An Empirical Analysis on the Usability and Security of Passwords

Kanwardeep Singh Walia
Department of Computer Science
California State University, Sacramento
Sacramento, CA, U.S.A.
kanwarsinghwalia@gmail.com

Shweta Shenoy
KLA Corporation
Milpitas, CA, U.S.A.
shwetashenoyk@gmail.com

Yuan Cheng
Department of Computer Science
California State University, Sacramento
Sacramento, CA, U.S.A.
yuan.cheng@csus.edu

***Abstract*—Security and usability are two essential aspects of a system, but they usually move in opposite directions. Sometimes, to achieve security, usability has to be compromised, and vice versa. Password-based authentication systems require both security and usability. However, to increase password security, absurd rules are introduced, which often drive users to compromise the usability of their passwords. Users tend to forget complex passwords and use techniques such as writing them down, reusing them, and storing them in vulnerable ways. Enhancing the strength while maintaining the usability of a password has become one of the biggest challenges for users and security experts. In this paper, we define the pronounceability of a password as a means to measure how easy it is to memorize - an aspect we associate with usability. We examine a dataset of more than 7 million passwords to determine whether the user-generated passwords are secure. Moreover, we convert the user-generated passwords into phonemes and measure the pronounceability of the phoneme-based representations. We then establish a relationship between the two and suggest how password creation strategies can be adapted to better align with both security and usability.**

***Index Terms*—authentication, passwords, phonemes, usability, security**

I. Introduction

Passwords are the most ubiquitous authentication mechanism, but they are not inherently secure. Especially with the ever-growing constraints associated with their creation, passwords no longer remain simple words. Consequently, people fall prey to insecure practices, such as scribbling the passwords on paper [1], reusing them across different accounts [1]–[4], or creating passwords that can be easily guessed [5]. A plausible rationale behind such practices is the stringent password creation policies. They make it challenging for users to create passwords, but do they at least guarantee secure and strong passwords?

Prior research shows that there is little rationale for why organizations prefer to use these policies [3]. In fact, it is difficult to determine a universal standard as different organizations and applications enforce different password creation policies. Studies show that even the most popular guidelines are based on theoretical estimates [6] or small-scale research studies [1], [7].

Historically, these policies were believed to be adequate and were implemented to help prevent password guessing attacks [8]. In those times, computational power was far scarcer, and passwords were not as large of a foothold to adversaries as it is today. Thus, these policies seemed to be reasonable at providing users with some degree of security. Furthermore, these password creation policies act as a fail-safe mechanism to at least prevent users from creating extremely vulnerable passwords [9]. Many researchers support the new guideline published by the National Institute of Standards and Technologies (NIST) [6]. It recommends eliminating or reducing complex rules, such as allowing all printable characters (e.g., whitespaces), increasing the maximum length to 64 characters, and not requiring special characters. However, very few organizations have implemented these guidelines. As a result, strict password policies are still a widespread practice today.

Passwords are supplemented by other authentication methods, such as fingerprint and face recognition, but they are merely an adjunct to passwords, not a replacement. Security experts claim that biometrics facilitate ease of access to systems; on the other hand, passwords are used to establish the initial trust and as a fallback when biometrics fail [10]. In fact, the amount of risk a compromised password constitutes depends on how it is used and what it is protecting [11].

Therefore, we tend to resort back to text-based passwords. Good passwords should be both usable and secure. In this work, we empirically investigate a dataset of about 7.75 million user-generated passwords and aim to find a balance between usability and security of passwords.

The contribution of this work is two-fold. First, we study the strength of passwords of various categories based on their length and components. We also discuss our findings regarding some particular types of passwords, such as passwords containing dictionary words, passwords with repeating patterns, and munged passwords. Second, we suggest how phonemes can be used to estimate password pronounceability. We propose a pronounceability scoring scheme and use it to establish a relationship between the strength and pronounceability of the passwords in the dataset.

The remainder of this paper is organized as follows. Section II discusses the literature relevant to this study. Section III first proposes our methodology and then describes the dataset we used and its characteristics. Section IV delves into some

978-1-7281-1054-7/20/$31.00 ©2020 IEEE
DOI 10.1109/IRI49571.2020.00009

common types of passwords and examines their strength. We introduce our approach in evaluating password pronounceability and apply the approach to the passwords in the dataset in Section V. Finally, we conclude with a discussion of our contributions and future work in Section VI.

II. Related Work

There is an excellent body of existing work in the field of passwords, and emerging within is the effort to solve the complexity and frustrations around passwords.

Several studies have examined how user-generated passwords are not secure [5], [12], and have related the lack of security to password creation policies [3]. The original purpose of having stringent policies, such as requiring digits, uppercase letters, and special characters in passwords, was to increase the password space. However, most people respond to such complex policies by taking their existing passwords and munging them to meet the minimum requirements [1], [13]. For example, "password" becomes "P@$$w0rd", or even "P@$$w0rdP@$$w0rd!" to meet all the requirements. These l33t (or leet speak) passwords are not as strong as people would think. Experts from both sides have quickly caught on with this trend, and have now devised systems that consider all possible changes to dictionary words to crack passwords or measure password strength [14], [15]. In our study, we examine the popularity of munged passwords and argue that password munging does not help improve password strength at all.

Other studies dwell upon the mathematical aspects of password security, such as password entropy, statistical analysis [16], and possibilities of attacks based on the password space [8], [17]. Entropy is a common measure of password strength [18], [19]. There are two widely used entropy measures, Shannon entropy [20] and the NIST entropy [21]. In our research, we are interested in determining the relative strength of each password in the dataset. Since Shannon entropy can specify the theoretical lower bound of the password space [22], we adopt the concept and analyze where the majority of the user-generated passwords fall on the entropy levels, and propose empirical observations on the results.

Even if we assume that password policies result in stronger passwords, they make those passwords difficult to remember or type. There are a few published studies that examine the ease and convenience of using and remembering strong passwords. Most related to our work is the research that combines the ease of memorability of passwords with its security [23]–[25]. Kelley et al. conducted a study where the users were shown passwords of varying complexity for a certain amount of time, after which they were to reproduce all the passwords they could remember [24]. Perhaps not surprisingly, the result were that users could easily remember words and sentences as opposed to random strings of characters. Gao et al. developed a model that uses login frequency and amount of practice to predict successful login duration and recall odds [23]. A major takeaway from their study is that remembering (or forgetting) passwords is related to not only password complexity but also the environment of use and how human memory functions. Shay et al. studied the usability of system-generated passwords and passphrases [25]. Contrary to common belief, system-generated passphrases did not outperform system-generated passwords of similar entropy. However, they found that pronounceable passwords are good candidates for creating system-generated passwords.

Our study focuses on the relationship between usability and security in user-generated passwords. More specifically, we aim to explore the pronounceability of user-generated passwords, since we value it as a significant factor of usability. Pronounceable passwords have quite a long history. Gasser et al. conducted an initial study of phoneme-based pronounceable passwords in the 1970s and showed that they were usable [26]. The scheme was later adopted in a NIST standard [27]. Ganesan et al. proposed an attack against the Gasser/NIST password scheme [28]. They revealed that it is vulnerable to the proposed attack, as the distribution of user-generated pronounceable passwords is highly non-uniform. In a relatively recent user study [29], the result showed that participants were able to recall long pronounceable passwords and short random passwords at the same rate. The authors thus argued that pronounceable passwords might be able to offer additional security without negatively affecting memorability. Instead of studying the nature of pronounceable passwords again, we try to convert an arbitrary password to its phoneme representation and generate a pronounceability score based on the representation. We then use the pronounceability scores of passwords along with their entropy to determine the correlation between usability and security of these passwords.

III. Dataset and Methodology

In this section, we first describe the methodology used to conduct the experiments. We then discuss the dataset and the characteristics of the passwords inside.

A. Methodology

We aim to conduct an empirical study of the password dataset in terms of security and usability. Before running the experiments, we need to pre-process the dataset by removing duplicates and non-Unicode passwords.

We then perform a statistical analysis of these passwords based on various categories of password composition, such as length, alphabets, and starting or ending character. In this step, one of the attributes we specifically focus on is the password strength, which we evaluate using password entropy. We also take a close look at some particular types of passwords, such as those made of dictionary words, those with repeated strings, and munged passwords.

In the next step, we look at the second attribute of passwords, usability. As usability is hard to quantify, we use the pronounceability of passwords to approximate it. The underlying assumption is that if a password is easy to pronounce, it will be easy to remember, which eventually can be considered an aspect of usability. We first introduce a phoneme-based scoring scheme to quantify the password pronounceability. We

then apply this scoring scheme to generate pronounceability scores for all the passwords in the dataset. At last, we try to correlate password entropy and pronounceability score to determine if user-generated passwords can be both secure and usable.

B. Dataset and Pre-processing

The dataset used was provided by Wang et al. [30], which includes 28,836,775 users and their passwords. Each user is identified by a unique user ID, which corresponds to a row in the file. One or more passwords are associated with each user ID. Table I shows two rows sampled from the dataset, containing two and three passwords, respectively.

TABLE I
SAMPLE ROWS FROM THE DATASET

id	Password	Password	Password
1	P@rk3rli1	parkerms1	
2	!!weed	Lovelady3	!!weed

Due to computational limits, we decide to extract about 7.6 million users from the original dataset, resulting in about 15 million passwords. Since our study focuses on password strength and usability rather than popularity, we clean the extracted data by removing all the duplicate passwords and only consider distinct passwords made of Unicode characters. This leads to around 7.75 million passwords, and we refer to this reduced dataset as 'main data.'

C. Characteristics of the Dataset

The first characteristic we look at is the strength of the passwords in the dataset. We adopt a variation of *Shannon entropy* for calculating password strength, which essentially estimates how unpredictable a password is [20]. The entropy $H = log_2 N^L$ is mainly determined by N, the number of character sets the password is based on, and L, the password length. One of Shannon entropy's counterparts is the NIST entropy, whose calculation is relatively easier [21]. However, it limits the impacts of N and L specified above. In this study, we aim to investigate the impacts of these two factors in user-generated passwords; thus, we choose the former option. Among the 7.75 million user-generated passwords, we find that the average entropy is 24.69 bits, which is by no means strong. We also examine the number of passwords that have Shannon entropy bits above certain thresholds, as shown in Figure 1. It turns out only 5.98% of the passwords have 40 or more bits of Shannon entropy. The remaining 94% can hardly be considered strong.

We now discuss other interesting characteristics of the dataset, such as password length, starting and ending characters, and character sets that the passwords are based on.

Shorter password length is considered less secure. NIST recommends that a user-generated password should be at least 8 characters or more in length [6]. As shown in Table II, about 30% of the passwords are made of 7 characters or less. This is probably a legacy issue since today most commercial websites

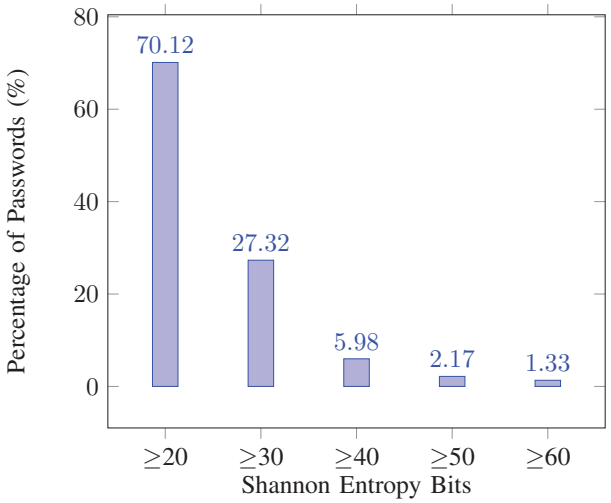

Fig. 1. Distribution of passwords with different Shannon entropy bits

and organizations require passwords of at least 8 characters long. We also notice that only 1.56% of the passwords are at least 16 characters long. Moreover, the average Shannon entropy of passwords with 16 or more characters is 71.01 bits, which suggests that they are secure. Several studies advocate passphrases a better choice of strong passwords [31]–[33]. The NIST password guideline also recommends password systems to allow at least 64 characters to support passphrases [6]. However, according to our findings, long passwords, such as passphrases, are still far from popular.

TABLE II
PASSWORDS ABOVE A CERTAIN LENGTH

Length	Number of Passwords	Percentage of Passwords	Avg. Shannon Entropy
$\geq$ 08	5,436,126	70.05	28.23
$\geq$ 10	2,252,772	29.03	35.83
$\geq$ 12	852,231	10.98	45.40
$\geq$ 14	442,660	5.70	52.48
$\geq$ 16	121,627	1.56	71.01

Passwords usually consist of lowercase letters, uppercase letters, digits, and special characters. We analyze the passwords with respect to their starting and ending characters, as shown in Table III. We notice that lowercase letters and digits are the users' favorite choice for starting or ending a password. Lowercase letters are used as the first character in 71.27% of the passwords. The average Shannon entropy of these passwords is 24.59 bits, lower than the average entropy of passwords starting with an uppercase letter (26.62 bits). We also find that 62.26% of the passwords end with a digit. We attribute this observation to password creation policies. Upon the suggestion of password creation policies, users may add a digit to the end of a password to simply get it accepted. Moreover, we notice that the average Shannon entropy of passwords ending with a digit is 24.99 bits, which is higher

than the average Shannon entropy of passwords ending with a lowercase letter (23.97 bits).

TABLE III
PASSWORDS STARTING AND ENDING CHARACTERISTICS

Category Type	Number of Passwords	Percentage of Passwords	Avg. Shannon Entropy
starts with a lowercase letter	5,531,043	71.27	24.59
starts with an uppercase letter	589,781	7.60	26.62
starts with a digit	1,601,896	20.64	24.17
starts with a special character	29,772	0.38	26.31
ends with a lowercase letter	2,625,504	33.83	23.97
ends with an uppercase letter	127,901	1.64	26.82
ends with a digit	4,831,556	62.26	24.99
ends with a special character	129,817	1.67	25.31

Table IV shows the characteristics of the passwords based on various components. We find that 56.69% of the passwords are created using just lowercase letters and digits. The Shannon entropy of this category of passwords is 25.17 bits, which is above the entire dataset's average entropy. The other two popular component categories are all lowercase (19.80%) and all digits (10.60%). It is not surprising to see that the all-digits passwords have an average Shannon entropy of 22.09 bits, which is among the lowest entropy levels. The title of Shannon entropy goes to the category with all four components (i.e., lowercase letters, uppercase letters, special symbols, and digits) enabled, which is 36.38 bits. However, this type of passwords only accounts for 0.57% of the whole population.

TABLE IV
PASSWORDS CHARACTERISTICS

Category Type	Number of Passwords	Percentage of Passwords	Avg. Shannon Entropy
all digits	822,588	10.60	22.09
all symbols	10,419	0.13	28.05
all lowercase	1,536,896	19.80	22.49
all uppercase	38,122	0.49	20.15
lowercase + digits	4,399,702	56.69	25.17
lowercase + symbols	108,083	1.39	25.70
lowercase + uppercase	102,326	1.31	24.12
uppercase + digits	93,936	1.21	24.03
uppercase + symbols	2,135	0.02	22.13
digits + symbols	12,678	0.16	22.64
lowercase + digits + symbols	116,400	1.50	34.06
uppercase + digits + symbols	3,809	0.04	30.07
lowercase + uppercase + digits	463,444	5.97	28.91
lowercase + uppercase + symbols	9,626	0.12	29.65
lowercase + uppercase + symbols + digits	44,808	0.57	36.38

IV. COMMON PASSWORD TYPES AT A GLANCE

In this section, we take a close look at some common types of passwords as we are particularly interested in their strength.

A. Dictionary Words as Passwords

Picking dictionary words as a password is a poor practice of security. However, users tend to create a password that is easy to remember. This usually leads to the heavy use of dictionary words. We want to figure out how many such vulnerable passwords exist in the dataset.

Our first step is to extract the passwords made up of the English alphabet. Out of 7.75 million passwords, around 1.57 million (20.23%) passwords are entirely made up of lowercase and uppercase letters. We also find 48,528 (0.62%) passwords that exactly match the words from an English dictionary. We refer to these 48,528 passwords as pure dictionary word passwords.

Next, we perform an experiment to analyze how many passwords contain at least one or more dictionary words. We remove the pure dictionary word passwords from the password set that is only made of lowercase and upper case letters. This step results in a total of 866,638 passwords. We find that 7.52% (65,209) of the passwords contain at least one or more dictionary words plus some additional characters.

B. Finding Repeating Patterns in Passwords

A password is generally considered more secure as the length increases. NIST recommends increasing the maximum length to 64 characters [6]. However, if a password has a repeating pattern in it, the length alone will give a false sense of security. An example of passwords with repeating patterns is "PASSWORDPASSWORD." The substring "PASSWORD" appears twice. In our experiment, we look up repeating patterns of three or more characters in a password. We notice that 108,919 passwords contain repeating patterns. It implies that users tend to increase the length of passwords to meet the password creation policies by repeating certain strings.

We define *maximum Shannon entropy* as a criterion to indicate the strength loss of passwords with repeating patterns. The Shannon entropy of a password depends on the length of the password and the character sets used in the password creation. Generally, the longer a password is, the higher the entropy is. However, the repetition of characters decreases the entropy bits. Thus, a password with repeating strings is not able to reach its maximum Shannon entropy. For example, let's assume a random password "odhrkna," which is 7 characters long. This password has 21 bits of Shannon entropy. The next password we choose is "ketiket," which has "ket" being repeated. It also has a length of 7 characters. However, "ketiket" has only 14 bits of Shannon entropy. Although both passwords have the same length and are made from the same alphabet, the password with a repeating pattern results in a lower Shannon entropy (reduced from 21 bits to 14 bits). It is clearly not a good idea to use repeating patterns to create a password. We sample 15 passwords with repeating patterns from the dataset and show both the actual Shannon entropy

and the maximum Shannon entropy for each password in Table V.

TABLE V
REPEATING PASSWORD, ENTROPY AND ENTROPY OF PASSWORD OF SAME LENGTH WITH UNIQUE ALPHABETS

Repeating Password	Shannon Entropy	Maximum Shannon Entropy
ketiket	14	21
arunaru	14	21
savasava	16	24
112233112	18	27
3uben3uben	20	30
xxx1988xxx	20	30
newbynewbs	30	30
12341234go	30	30
topic1topic1	36	48
19821985stas	36	48
13542201354220	42	56
cmcc123987cmcc	42	56
facebookfacebook	48	64
2804198728041987	48	64
principleprinciple	54	72

We also calculate the average Shannon entropy of all the passwords with repeating patterns, which is 25.86 bits. The irony is that this is more than 1 bit higher than the average entropy of the entire 7.75 million passwords, which is 24.69 bits. We also notice that 56% of the passwords with repeating patterns are at least 10 characters long, whereas only 29% of the passwords from our main dataset are greater than or equal to 10 characters. We conclude that even though the passwords with repeating patterns cannot achieve the maximum entropy level in theory, they tend to have a greater length, resulting in a slightly higher entropy than normal passwords.

C. Munged Passwords

A good password is easy to remember and hard to guess. Users often try to find a way to create an easy-to-remember password that also fulfills the requirements of password creation policies. This often leads to the use of munged passwords. The term *munge* stands for 'Modify Until Not Guessed Easily'. It refers to the practice of creating a password with common replacement strategies. For example, replacing 'a' with '@' in a password. Since users are required to include special characters, they may decide to replace all the 'a' with '@' or use '3' instead of 'E.' This type of passwords are also called *leet speak passwords*.

In our study, we create a munged password conversion tool. We define the term *normal representation*, meaning the original alphabet. For example, 'S' can be replaced with '5' when converting into a munged representation. Here, 'S' is the normal representation, whereas '5' is the munged representation. We use two methods to look for munged passwords.

In the first method, we try to locate the dictionary words used in the munging process. We convert every possible munged representation of a password into its normal representation. We then check if the newly created string matches a dictionary word. For example, the munged password "5@!l0r" can be converted into a normal representation "sailor," which is a dictionary word.

In the second method, we attempt to identify one munged character at a time, convert it into its normal representation, and check if the resulting string matches a dictionary word. We keep moving forward in the password string and repeating the same process until we hit the end. The motivation behind the second approach is that many users prefer a partial munging instead of a full one. For example, let us consider a password "pa1nting," which is derived from "painting" by substituting 'i' with '1.' In this example, only one of the 'i' is converted to '1,' although there are two 'i' in the string. Moreover, the user did not convert 'a' to '@,' which is another common strategy. Therefore, we have to try out all possible scenarios by converting every possible munged character to see if it is a dictionary word. Given the explosive workload, we decide to run the test on a smaller dataset of 268,678 passwords. We find a total of 300 passwords that are exactly munged forms of dictionary words. The number of passwords that contain munged dictionary words inside is, of course, way more than that. Some examples of munged passwords found in the dataset are shown in Table VI.

TABLE VI
PASSWORDS AND THEIR MUNGED PASSWORD TRANSFORMATION

Password	Munged Password
tiger5	tigers
v3ritas	veritas
topper5	toppers
papercl1p	paperclip
pa1nting	painting

Munging a password does not make the password more secure, as it uses the same character sets and does not increase the password length. Munged passwords are still vulnerable to brute force guessing attacks, especially when a user strictly follows the munging rules and munges every possible character. Let's consider the password 'Medieval' as an example, which has 24 bits of Shannon entropy. If a user decides to munge the password by replacing all e's with 3's, the resulting password, 'M3di3val,' still has 24 bits of Shannon entropy. Doing partial munging substitution may create a slightly stronger password than a full substitution though, as cracking the former requires to check more combinations than cracking the latter in a brute force guessing attack (e.g., {M3di3val, Mediev@l, Med!eval} vs. M3d!3v@l). Overall, we do not recommend users to use munged passwords.

V. PASSWORD PRONOUNCEABILITY

In this section, we investigate the usability of passwords. We develop a scheme to calculate how pronounceable a password is and use it to estimate the password usability.

A. Password-to-Phoneme Conversion

The concept of pronounceable passwords dates back to several decades ago. Gasser et al. suggested randomly generated, pronounceable passwords a viable approach for password

generators [26]. A *phoneme*, which is a unit of sound, is an essential part that constructs pronounceable passwords. In Gasser et al.'s method, 34 phonemes are used as input to the password generator.

Instead of using phonemes to generate passwords, we use phonemes to construct an intermediate representation of passwords before examining their pronounceability. The set of phonemes are taken from the CMU Pronouncing Dictionary [34]. They have identified 39 phonemes for the English language. Advanced Research Projects Agency (ARPA), developed a set of phonetic transcription codes - ARPAbet [35], which we use to represent the phoneme set in this study.

The first step is to convert passwords to phonemes. We use Python's `g2p_en` library to create the phoneme representation of a string [36]. This library adopts the set of phonemes from the CMU Pronouncing Dictionary and the underlying phonetic transcription codes from ARPAbet. `g2p_en` takes an input string and converts each character of the string into its phonetic transcription. The output is an array of phonetic transcriptions. We join all the transcriptions into a single string and call it the phoneme representation of the input string.

Consider the password "medieval" as an example. We pass the string to `g2p_en`, resulting in an array, ['M', 'IH0', 'D', 'IY1', 'V', 'AH0', 'L']. We then join all the elements of the array into a single string "MIH0DIY1VAH0L," which is the phoneme representation of the password "medieval."

B. Phoneme Matching

We then map the phoneme representation of a password against the set of all possible phoneme combinations in the English language to estimate the ease of memorability, as shown in Fig. 2.

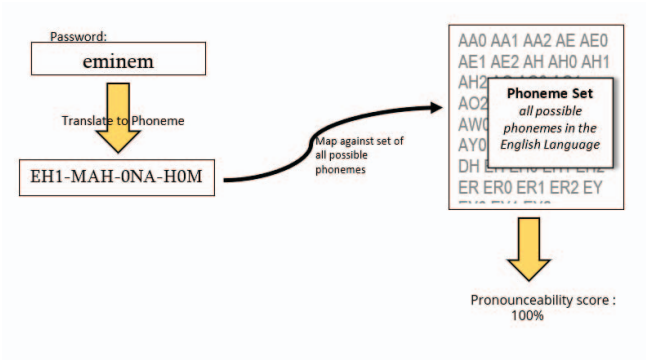

Fig. 2. An example of phoneme translation

We use Python's `fuzzywuzzy` library to implement the comparison [37]. `fuzzywuzzy` is a string-matching algorithm that uses fuzzy string-matching and Levenshtein Distance [38] to calculate the differences between two strings. Fuzzy string-matching is a process of finding strings that match a given pattern approximately, and Levenshtein Distance is a method used to measure the differences between two strings.

The passwords in the dataset are not likely to precisely match the existing phonemes. The fuzzy string-matching technique is not only able to detect if two strings match, but also calculate how similar two strings are. For example, the word "medieval" can be represented phonematically as "m-ih0-d-iy1-v-ah0-l." When choosing passwords, users can choose to spell it in many different ways though, such as "medevel," "maideval," "midivl," etc. Each of these will be approximately matched to some existing phoneme combinations and scored based on how closely they map to the existing phonemes. Additionally, the algorithm computes how similar or different it is from the existing phonemes and assigns a score. In the context of our project, this logic is applied to user-generated passwords. Each password is fuzzy string-matched with the existing phonemes in the English language to predict how pronounceable a password is.

Expanding on the example mentioned, Table VII displays the phonemes and the estimated pronounceability for various versions of the same word. Here we have considered "medieval" to be the original word, and the words following it to be its versions. Though not real words, the phonemes of words 2, 3, and 4 are pronounceable in English. The original word serves as a demonstration example, but in reality, the phonemes are not mapped against a particular word, but all possible combinations of phonemes. Similarly, all the passwords in the dataset are matched against this set of phonemes.

TABLE VII
FUZZY STRING-MATCHING EXAMPLE.

No.	Word	Phoneme	Pronounceability Score
1	Medieval	MIH0DIY1VAH0L	100
2	Medevel	MEH1DIH0VAH0L	92
3	Maideval	MEY1DAH0VAH0L	90
4	Midivl	MIH1DIH0VAH0L	89

After evaluating how similar two strings are, `fuzzywuzzy` returns a similarity index out of 100. We define the similarity index as a pronounceability score for a password. A high pronounceability score indicates that the word is highly pronounceable, while a low score implies the opposite. At the end of this implementation, each password in the dataset gets assigned a pronounceability score. This pronounceability score estimates how pronounceable a password is, and hence how usable a password is. Using these two metrics, password entropy and password pronounceability, we map the security and usability of passwords together.

C. Pronounceability of Passwords in the Dataset

For this experiment, we choose to use the smaller dataset, which contains pure dictionary words as passwords, as mentioned in section IV-A. We know that dictionary words are a terrible choice for passwords. However, in this experiment, we focus on the usability aspect, and thus passwords that contain highly pronounceable dictionary words are of particular interest.

TABLE VIII
STRENGTH AND USABILITY OF PHONEMES

Passwords	Shannon Entropy	Phoneme Representation	Pronounceability Score
antidisestablishmentarianism	84	AE2NTAY0DIH0SAH0STAE2BLIH0SHMAH0NTEH1RIY0AH0NIH0ZAH0M	96
supercalifragilistic	60	SUW2PER0KAE2LAH0FRAE1JHAH0LIH2STIH0K	100
telecomunicaciones	54	TEH2LAH0KAH0MYUW2NAH0SEY1SHAH0NZ	97
telecommunication	51	TEH2LAH0KAH0MYUW2NIH0KEY1SHAH0N	100
electrocardiogram	51	IH0LEH2KTROW0KAA1RDIY0AH0GRAE2M	97
telecomunicazion	48	TEH2LAH0KAH0MYUW2NAH0SEY1SHAH0N	95
electromagnetism	48	IH0LEH2KTROW0MAE1GNAH0TIH2ZAH0M	97
telecomunicacion	48	TEH2LAH0KAH0MYUW2NAH0KEY1SHAH0N	98
prestidigitation	48	PREH2STIH0DIH0JHAH0TEY1SHAH0N	100
revolucionario	42	REH2VOW0LUW2SIY0AH0NEH1RIY0OW0	100

We calculate the pronounceability score for the phoneme representation of all the passwords in the dataset. Table VIII displays the top 10 passwords based on their Shannon entropy. Most of them appear to be either English compound words or variations of compound words in other languages. As we can see, these compound words-based passwords have long entropy bits and high pronounceability scores ($\geq$ 95%) at the same time. Similar to compound words, passphrases also consist of multiple dictionary words, and they are pronounceable. This implies that passphrases can be good candidates for strong and pronounceable passwords as well.

Figure 3 plots the pronounceability score vs. the Shannon entropy of the passwords in the chosen set. We can see that the entropy bits of most passwords lie between 30 to 40 bits, and their pronounceability scores lie between 80% to 100%. It seems that most passwords from this smaller set have a decent pronounceability. On the other hand, their entropy scores generally fall in the range of weak to reasonable, according to the scoring table in [39]. We realize that there is no consensus on how many Shannon entropy bits can be considered strong, medium, or weak. The same applies to our proposed pronounceability scoring. However, this figure can give us a brief idea about how user-generated passwords in the dataset align with the two crucial criteria of passwords.

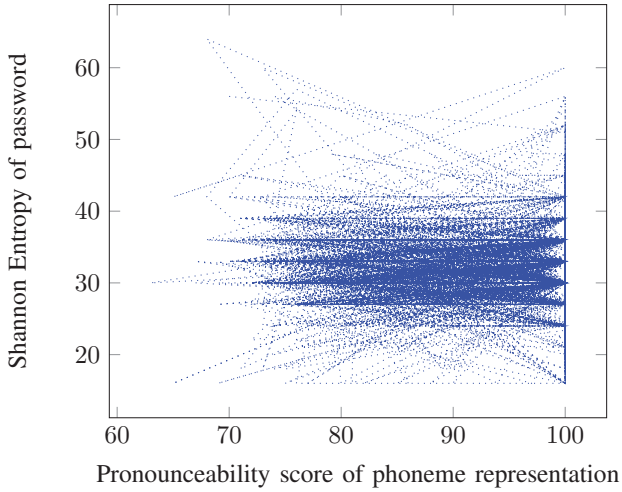

Fig. 3. Pronounceability score vs. Shannon entropy

VI. CONCLUSION AND FUTURE WORK

In this paper, we empirically examined the characteristics of passwords in a huge dataset. The experiments focused on two aspects, security and usability of passwords. We used Shannon entropy to evaluate the password strength, while we proposed a pronounceability scoring scheme to estimate the usability of a password. We applied these two measures to the passwords in the dataset and found several interesting insights. Based on our findings, we suggested that using passphrases as passwords is a promising way of achieving both security and usability.

In the future, we plan to further investigate the two measures, and come up with meaningful scoring criteria for both Shannon entropy and pronounceability score (e.g., How many bits are considered strong, medium, or weak? How much score is considered pronounceable?). We are also interested in investigating passphrase datasets and conducting user surveys to gain a more in-depth understanding of strong and usable passwords.

REFERENCES

[1] E. Stobert and R. Biddle, "The password life cycle: user behaviour in managing passwords," in *10th Symposium On Usable Privacy and Security (SOUPS 2014)*, 2014, pp. 243–255.

[2] D. Florencio and C. Herley, "A large-scale study of web password habits," in *Proceedings of the 16th International Conference on World Wide Web*. ACM, 2007, pp. 657–666.

[3] E. Von Zezschwitz, A. De Luca, and H. Hussmann, "Survival of the shortest: A retrospective analysis of influencing factors on password composition," in *IFIP Conference on Human-Computer Interaction*. Springer, 2013, pp. 460–467.

[4] R. Wash, E. Rader, R. Berman, and Z. Wellmer, "Understanding password choices: How frequently entered passwords are re-used across websites," in *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*, 2016, pp. 175–188.

[5] B. Ur, F. Noma, J. Bees, S. M. Segreti, R. Shay, L. Bauer, N. Christin, and L. F. Cranor, ""I Added '!' at the End to Make It Secure": Observing Password Creation in the Lab," in *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*, 2015, pp. 123–140.

[6] P. Grassi, M. Garcia, and J. Fenton, "Digital identity guidelines," National Institute of Standards and Technology, Tech. Rep., 2017.

[7] D. Malone and K. Maher, "Investigating the distribution of password choices," in *Proceedings of the 21st International Conference on World Wide Web*. ACM, 2012, pp. 301–310.

[8] R. V. Yampolskiy, "Analyzing user password selection behavior for reduction of password space," in *Proceedings of the 40th Annual 2006 International Carnahan Conference on Security Technology*. IEEE, 2006, pp. 109–115.

[9] R. W. Proctor, M.-C. Lien, K.-P. L. Vu, E. E. Schultz, and G. Salvendy, "Improving computer security for authentication of users: Influence of proactive password restrictions," *Behavior Research Methods, Instruments, & Computers*, vol. 34, no. 2, pp. 163–169, 2002.

[10] A. Rao, B. Jha, and G. Kini, "Effect of grammar on security of long passwords," in *Proceedings of the Third ACM Conference on Data and Application Security and Privacy*. ACM, 2013, pp. 317–324.

[11] A. Conklin, G. Dietrich, and D. Walz, "Password-based authentication: a system perspective," in *Proceedings of the 37th Annual Hawaii International Conference on System Sciences, 2004.* IEEE, 2004, pp. 10–pp.

[12] C. Kuo, S. Romanosky, and L. F. Cranor, "Human selection of mnemonic phrase-based passwords," in *Proceedings of the Second Symposium on Usable Privacy and Security*, 2006, pp. 67–78.

[13] M. Jakobsson and M. Dhiman, "The benefits of understanding passwords," in *Mobile Authentication*. Springer, 2013, pp. 5–24.

[14] P. Cao, H. Li, K. Nahrstedt, Z. Kalbarczyk, R. Iyer, and A. J. Slagell, "Personalized password guessing: a new security threat," in *Proceedings of the 2014 Symposium and Bootcamp on the Science of Security*, 2014, pp. 1–2.

[15] D. L. Wheeler, "zxcvbn: Low-budget password strength estimation," in *25th USENIX Security Symposium (USENIX Security 16)*, 2016, pp. 157–173.

[16] J. Bonneau, "The science of guessing: analyzing an anonymized corpus of 70 million passwords," in *2012 IEEE Symposium on Security and Privacy*. IEEE, 2012, pp. 538–552.

[17] Z. Li, W. Han, and W. Xu, "A large-scale empirical analysis of chinese web passwords," in *23rd USENIX Security Symposium (USENIX Security 14)*, 2014, pp. 559–574.

[18] M. Weir, S. Aggarwal, M. Collins, and H. Stern, "Testing metrics for password creation policies by attacking large sets of revealed passwords," in *Proceedings of the 17th ACM Conference on Computer and Communications Security*. ACM, 2010, pp. 162–175.

[19] R. Shay, S. Komanduri, P. G. Kelley, P. G. Leon, M. L. Mazurek, L. Bauer, N. Christin, and L. F. Cranor, "Encountering stronger password requirements: user attitudes and behaviors," in *Proceedings of the Sixth Symposium on Usable Privacy and Security*. ACM, 2010, pp. 1–20.

[20] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[21] W. Burr, D. Dodson, R. Perlner, S. Gupta, and E. Nabbus, "Nist sp800-63-2: Electronic authentication guideline," National Institute of Standards and Technology, Reston, VA, Tech. Rep., 2013.

[22] J. L. Massey, "Guessing and entropy," in *Proceedings of 1994 IEEE International Symposium on Information Theory*. IEEE, 1994, p. 204.

[23] X. Gao, Y. Yang, C. Liu, C. Mitropoulos, J. Lindqvist, and A. Oulasvirta, "Forgetting of passwords: ecological theory and data," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 221–238.

[24] P. G. Kelley, S. Komanduri, M. L. Mazurek, R. Shay, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, and J. Lopez, "Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms," in *2012 IEEE Symposium on Security and Privacy*. IEEE, 2012, pp. 523–537.

[25] R. Shay, P. G. Kelley, S. Komanduri, M. L. Mazurek, B. Ur, T. Vidas, L. Bauer, N. Christin, and L. F. Cranor, "Correct horse battery staple: Exploring the usability of system-assigned passphrases," in *Proceedings of the Eighth Symposium on Usable Privacy and Security*. ACM, 2012, pp. 1–20.

[26] M. Gasser, "A random word generator for pronounceable passwords," MITRE CORP, Bedford, MA, Tech. Rep., 1975.

[27] F. PUB, "Automated password generator (apg)," 1993.

[28] R. Ganesan, C. Davies, and B. Atlantic, "A new attack on random pronounceable password generators," in *17th NIST-NCSC National Computer Security Conference*. Citeseer, 1994, pp. 184–197.

[29] S.-h. Lau, S. Siena, A. Pandey, S. Sosothikul, L. Cranor, B. Ur, and R. Shay, "Exploring the usability of pronounceable passwords," *SOUPS Poster*, 2014.

[30] C. Wang, S. T. Jan, H. Hu, D. Bossart, and G. Wang, "The next domino to fall: Empirical analysis of user passwords across online services," in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*. ACM, 2018, pp. 196–203.

[31] S. A. Kurzban, "Easily remembered passphrases: a better approach," *ACM SIGSAC Review*, vol. 3, no. 2-4, pp. 10–21, 1985.

[32] M. Keith, B. Shao, and P. J. Steinbart, "The usability of passphrases for authentication: An empirical field study," *International Journal of Human-Computer Studies*, vol. 65, no. 1, pp. 17–28, 2007.

[33] R. Munroe, "xkcd: Password strength," https://www.xkcd.com/936/, 2012.

[34] "The CMU pronouncing dictionary," www.speech.cs.cmu.edu/cgi-bin/cmudict, Last accessed 19 April 2019.

[35] A. Klautau, "Arpabet and the timit alphabet," 2001.

[36] "g2p-en," https://pypi.org/project/g2p-en/, Last accessed 2 May 2020.

[37] "Fuzzywuzzy," github.com/seatgeek/fuzzywuzzy, Last accessed 19 April 2019.

[38] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet Physics Doklady*, vol. 10, no. 8, 1966, pp. 707–710.

[39] "Calculating password entropy," https://www.pleacher.com/mp/mlessons/algebra/entropy.html, Last accessed 14 May 2020.