", N-grams: based on 520 million word COCA corpus. [Online]. Available: https://www.ngrams.info/. [Accessed: Oct-2018].

IX. Appendix

TABLE I
WEBSITE POPULARITY
[1, TAB.1]

Alexa Ranking	Website	Description
1	Google	Web Portal
3	Facebook	Social Media Website
10	Amzon	International Online Retailer
12	Twitter	Social Media Microblogging Website
28	LinkedIn	Business Networking Website

TABLE II
ZXCVBN VS. ZXCVBN-GRAMS RESULTS (1-50)
[2, TAB.2]

	Google	Facebook	Amazon	Twitter	LinkedIn
Scale or Meter	None	None	None	None	None
Minumum Length	8	6	6	6	6
Maximum Length	100	None	128	None	None
Charset Requirement	None	None	None	None	None
Allow User Information	No	Yes	Yes	No	Yes
Feedback or notes	Tells the user to not use common words	Specifies if password is too short	Specifies if password is too short	Requests the user to use a stronger password	Specifies if password is too short

TABLE III
POPULAR WEBSITES ACCEPTANCE OF TOP 25 WORSE PASSWORDS OF 2017
[3, TAB.3]

Rank	Password	Google	Facebook	Amazon	Twitter	Linkedin
1	123456	no	no	yes	no	no
2	password	no	no	yes	no	no
3	12345678	yes	no	yes	no	no
4	qwerty	no	no	yes	no	no
5	12345	no	no	no	no	no
6	123456789	yes	no	yes	yes	no
7	letmein	no	no	yes	no	no
8	1234567	no	no	yes	no	no
9	football	yes	no	yes	no	no
10	iloveyou	no	no	yes	yes	no
11	admin	no	no	no	no	no
12	welcome	no	no	yes	no	no
13	monkey	no	no	yes	no	no
14	login	no	no	no	no	no
15	abc123	no	no	yes	no	no
16	starwars	no	no	yes	no	no
17	123123	no	no	yes	no	no
18	dragon	no	no	yes	no	no
19	passw0rd	yes	no	yes	yes	no
20	master	no	no	yes	no	no
21	hello	no	no	yes	no	no
22	freedom	yes	no	yes	no	no
23	whatever	no	no	yes	no	no
24	qazwsx	no	no	yes	no	no
25	trustno1	yes	no	yes	no	no

TABLE IV
ZXCVBN VS ZXCVBN-GRAMS MANUAL SELECTION
[4, TAB.4]

Password	zxcvbn Scores	zxcvbn Guesses Log_{10}	zxcvbN-grams Scores	zxcvbN-grams Guesses Log_{10}
thetop0fthetow3r!	4	15.36231	2	7.91741
Ababyinthehouse	4	11.56714	0	0.47712
housetheinbabyA	4	11.56714	4	11.56714
GoodPassword123	1	5.40271	1	5.40271
Adayatthe0ffice!	4	12.92794	1	4.50634
Hundr3dsofmillionsofdollar$	4	20.0006	2	7.93931
magniFicent6doork&ob	4	13.44297	4	13.44297

TABLE V
ZXCVBN VS. ZXCVBN-GRAMS RESULTS (1-50)
[5, TAB.5]

zxcvbn	zxcvbN-grams
whoplayedtherole 4 11.54543082946535	whoplayedtherole 4 11.54543082946535
enjoysomeof 3 8.04392198022434	enjoysomeof 3 8.04392198022434
youforajob 3 8.397940008672036	youforajob 3 8.397940008672036
youwakeupwitha 4 11.368955945345503	youwakeupwitha 1 5.806562527458186
fieldgoalto 3 8.179223648512155	fieldgoalto 3 8.179223648512155
whathisown 3 8.000043427276863	whathisown 3 8.000043427276863
wasgivinghimthe 4 11.133538908370216	wasgivinghimthe 4 11.133538908370216
mindatthelastminute 4 13.07907549694405	mindatthelastminute 1 5.43866099227323
betterhere 1 4.336459733848529	betterhere 1 4.336459733848529
lotofcapital 2 7.977769318091574	lotofcapital 2 7.977769318091574
highbloodpressureandcholesterol 4 16.052553398436224	highbloodpressureandcholesterol 1 4.299441920802092
bepartofthefuture 4 12.244571092348865	bepartofthefuture 1 5.269641345646085
aboutaprocess 3 8.006232358910914	aboutaprocess 3 8.006232358910914
emergencyward 1 5.102021856924152	emergencyward 1 5.102021856924152
Igotdowntothe 4 11.717753693210714	Igotdowntothe 4 11.717753693210714
thanthefactthatyou 4 13.673941998634087	thanthefactthatyou 1 4.976285770215319
followinghisdeath 3 8.027458881276273	followinghisdeath 3 8.027458881276273
historymighthavebeen 4 12.000432550287245	historymighthavebeen 4 12.000432550287245
oftheCatholicChurchand 4 12.941451902582699	oftheCatholicChurchand 4 12.941451902582699
thedugouts 2 6.406659382166752	thedugouts 2 6.406659382166752
wasdesigning 1 5.668758541750957	wasdesigning 1 5.668758541750957
sidewiththeUnitedStates 4 15.1085136088817	sidewiththeUnitedStates 4 15.1085136088817
TheRosenberg 1 5.414304688128331	TheRosenberg 1 5.4139699717480605
asCongressandthe 4 11.64276120326532	asCongressandthe 4 11.64276120326532
andburgeoning 2 6.021437396467089	andburgeoning 2 6.021437396467089
willnotstandin 3 8.321343204887823	willnotstandin 3 8.321343204887823
wanttoberid 3 9.000004342923104	wanttoberid 3 9.000004342923104
itcouldfall 3 8.041981741549396	itcouldfall 3 8.041981741549396
Thenwehaveto 4 10.178976947293169	Thenwehaveto 4 10.178976947293169
fairtothem 3 8.067591176601178	fairtothem 3 8.067591176601178
RockShoxSID 3 9.244137035147638	RockShoxSID 3 9.244137035147638
shewantedhimto 3 9.627365856592732	shewantedhimto 3 9.627365856592732
qualitiesofa 2 6.7901443650429005	qualitiesofa 2 6.7901443650429005
dayhewasarrested 4 10.475089803389006	dayhewasarrested 4 10.475089803389006
abouttwo-thirdsthesizeof 4 19.12404751911003	abouttwo-thirdsthesizeof 1 5.8144067634647305
ofoureconomicsystem 4 10.72274577185978	ofoureconomicsystem 4 10.72274577185978
LikeWaterforChocolate 4 12.036198229646015	LikeWaterforChocolate 4 12.036198229646015
themilling 1 5.973820324352683	themilling 1 5.973820324352683
howtogetby 3 8.57518784492766	howtogetby 3 8.57518784492766
itsdiscount 1 5.53529412004277	itsdiscount 1 5.53529412004277
thattherewasnosexual 4 12.883888457888478	thattherewasnosexual 1 4.626843159658238
natureofthelaw 4 10.224184185857872	natureofthelaw 4 10.224184185857872
theevidencethatyou 4 12.00083304593682	theevidencethatyou 4 12.00083304593682
aretestifying 1 5.615739688619154	aretestifying 1 5.615739688619154
waytheyalwaysdid 4 11.61076659477327	waytheyalwaysdid 4 11.61076659477327
sworeI 1 4.675044735955892	sworeI 1 4.675044735955892
liquiditycrisis 2 7.60120130281087	liquiditycrisis 2 7.60120130281087
toourinterview 3 8.11833980326113	toourinterview 3 8.11833980326113
currentapproaches 2 6.321710251557297	currentapproaches 2 6.321710251557297
shoulderandhe 3 8.079543007402906	shoulderandhe 3 8.079543007402906

TABLE VI
ZXCVBN VS. ZXCVBN-GRAMS RESULTS (51-100)
[6, TAB.6]

zxcvbn	**zxcvbN-grams**
waitforthegovernmentto 4 13.221800105910283	waitforthegovernmentto 1 5.9513875520788835
tolifeagain 3 8.015208816042277	tolifeagain 3 8.015208816042277
theMassachusettsSupremeCourt 4 12.585726248662326	theMassachusettsSupremeCourt 4 12.585726248662326
abrotherto 3 8.004193216503454	abrotherto 3 8.004193216503454
significantlyandpositivelycorrelatedwith 4 16.389918282685557	significantlyandpositivelycorrelatedwith 1 5.441751676630346
dothatandIthink 4 12.000000004342944	dothatandIthink 4 12.000000004342944
hasoffended 1 5.474361976032631	hasoffended 1 5.474361976032631
noreasonforme 3 8.88178965720335	noreasonforme 3 8.88178965720335
informationinthecase 4 10.803078650250432	informationinthecase 4 10.803078650250432
andPBS 1 5.0413926851582245	andPBS 1 5.0413926851582245
andkneaduntil 3 9.499687082618404	andkneaduntil 3 9.499687082618404
productionofnuclearweapons 4 12.188247383463425	productionofnuclearweapons 4 12.188247383463425
toruboff 2 7.288025535388362	toruboff 2 7.288025535388362
andyouseethat 3 9.895974732359063	andyouseethat 3 9.89762709129044
AREAOF 1 4.471291711058938	AREAOF 1 4.471291711058938
theirhighestlevelof 4 12.004793079523322	theirhighestlevelof 4 12.004793079523322
fourparties 1 5.422983579209279	fourparties 1 5.422983579209279
itdidseemto 4 10.084933574936715	itdidseemto 4 10.084933574936715
Iguesswecouldcall 4 12.634406729365537	Iguesswecouldcall 4 12.634406729365537
suspectitis 2 7.317854489331469	suspectitis 2 7.317854489331469
shooktheirhands 3 8.393947922387415	shooktheirhands 3 8.39375064034808
arethedetails 3 8.062788741900196	arethedetails 3 8.062788741900196
arelationshipand 3 8.006712429115112	arelationshipand 3 8.006712429115112
largeenoughtoholdthe 4 13.611901661289272	largeenoughtoholdthe 1 5.612267693004007
Daywhen 1 4.54654266347813	Daywhen 1 4.54654266347813
liketosaythatthey 4 12.113943352306837	liketosaythatthey 1 5.855376184441496
dowhattheydid 4 10.178976947293169	dowhattheydid 4 10.178976947293169
somethingneededtobe 3 9.283662990058831	somethingneededtobe 3 9.283662990058831
anditssmaller 3 8.057742155828752	anditssmaller 3 8.057742155828752
wasagenius 2 6.911157608739976	wasagenius 2 6.911157608739976
formersuperintendent 2 6.066211092413128	formersuperintendent 2 6.066211092413128
agreatblueheron 4 12.022505804473537	agreatblueheron 4 12.022502521136904
millionbail 1 5.829400271994159	millionbail 1 5.829400271994159
thecostofeducating 4 12.143309245807911	thecostofeducating 4 12.143309245807911
setasidemoneyfor 4 12.017506862929277	setasidemoneyfor 4 12.017506862929277
thesametimeare 3 8.662006608180395	thesametimeare 3 8.662006608180395
theaimsandobjectivesof 4 15.345491356474643	theaimsandobjectivesof 1 5.876180877786232
inthatcity 3 8.003503614742536	inthatcity 3 8.003503614742536
eventsandtheir 3 8.024444617131348	eventsandtheir 3 8.024444617131348
tobuildaround 3 8.144474573074602	tobuildaround 3 8.144474573074602
bothpolitical 1 4.753123244681713	bothpolitical 1 4.753123244681713
toooftenfor 3 8.0210507163411	toooftenfor 3 8.0210507163411
bytakingintoconsiderationthe 4 16.000542741630863	bytakingintoconsiderationthe 1 5.8271339664748405
andafterexercise 3 8.107684719655841	andafterexercise 3 8.107684719655841
Therearetwosides 4 11.799756721204808	Therearetwosides 4 11.799756721204808
centralelements 1 5.609803503153702	centralelements 1 5.609803503153702
heaskedmewhyI 4 10.988112840268352	heaskedmewhyI 4 10.988112840268352
don'ttakeintoaccount 4 12.002032347615101	don'ttakeintoaccount 1 5.580205114791282
tobethemoreimportant 4 11.960042455726839	tobethemoreimportant 1 5.007167193147053
economywiththe 3 8.06177280390343	economywiththe 3 8.06177280390343

TABLE VII
PERFORMANCE OF ZXCVBN-GRAMS VS. POPULAR PASSWORD METERS
[7, TAB.7]

Password Meter	**Evaluation**	**Time to Guess**
zxcvbN-grams	somewhat guessable	2 hours
howsecureismypassword.net	very strong	53 deillion years
kaspersky	weak	n/a
passwordmeter.com	very strong	n/a
my1login.com	very strong	225 years