IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 29 April 2024, accepted 10 May 2024, date of publication 15 May 2024, date of current version 22 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3401195

RESEARCH ARTICLE

A Study on Markov-Based Password Strength Meters

BINH LE THANH THAI AND HIDEMA TANAKA
Department of Computer Science, National Defense Academy of Japan, Yokosuka, Kanagawa 239-8686, Japan
Corresponding author: Binh Le Thanh Thai (binhbe603501@gmail.com)

ABSTRACT Nowadays, passwords play an important role in ensuring practical security. Password strength meters (PSMs) are typical and well-known tools developed to help users estimate password strength. Among existing PSMs, Markov-based PSMs offer high accuracy and reliability. However, these PSMs are often compared with other types of PSMs, and no study has compared their accuracy in detail. Two types of Markov models introduced in previous studies are the Simple Markov Model (SMM) and the Layered Markov Model (LMM). When calculating the probability of a password, these models consider two factors: character position and password length. However, these two models do not cover all possible cases of these two factors. In this paper, we introduce the application of two new Markov models, which can be seen as "hybrid models" of SMM and LMM, into PSMs, called *Simple Markov Model with password length consideration* (SMMl) and *unique Layered Markov Model* (uLMM), to cover all the missing cases. Then, we present two specific scenarios and compare the effectiveness of four types of Markov models in evaluating passwords based on these two scenarios. The experimental results indicate that the SMMl model, one of the two models proposed in this paper, yields the best effectiveness. This result also suggests that the number of samples of sub-string sequences obtained during the training process affects the effectiveness of password evaluation. Therefore, we conduct a detailed analysis related to this issue. These results will support us in developing new PSMs in the future.

INDEX TERMS Markov model, password strength, privacy, security.

I. INTRODUCTION

A. BACKGROUND

In the field of information security, authentication is one of the critical factors that determines system security. Authentication methods have significantly diversified. However, password-based authentication remains widely used due to its simple implementation and user-friendliness. With the remarkable advancement of technology and the Internet, nearly all services necessitate password usage, including logging into computers or creating accounts for online shopping. However, previous studies [1], [2], [3] have shown that this authentication method has many vulnerabilities, and recently, there have been numerous cases of massive password leaks [4], [5], [6]. Therefore, issues related to passwords need to be given more attention to ensure the maintenance of system security and the protection of user privacy.

The main topic surrounding passwords is their strength or robustness against password-cracking methods. A trade-off relationship always exists between convenience and security when users create passwords [7]. Simple passwords such as combinations of names and birth years of users, e.g., `shane1983`, are easy to remember and therefore convenient, but extremely predictable for hackers. Conversely, long, randomly generated complex passwords, e.g., `F@!sdkg6`, offer higher security but are very inconvenient. Unfortunately, recent studies [8], [9] have shown that despite being aware of the significant threats posed by easily remembered passwords to both system security and personal privacy, users still prioritize using them. Furthermore, they often reuse the same passwords across multiple websites [10], [11].

The associate editor coordinating the review of this manuscript and approving it for publication was Pedro R. M. Inácio.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

To assist users in estimating password strength, password strength meters (PSMs) are commonly used and well-known tools [12], [13], [14]. Many PSMs have been proposed, and they often rely on rules such as the LUDS requirement. This requirement suggests that passwords should consist of characters from **L**owercase (**L**), **U**ppercase (**U**), **D**igits (**D**), and **S**ymbols (**S**). In this paper, we refer to the set **C** ($C = L \cup U \cup D \cup S$) as the set of all 94 printable ASCII characters. Another approach used in rule-based PSMs is to analyze patterns in passwords, which assesses the commonality of a password by considering various sources such as census data for common names and general words from Wikipedia.

However, these rules are heuristic and primarily designed to counter outdated attack methods, such as brute-force attacks [15] or attacks based on lists of common passwords [16]. Consequently, the accuracy of such PSMs is often low, leading users to choose passwords with high scores but low actual security. Conversely, many modern and effective password-cracking methods have been developed. Specifically, probabilistic methods using Markov models [17], [18], [19] have shown very good effectiveness. In general, these methods attempt to simulate the passwords that users may create in probability from high to low and then try these passwords one by one. They have a notable strength in being able to crack even passwords not in the training dataset. Fortunately, this result suggests that using Markov models to estimate the strength of passwords is both reasonable and effective as well.

B. MOTIVATIONS AND CONTRIBUTIONS

In fact, many Markov-based PSMs have been proposed, and their effectiveness has been confirmed [20], [21], [22]. Markov-based PSMs primarily focus on measuring the strength of passwords against dictionary attacks and probabilistic attacks. Similar to the case of probabilistic attacks, the main advantage of Markov-based PSMs over others is that they can accurately evaluate passwords not appearing in the training dataset.

However, except for [22], other papers have been reported quite a long time ago. Since then, numerous breaches [4], [5], [6] have occurred, resulting in an incredibly large number of leaked passwords. Additionally, with the advancement of computers and programming languages, developing password-cracking methods becomes more practical and easier, especially password-cracking methods using leaked passwords. Furthermore, previous studies only compare Markov-based PSMs with other types of PSMs without comparing among Markov-based PSMs in detail. Therefore, comparing the effectiveness of these PSMs when using up-to-date datasets and developing new PSMs with higher accuracy is very valuable. This is the motivation behind our research.

Two types of Markov models have been used in previous studies: the *Simple Markov Model* (referred to as SMM, used in [21], [22], and [20]) and the *Layered Markov Model* (referred to as LMM, introduced in [21]). These models differ in how they consider two factors: character position and password length when calculating the probability of passwords. While SMM does not consider both of these factors, LMM, on the contrary, considers both. More specifically, SMM uses a simple probability model (a probability model that does not consider character position) for all possible password lengths. This makes SMM both simple and ensures a huge amount of samples for each sub-string. In contrast, LMM uses a layered probability model (a probability model that considers character position) for each possible password length. This approach, although providing detailed characteristics of character usage, significantly reduces the number of samples obtained for each sub-string. So why not try using ''hybrid models'' of these two Markov models?

TABLE 1. Summary of the differences among Markov models.

Factor	Markov model			
	SMM	SMMl	uLMM	LMM
Character position	No	No	Yes	Yes
Password length	No	Yes	No	Yes
Used in	[20]–[22]	This paper	This paper	[21]

The ''hybrid models'' we refer to here are Markov models that only consider either character position or password length. We call them, respectively, the *unique Layered Markov Model* (uLMM) and the *Simple Markov Model with password length consideration* (SMMl). More specifically, uLMM utilizes a unique layered probability model for all possible password lengths, whereas SMMl employs a simple probability model for each possible password length. Clearly, using these two models will mitigate the imbalance between i) overly focusing on simplicity and the number of samples of sub-strings as in SMM and ii) overly focusing on detail but lacking the necessary samples of sub-strings as in LMM. In this paper, we apply the idea of using these two Markov models to evaluate passwords. Table 1 provides an overview of the differences between the four Markov models used in this paper, and their details are described in Section III.

Another noteworthy point is the training dataset. Reference [23] has shown that a password, even if evaluated as strong by a certain PSM, is not actually strong if it has been leaked before. Taking this into account, Markov-based PSMs are usually trained with a large dataset built from leaked passwords to ensure the creation of a probability model accurate enough to evaluate passwords. Reference [20] used a dataset containing 32.6 million leaked passwords and showed that this size of the training dataset provides a good evaluation. In [21], a dataset of up to 75 million passwords was used; however, this dataset only contains unique passwords. Since weak and simple passwords are usually reused and therefore will appear more times in real-world leaked password lists, we contend that utilizing such a dataset

may impact the accuracy of the evaluation results. Therefore, in this paper, we utilize a training dataset with 80 million real-world leaked passwords, with duplicates retained to preserve the distribution of user-selected characters.

We all know that ''evaluating strong passwords as weak'' does not increase security risk but ''evaluating weak passwords as strong'' does. Based on this reasoning, we focus on accuracy in detecting weak passwords to determine two specific scenarios for comparing the effectiveness of PSMs using the four types of Markov models. Since [20] has demonstrated through analysis that password distribution varies among different websites, we consider two scenarios A and B as follows.

Scenario A occurs when a hacker obtains a portion of passwords from a service and attempts to crack the passwords of other users within the same service. On the other hand, **Scenario B** arises when a hacker acquires passwords from one service and endeavors to crack the passwords of users in another service. The experimental results indicate that SMMl, one of the two models introduced in this paper, achieves the highest effectiveness in both scenarios. These results imply the importance of the number of types of sub-string sequences obtained as well as the frequency of their usage in password evaluation. Based on these results, we also conducted an analysis related to this issue.

In summary, the contributions of this paper are as follows:

- We propose the application of two new Markov models, SMMl and uLMM, in password evaluation. These models are considered hybrids of the Markov models used in previous studies.
- This paper, for the first time, utilizes a dataset containing the latest real-world leaked passwords to compare the effectiveness of Markov-based approaches under two specific scenarios.
- The experimental results demonstrate that SMMl, one of the two models we introduce, achieves the highest effectiveness in mitigating security risks from weak passwords in both scenarios.
- We conducted an analysis related to the number of types of character sequences obtained as well as the frequency of their usage. This sheds light on new directions for developing PSMs in the future.

The rest of this paper is organized as follows: Section II presents a summary of related works, including the evolution of Markov-based PSMs and other trends in password strength research. In Section III, we explain the Markov models utilized in this study, including Simple Markov Model, Layered Markov Model, unique Layered Markov Model, and Simple Markov Model with password length consideration. Next, Sections IV and V are dedicated to presenting the experiments based on scenarios A and B, respectively. Then, in Section VI, we describe the analysis related to the number of types of sub-string sequences obtained and the frequency of their usage. Section VII discusses the findings and outlines future work directions. Finally, Section VIII concludes the paper.

II. SUMMARY OF RELATED WORKS

In this section, we present a summary of related works, including the evolution of Markov-based PSMs and other trends in password strength research.

A. EVOLUTION OF MARKOV-BASED PASSWORD STRENGTH METERS

The idea of applying Markov models to password evaluation was first utilized by Castelluccia et al. [20] in 2012. The paper [20] focused on the SMM and aimed to develop an adaptive meter capable of dynamically responding to changes in how users choose passwords on different websites. However, this approach did not emphasize password categorization criteria.

Two years later, in 2014, Galbally et al. [21] introduced the LMM for password evaluation and compared its performance with the use of SMM. Similar to [20], and [21] did not provide specific criteria for password categorization. Additionally, the dataset used in their study contained only unique passwords, whereas Markov-based password-cracking methods are typically trained on real-world leaked datasets, which include many reused passwords.

Most recently, in 2024, Thai and Tanaka [22] proposed a new PSM that applies statistical criteria based on the 3σ rule [24] to address the lack of specific criteria for password categorization. This PSM uses a huge dataset containing passwords collected from Collection #1 [4], which is the largest and most recent security breach, allowing for a more accurate analysis of user character usage.

B. OTHER TRENDS IN PASSWORD STRENGTH RESEARCH

In addition to the Markov-based PSMs, there are many PSMs as well as other research trends in password strength. In this section, we introduce and summarize some other PSMs and trends in their development.

One of the most well-known PSMs is the NIST entropy. The NIST entropy, defined by the National Institute of Standards and Technology in NIST Special Publication 800-63B [25], is a method that quantifies the randomness and complexity of a password based on entropy. This approach considers factors such as password length, character diversity, and the use of various character types, including uppercase letters, lowercase letters, digits, and special characters.

Some PSMs take additional factors into account. Hu [26] proposes a method that incorporates both the LUDS requirements and the patterns in passwords. Galbally et al. [27] introduce a multimodal strength metric that combines multiple individual metrics to leverage their strengths and mitigate their weaknesses. Dong et al. [28] propose a meter based on user behavior, specifically focusing on password reuse, Leet transformation (substituting letters with numbers or symbols), and separation.

Besides, many analyses on password strength have also been conducted. Bailey et al. [29] present an analysis of both password reuse and the correlation between the value of an

account and the strength of the passwords associated with those accounts. Dupuis and Khan [30] investigate the effect of social influence, which refers to the influence others have on attitudes and behaviors, on password strength. Golla and Dürmuth [31] provide an overview of the accuracy of PSMs and propose a set of properties that a PSM must fulfill to be considered highly accurate.

Recently, there has been a growing trend in applying machine learning to the development of PSMs. Melicher et al. [32] develop a PSM using artificial neural networks to model the resistance of passwords to password cracking. They demonstrate how different architectures and training methods impact the guessing effectiveness of neural networks. Pasquini et al. [33] present theoretical foundations for interpretable probabilistic PSMs and describe their implementation using a lightweight deep learning framework. Kim and Lee [34] propose a PSM that employs deep-learning based multi-class classification. They specifically address the issue of leaked frequency not being considered during the evaluation process.

III. MARKOV-BASED PASSWORD STRENGTH METERS

In this section, we describe the details of the four Markov models used in this paper, including two models used in previous studies and two new models proposed in this paper.

A. PASSWORD STRENGTH ESTIMATION

Markov models are sequences of states in which the probability depends only on the state immediately preceding it. The fundamental idea is that user-created passwords often exhibit certain patterns and regularities between adjacent letters, rather than being chosen independently. For example, in the passwords of English-speaking users, the 2-gram `th` is much more likely than `tz`, and the letter `e` is frequently found following `th`.

Based on this idea, Markov-based PSMs are trained on a large dataset, usually leaked passwords, to create a large probabilistic model and use this model to evaluate the probability that a user will generate a certain password. Let s and $I(s)$ denote a password and its strength, respectively. In general, $I(s)$ is calculated as follows:

$$I(s) = -log_2\Big(P(s)\Big) \tag{1}$$

where $P(s)$ denotes the probability of s.

In general, an n-gram Markov model generally considers its probability using the previous $n-1$ characters. However, different Markov models have different ways of calculating probabilities. In this paper, prioritizing the comparison of algorithm effectiveness as well as facilitating easy computation, we describe the simplest case: the 2-gram Markov models.

B. SIMPLE MARKOV MODEL

The SMM simply focuses on the transition probability between characters in passwords without considering either character positions or password length. That is, it employs a unique simple probability model for all possible password lengths.

For a given m-character password $s = s_1, s_2, \ldots, s_m$ (each character s_i is selected from character space $\mathbf{C}$), we can express its probability as follows:

$$P(s) = P(s_1)P(s_2|s_1)P(s_3|s_2)\ldots P(s_m|s_{m-1}) \tag{2}$$

where $P(\cdot)$ and $P(\cdot|\cdot)$ denote the probability and the conditional probability, respectively, which are calculated as follows:

$$P(s_1) = \frac{count(s_1)}{\sum_{x\in C} count(x)}$$
$$P(s_i|s_{i-1}) = \frac{count(s_{i-1}, s_i)}{count(s_{i-1})},$$

where $count(s_{i-1}, s_i)$ indicates the track of the 2-gram counts.

For example, the probability of the password `dog!` is: the probability of `d` being at the beginning, multiplied by the probability that `o` follows `d`, multiplied by the probability that `g` follows `o`, and multiplied by the probability that `!` follows `g`.

$$P(\texttt{dog!}) = P(\texttt{d})P(\texttt{o}|\texttt{d})P(\texttt{g}|\texttt{o})P(\texttt{!}|\texttt{g})$$

The pros and cons of this model are as follows: This is the simplest and easiest-to-implement model as it does not consider either character position or password length. This approach also allows us to construct a probability model where the sample size of character sub-strings is very large. However, it lacks detail in simulating the character usage patterns of users.

C. LAYERED MARKOV MODEL

The LMM, introduced in [21], is based on the fact that certain characters are more probable to appear in a certain position of a password. For example, users tend to put symbols at the end of passwords. Furthermore, such position dependency is also related to the length of the password; for example, the 8^{th} character of an 8-character password is not the same as that of a 10-character password. Hence, a distinct layered probability model will be constructed for every potential password length.

In this case, the probability of the password s, as mentioned Section III-B, can be expressed as follows:

$$P(s) = P^1_m(s_1)P^2_m(s_2|s_1)\ldots P^m_m(s_m|s_{m-1}) \tag{3}$$

where $P^i_m(\cdot)$ denotes the probability in the i-th position of an m-character password.

For example, the probability of the password `dog!` is: the probability of `d` being at the beginning of a 4-character password, multiplied by the probability that `o` follows `d` in the 2^{nd} position of a 4-character password, multiplied by the probability that `g` follows `o` in the 3^{rd} position of a 4-character

password, and multiplied by the probability that ! follows g in the 4^{th} position of a 4-character password.

$$P(\texttt{dog!}) = P_4^1(\texttt{d})P_4^2(\texttt{o}|\texttt{d})P_4^3(\texttt{g}|\texttt{o})P_4^4(\texttt{!}|\texttt{g})$$

The pros and cons of this model are as follows: This is the most complex model among the four models discussed in this paper. Considering both character position and password length seems to provide a more detailed and accurate assessment. However, it requires a sufficiently large training dataset for each password length to obtain a large enough sample size for sub-string sequences. As mentioned above, Markov-based PSMs are often trained using a dataset containing real-world leaked passwords. Such datasets often cannot meet the requirements. Clearly, longer passwords, for example, 13-character passwords, are less likely to be leaked in reality. Therefore, in the layered probability model for 13-character passwords, the number of samples of sub-string sequences obtained from this dataset will be very small. This can lead to a decrease in accuracy in the password evaluation process.

D. UNIQUE LAYERED MARKOV MODEL

This is a new model introduced in this paper. As mentioned before, it can be seen as a "hybrid model" between SMM and LMM as it only considers character position without regard to password length. Put simply, uLMM uses a unique layered probability model for all potential password lengths.

In this case, the probability of the password s, as mentioned Section III-B, can be expressed as follows:

$$P(s) = P^1(s_1)P^2(s_2|s_1)\dots P^m(s_m|s_{m-1}) \tag{4}$$

where $P^i(\cdot)$ denotes the probability in the i-th position.

For example, the probability of the password dog! is: the probability of d being at the beginning, multiplied by the probability that o follows d in the 2^{nd} position, multiplied by the probability that g follows o in the 3^{rd} position, and multiplied by the probability that ! follows g in the 4^{th} position.

$$P(\texttt{dog!}) = P^1(\texttt{d})P^2(\texttt{o}|\texttt{d})P^3(\texttt{g}|\texttt{o})P^4(\texttt{!}|\texttt{g})$$

The pros and cons of this model are as follows: Compared to LMM, this model is less complex as it only considers the character position. Considering the character position may provide higher detail than SMM in the password evaluation process. Ignoring the password length factor also helps improve the number of samples obtained for sub-string sequences. However, it is evident that the number of samples for sub-string sequences at larger positions, for example, the 13^{th} position, is still not significantly improved.

E. SIMPLE MARKOV MODEL WITH PASSWORD LENGTH CONSIDERATION

The SMMl is also a new model introduced in this paper. Similar to uLMM, it can be regarded as a "hybrid model" of SMM and LMM as it only considers the password length. In SMMl, a distinct simple probability model will be built for each potential password length.

In this case, the probability of the password s, as mentioned Section III-B, can be expressed as follows:

$$P(s) = P_m(s_1)P_m(s_2|s_1)\dots P_m(s_m|s_{m-1}) \tag{5}$$

where $P_m(\cdot)$ denotes the probability in an m-character password.

For example, the probability of the password dog! is: the probability of d being at the beginning of a 4-character password, multiplied by the probability that o follows d in a 4-character password, multiplied by the probability that g follows o in a 4-character password, and multiplied by the probability that ! follows g in a 4-character password.

$$P(\texttt{dog!}) = P_1(\texttt{d})P_2(\texttt{o}|\texttt{d})P_3(\texttt{g}|\texttt{o})P_4(\texttt{!}|\texttt{g})$$

The pros and cons of this model are as follows: By constructing separate simple probability models for each possible password length, SMMl can provide slightly more detailed evaluations than SMM. Additionally, solely utilizing simple probability models helps ensure the maintenance of the sample count for sub-string sequences. However, only considering the password length while disregarding the character position may lead to the loss of important information related to the characteristics of character usage during the evaluation process.

IV. SCENARIO A

In this section, we will describe the experiment in **Scenario A**: when a hacker has obtained a portion of passwords from a service A and uses them to attempt to crack the passwords of other users within the same service. Fig. 1 presents a simple illustration of this scenario.

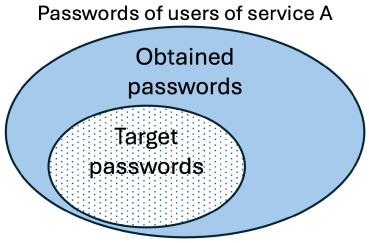

FIGURE 1. Illustration for scenario A.

A. PRELIMINARIES

1) DATASETS

To build training and testing datasets, we first collect real-world leaked passwords from Collection #1 [4], which is recognized as the largest security breach so far. Next, we cleanse the data by removing passwords containing non-ASCII characters. Furthermore, since passwords less than 6 characters are almost no longer used and passwords longer than 16 characters are also very rare, we continue to only retain passwords with lengths ranging from 6 to 15. Additionally, we also retain duplicate passwords to preserve

the probability distribution of user-generated passwords. In other words, reused passwords will appear many times. Finally, after the data cleansing process, we obtain dataset D containing 80 million passwords.

Our experimental environment is as follows: Apple M1 CPU, 16GB memory, macOS Monterey Version 13.4 operating system, and the programs are written using Python version 3.9.6.

2) PASSWORD EVALUATION CRITERIA

Regarding the criteria for password evaluation, [20] and [21] do not employ any specific evaluation criteria. Meanwhile, [22] introduced the concept of using the evaluation criteria based on 3σ rule [24] and showed that *SMM-PSM* can outperform other rule-based PSMs [25], [26], [35], and commercial PSMs used by top technological companies [36], [37], [38] when using these criteria.

The 3σ rule states that, for a dataset following a normal distribution, 68%, 95%, and 99.7%, respectively, of the values lie within 1, 2, and 3 standard deviations of the mean. In the case of datasets with non-normal distributions, the weak 3σ rule derived from Chebyshev's inequality shows that at least 75% and 88.8%, respectively, of the values lie within 2 and 3 standard deviations of the mean.

$$P(\mu - 2\sigma \leq X \leq \mu + 2\sigma) \geq 75\%$$
$$P(\mu - 3\sigma \leq X \leq \mu + 3\sigma) \geq 88.8\%$$

With the setting of $I_{normal} = \mu + 2\sigma$ and $I_{strong} = \mu + 3\sigma$, the passwords with a strength exceeding I_{strong} are considered to be significantly distant from at least 88.8% of leaked passwords and therefore, can be evaluated as strong. Similarly, the passwords with a strength exceeding I_{normal} have a significantly distance from 75% of leaked passwords and are rated as normal.

$$\begin{cases} I(s) < I_{normal} & : \text{Weak (score: 1/3)} \\ I_{normal} \leq I(s) < I_{strong} & : \text{Normal (score: 2/3)} \\ I_{strong} \leq I(s) & : \text{Strong (score: 3/3)} \end{cases} \quad (6)$$

In this paper, we apply these evaluation criteria to categorize passwords. Note that the values of μ and σ are derived from all passwords in the training dataset.

B. EXPERIMENT PROCEDURE

1) STEP 1: CASE SPLITTING

In this experiment, we assume that D contains all the data of a service and the attacker has obtained 80% of the passwords in D. Then, the attacker will use this set of passwords to attempt to crack the remaining 20% of passwords.

Under this assumption, we randomly divide D into two sub-datasets: one sub-dataset for training ($D_{training}$) and one sub-dataset for testing ($D_{testing}$). The data ratio between $D_{training}$ and $D_{testing}$ is 80:20. This means that the training set contains 64 million, whereas the testing set contains 16 million passwords. We use up to 64 million passwords for training to ensure that the number of types of sub-string sequences obtained in both LMM and uLMM would not be too small. This random splitting process will be conducted ten times, corresponding to ten cases from 1 to 10.

2) STEP 2: CALCULATING EXPECTED VALUE

As mentioned in Section I-B, evaluating a "strong" password as "weak" will not increase security risks, but rating a "weak" password as "strong" will cause trouble. Therefore, it is more crucial for a PSM to accurately classify a password containing security risks as "weak". To achieve this, we need to calculate the expected number of weak passwords contained within the 16 million passwords in $D_{testing}$.

First, we calculate the coverage level of the 3σ rule for the distribution of password strength across the entire 80 million passwords in D. By applying Eq. (1), we can calculate the mean μ and standard deviation σ of the passwords in D. Note that the probabilities of the passwords are calculated using Eqs. (2), (3), (4), and (5) in the cases of SMM, LMM, uLMM, and SMMl, respectively. From the values of μ and σ, we can determine the values of I_{normal} and I_{strong}. Then, we proceed to classify passwords in D and calculate the number of passwords classified as "weak". Assuming the distribution of password strength remains unchanged when randomly splitting D into two parts, we can calculate the expected value of "weak" passwords in the testing set as follows:

$$N_{Exp} = (\text{Number of "weak" password in } D) \times 0.2 \quad (7)$$

3) STEP 3: COMPARISON

Based on the criteria in Eq. (6), we classify passwords in the testing dataset $D_{testing}$ using the four types of Markov-based PSMs across ten cases and calculate the average number of passwords classified as weak (N_{Aver}) by each PSM. Then, we compute the deviation of this average value from the expected value as follows:

$$N_{Dif} = N_{Aver} - N_{Exp} \quad (8)$$

This value will indicate how well the PSM can detect weak passwords in this scenario. If N_{Dif} is positive, it means that the PSM is very good because it detects more passwords that have potential risks than the expected value. In this case, the larger the N_{Dif} is, the better the PSM performs. Conversely, if N_{Dif} is negative, it means that the PSM failed to detect some weak passwords. Therefore, in this case, the smaller N_{Dif} is, the higher the accuracy of the PSM.

C. EXPERIMENTAL RESULTS

First, we classify passwords in the dataset D based on the threshold values obtained after computing the mean value μ and the standard deviation σ relative to the passwords in D. Next, based on the number of passwords classified as weak in D, we calculate the values of N_{Exp} for the fours PSMs using Eq. (7). Then, we classify passwords in the testing dataset

$D_{testing}$ for cases ranging from 1 to 10 and calculate the value of N_{Aver}. Finally, we can obtain the value of N_{Dif} using Eq. (8). The number of passwords classified as weak in each case, along with the values of N_{Aver} and N_{Dif}, are presented in Table 2. The best result is highlighted in bold.

Overall, all four PSMs failed to detect the expected number of weak passwords, resulting in negative N_{Dif} values. This indicates that a lower N_{Dif} signifies better performance. Therefore, the effective order from highest to lowest is *SMMl-PSM*, *SMM-PSM*, *uLMM-PSM*, and *LMM-PSM*.

These results highlight the weaknesses of the two models considering the character positions, uLMM and LMM. As discussed in Section III-C, the detailed evaluation considering both character position and password length in *LMM-PSM* impacted the sample size of sub-string sequences obtained at each position for each possible length, leading to a significant reduction. The obtained samples were insufficient to accurately simulate user character usage patterns, resulting in the inferior performance of *LMM-PSM*. *LMM-PSM* missed an average of 3,396 weak passwords, a significantly large number (almost 6 times) compared to the best-performing PSM, *SMMl-PSM*. By eliminating the password length factor to improve the number of sample sub-string sequences at each position, the effectiveness improved significantly, with a significant reduction in the number of missed weak passwords to only 866 with *uLMM-PSM*.

Meanwhile, *SMM-PSM*, despite using the simplest model, yielded relatively good results, missing only an average of 616 passwords. As mentioned in Section III-B, disregarding the character position and password length allows for obtaining a vast number of samples for all sub-string sequences. However, as anticipated, this approach is overly simplistic and lacks detail. Therefore, by enhancing the detail while still ensuring an adequate number of samples for character sequences, *SMMl-PSM* yielded the best evaluation results, missing only 567 weak passwords.

These results emphasize the importance of the number of sample character sequences in the evaluation process. For this scenario, we can conclude that *SMMl-PSM* demonstrates the highest effectiveness.

TABLE 2. Number of passwords classified as weak in each case, along with the values of N_{Exp}, N_{Aver} and N_{Dif} - scenario A.

Case	Model			
	SMM	SMMl	uLMM	LMM
1	15,315,905	15,334,856	15,317,279	15,322,662
2	15,315,687	15,336,339	15,316,788	15,322,812
3	15,316,772	15,337,821	15,317,964	15,323,674
4	15,317,417	15,337,696	15,317,758	15,323,342
5	15,316,762	15,335,936	15,316,859	15,322,850
6	15,315,512	15,335,359	15,316,986	15,322,813
7	15,316,727	15,337,062	15,317,994	15,323,358
8	15,318,058	15,337,523	15,319,446	15,324,845
9	15,317,181	15,336,222	15,317,718	15,323,588
10	15,316,596	15,335,792	15,318,410	15,324,620
N_{exp}	15,317,277	15,337,027	15,318,586	15,326,852
N_{Aver}	15,316,661	15,336,460	15,317,720	15,323,456
N_{dif}	-616	**-567**	-866	-3396

V. SCENARIO B

In this section, we will describe the experiment in **Scenario B**: when a hacker has obtained passwords from one service and attempts to crack the passwords of users in another service. Fig. 2 presents a simple illustration of this scenario.

A. PRELIMINARIES

When considering this scenario, we consider D contains passwords from a service that the hacker has already obtained and used to attempt to crack the passwords in another service. In other words, we utilize dataset D, mentioned in Section IV-A, as the training dataset.

For the testing datasets, we also use leaked passwords collected from Collection #1 [4], but from different services than the one used to create D. Therefore, we can consider these password datasets as being used by another service that the hacker is attempting to crack using D. These passwords have indeed been leaked, so we can consider them all "weak". Therefore, theoretically, they should not be used. Based on this consideration, we expect that the PSMs will detect as many weak passwords as possible among these passwords. That is to say, the more passwords classified as weak, the better the PSM.

In this experiment, we conducted tests on five different datasets. Each dataset contains 16 million passwords and was created by collecting leaked passwords from five different services. We classified the passwords in these testing datasets and calculated the average values for the five cases.

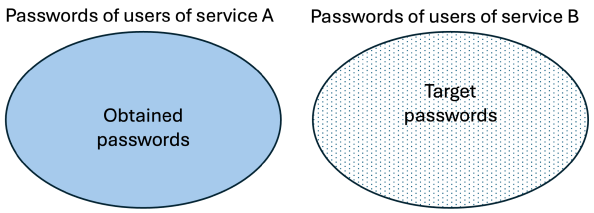

FIGURE 2. Illustration for scenario B.

TABLE 3. Number of passwords classified as weak - scenario B.

Case	Model			
	SMM	SMMl	uLMM	LMM
1	15,311,893	15,287,534	15,227,832	15,171,819
2	15,067,431	15,045,899	14,974,899	14,934,249
3	15,170,566	15,161,817	15,097,806	15,072,315
4	15,111,819	15,197,322	15,162,698	15,132,399
5	15,372,880	15,395,649	15,370,874	15,366,299
Aver.	15,206,917	**15,217,644**	15,166,821	15,135,416

B. EXPERIMENTAL RESULTS

Table 3 presents the number of weak passwords detected by the four PSMs and the average values across the five test cases. The best result is highlighted in bold. It is worth noting that, as mentioned before, password distribution

varies among different websites; therefore, we do not utilize the expected numbers of weak passwords as a comparison standard here. As mentioned above, the larger this number, the better for reducing security risks. Based on Table 3, we can observe that the effectiveness ranking is similar to the results in the experiment for **Scenario A**.

Overall, the Markov-based PSMs with evaluation criteria based on the 3σ rule perform very well in detecting weak passwords. Despite having the lowest effectiveness, *LMM-PSM* detected 15,135,416 weak passwords, which is approximately 94.60% of the total passwords identified as weak. *uLMM-PSM* slightly outperforms *LMM-PSM* by detecting 15,166,821 weak passwords, equivalent to 94.79% of the total passwords, which is 31,405 more than *LMM-PSM*. Ranked second, *SMM-PSM* also brings about relatively high effectiveness by detecting 15,206,917 weak passwords, equivalent to 95.04% of the total. Lastly, *SMMl-PSM* indicates that 95.11% of the total passwords are weak, precisely 15,217,644 passwords.

Once again, these results highlight the effectiveness of *SMMl-PSM*. Combined with the results in the experiment for **Scenario A**, we conclude that *SMMl-PSM* is the best among the four PSMs described in this paper. These findings once again underscore the importance of ensuring an adequate sample size for sub-string sequences when using Markov models to evaluate passwords. In the next section, we will conduct a detailed analysis of this matter.

VI. ANALYSIS RELATED TO SUB-STRING SEQUENCES

In this section, we will conduct an analysis related to the sample size of sub-string sequences from 80 million passwords in D. Table 4 presents the number of passwords with lengths ranging from 6 to 15 in D. We can observe that 8-character passwords are the most commonly used, and this number decreases as the length increases. The number of 15-character passwords is only around 540k, a very small number compared to the total of 80 million passwords. This clearly affects the models that consider the position of the characters, such as LMM or uLMM.

As mentioned in Section I-A, we assume that passwords are created from 94 printable characters. Therefore, a total of 8,836 types of 2-gram sub-string sequences can be generated from these 94 characters. Although in reality, some sequences may not be used in password creation due to overlapping with control characters, we do not pay much attention to this point.

In practice, after analyzing dataset D, up to 8,355 types of sub-string sequences have been used in the case of SMM, accounting for 93.55% of the total possible number. The total number of 2-gram sub-string sequences obtained from D is 592,622,378 sequences. This means, on average, each sub-string sequence is used approximately 70,930 times. This is a very high number, indicating that most of the sub-string sequences have been used by users when creating passwords.

In the case of SMMl, the simple probability model for 8-character passwords has the highest number of 2-gram sub-string sequences used, with 7,717 types (87.33%). On average, each type is used approximately 20,626 times, which is still incredibly impressive. For the probability model for 15-character passwords, despite having only over 540k passwords, there are still 6,468 types (73.20%) of 2-gram sub-string sequences used, with each type being used an average of 1,172 times. This is a respectable number in our assessment.

For the two models considering the position of characters, uLMM and especially LMM, this number is significantly lower. With LMM, the highest average number of samples obtained is in the probability model for 8-character passwords. Precisely, there are 6132, 6032, 6131, 6262, 6332, 6365, and 6436 types for positions from the 1^{st} to the 7^{th}. The highest number is 6,436 (72.83%) sequences used in the 7^{th} position, which is even less than the number of samples obtained from 15-character passwords in SMMl. On average, there are only 6,241 types for each position and each type is used approximately 3,643 times. These numbers are even lower in the probability models for passwords longer than 8 characters. For example, in the model for 15-character passwords, there are only an average of 4,278 types for each position and each type is averagely used only 126 times. These numbers are too low and certainly insufficient to evaluate user character usage trends.

Meanwhile, with uLMM, the number of sample types is significantly improved when using only one probability model for classification. Although there are only 4,304 types used at the 14^{th} position, with each type being used on average 125 times, there are up to 7,432 types (equivalent to 84.11%) used at the 5^{th}, with each type being used on average 10,764 times. This has significantly improved the effectiveness compared to LMM.

The results above provide us with preliminary insights into how the number of types of 2-gram sub-string sequences obtained and the frequency of their usage affect the accuracy of Markov-based PSMs. These findings will be beneficial for further developing factors to enhance the effectiveness of these PSMs in the future.

TABLE 4. Number of passwords for each length.

Length	Number of password	Length	Number of password
6	14,109,950	11	4,169,655
7	11,001,163	12	2,641,684
8	22,739,110	13	1,328,759
9	11,893,217	14	822,280
10	10,752,438	15	541,744

VII. DISCUSSION AND FUTURE WORK

We analyzed the influence of the number of types of sub-string sequences used as well as their frequencies on the password evaluation process. We also believe that considering the

length of the password, i.e., using separate probability models for different lengths, partly encapsulates factors related to character positions. To illustrate, consider the occurrence positions of vowels in passwords. The positions of vowels in a 10-character password are likely to differ from their positions in a 6-character password. Therefore, solely using simple probability models trained separately for possible password lengths may sufficiently encompass the character position factor while still ensuring the quality of the number of types of character sequences. However, this issue is speculative, and we will develop more specific analyses in the future.

The paper [22] showed the effectiveness of *SMM-PSM* in extracting the influence of linguistic factors on user-created passwords. However, the strength of user-created passwords is also influenced by not only linguistic factors but also various other factors, such as keyboard layout, Leet transformation, or the presence of symbols. For instance, a password that incorporates Leet transformation may appear stronger to humans, but it may not significantly enhance the resistance of the password against automated attacks. Similarly, the inclusion of symbols in a password may increase its complexity, but may not improve its resistance against dictionary attacks or probabilistic attacks. This is an interesting aspect that we aim to study further.

We also aim to develop a method that can suggest users change their chosen password in a way that retains its ease of remembering but increases its strength based on user factors such as nationality or language used. This method is expected to be highly practical.

VIII. CONCLUSION

The main focus of this paper is to revisit the effectiveness of the conventional Markov-based PSMs and propose new models with higher performance. Firstly, we introduced the concept of applying two new Markov models, SMMl and uLMM, which are considered as hybrid models of conventional Markov models used in previous research. Then, after clearly defining two possible scenarios, we compared the effectiveness of four Markov-based PSMs using a dataset containing real-world leaked passwords based on these scenarios. The experimental results showed that SMMl, one of the two models we proposed, provided the highest effectiveness in both scenarios. From these results, we recognized the influence of the number of types of substring sequences used and the number of times they are used, and therefore, conducted an analysis related to this issue. This analysis provides us with insights into the development of PSMs in the future.

Future work will focus on developing methods to suggest password changes to users that increase password strength. Additionally, we aim to conduct in-depth analyses on the influence of factors such as keyboard layout and Leet transformation on the strength of user-created passwords.

We recognize the growing challenge of selecting strong passwords and avoiding password reuse in light of the increasing number of online accounts. However, Markov-based PSMs offer high accuracy, operate efficiently even with modest computational resources, and enable numerical comparison of password strengths. We hope that this paper contributes to helping users better understand Markov-based PSMs and support their effective application in creating strong passwords.

REFERENCES

[1] R. Andrews, D. A. Hahn, and A. G. Bardas, "Measuring the prevalence of the password authentication vulnerability in SSH," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–7.
[2] K. Garrett, S. R. Talluri, and S. Roy, "On vulnerability analysis of several password authentication protocols," *Innov. Syst. Softw. Eng.*, vol. 11, pp. 167–176, Sep. 2015.
[3] W. Han, Z. Li, L. Yuan, and W. Xu, "Regional patterns and vulnerability analysis of Chinese web passwords," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 2, pp. 258–272, Feb. 2016.
[4] T. Hunt. (2019). *The 773 Million Record*. [Online]. Available: https://www.troyhunt.com/the-773-million-record-collection-1-data-reach/
[5] D. Miessler. (2017). *711 Million Email Accounts Susceptible To New Spambot*. [Online]. Available: https://www.theverge.com/2017/8/31/16232144/onliner-largest-malware-spambot
[6] D. Miessler. (2019). *763 Million Records Exposed in Verifications.Io Data Breach*. [Online]. Available: https://www.idtheftcenter.org/post/763-million-records-exposed-in-verifications-io-data-breach/
[7] N. Woods and M. Siponen, "Improving password memorability, while not inconveniencing the user," *Int. J. Hum.-Comput. Stud.*, vol. 128, pp. 61–71, 2019.
[8] M. Bishop and D. V. Klein, "Improving system security via proactive password checking," *Comput. Secur.*, vol. 14, no. 3, pp. 233–249, 1995.
[9] J. Yan, A. Blackwell, R. Anderson, and A. Grant, "Password memorability and security: Empirical results," *IEEE Secur. Privacy*, vol. 2, no. 5, pp. 25–31, Sep./Oct. 2004.
[10] A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang, "The tangled web of password reuse," in *Proc. NDSS*, vol. 14, 2014, pp. 23–26.
[11] W. Han, Z. Li, M. Ni, G. Gu, and W. Xu, "Shadow attacks based on password reuses: A quantitative empirical analysis," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 2, pp. 309–320, Mar. 2018.
[12] Kaspersky. (2023). *Password Checker*. [Online]. Available: https://password.kaspersky.com
[13] NordPass. (2023). *How Secure is My Password?* [Online]. Available: https://nordpass.com/secure-password/
[14] A. T. Secure. (2023). *Password Checker*. [Online]. Available: https://www.allthingssecured.com/password-checker/
[15] R. Morris and K. Thompson, "Password security: A case history," *Commun. ACM*, vol. 22, no. 11, pp. 594–597, 1979.
[16] M. Weir, S. Aggarwal, M. Collins, and H. Stern, "Testing metrics for password creation policies by attacking large sets of revealed passwords," in *Proc. 17th ACM Conf. Comput. Commun. Secur.*, 2010, pp. 162–175.
[17] A. Narayanan and V. Shmatikov, "Fast dictionary attacks on passwords using time-space tradeoff," in *Proc. 12th ACM Conf. Comput. Commun. Secur.*, 2005, pp. 364–372.
[18] M. Dürmuth, F. Angelstorf, C. Castelluccia, D. Perito, and A. Chaabane, "OMEN: Faster password guessing using an ordered Markov enumerator," in *Proc. Int. Symp. Eng. Secure Softw. Syst.*, Milan, Italy, 2015, pp. 119–132.
[19] M. Weir, S. Aggarwal, B. D. Medeiros, and B. Glodek, "Password cracking using probabilistic context-free grammars," in *Proc. 30th IEEE Symp. Secur. Privacy*, May 2009, pp. 391–405.
[20] C. Castelluccia, M. Dürmuth, and D. Perito, "Adaptive password-strength meters from Markov models," in *Proc. NDSS*, 2012.
[21] J. Galbally, I. Coisel, and I. Sanchez, "A probabilistic framework for improved password strength metrics," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, Oct. 2014, pp. 1–6.
[22] B. L. T. Thai and H. Tanaka, "A statistical Markov-based password strength meter," *Internet Things*, vol. 25, Apr. 2024, Art. no. 101057.

[23] X. D. C. D. Carnavalet and M. Mannan, "A large-scale evaluation of high-impact password strength meters," *ACM Trans. Inf. Syst. Secur. (TISSEC)*, vol. 18, no. 1, pp. 1–32, 2015.
[24] F. Pukelsheim, "The three sigma rule," *Amer. Statistician*, vol. 48, no. 2, pp. 88–91, 1994.
[25] W. Burr, D. F. Dodson, E. Newton, R. Perlner, W. Polk, S. Gupta, and E. Nabbus, "Nist special publication 800–63–2 electronic authentication guideline," Comput. Secur. Division, Inf. Technol. Lab., Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. SP 800-63-2, 2013.
[26] G. Hu, "On password strength: A survey and analysis," in *Proc. Int. Conf. Software Eng., Artif. Intell., Netw. Parallel/Distrib. Comput.*, 2017, pp. 165–186.
[27] J. Galbally, I. Coisel, and I. Sanchez, "A new multimodal approach for password strength estimation—Part I: Theory and algorithms," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 12, pp. 2829–2844, Dec. 2017.
[28] Q. Dong, C. Jia, F. Duan, and D. Wang, "RLS-PSM: A robust and accurate password strength meter based on reuse, leet and separation," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 4988–5002, 2021.
[29] D. V. Bailey, M. Durmuth, and C. Paar, "Statistics on password re-use and adaptive strength for financial accounts," in *Proc. Int. Conf. Secur. Cryptogr. Netw.*, Amalfi, Italy, 2014, pp. 218–235.
[30] M. Dupuis and F. Khan, "Effects of peer feedback on password strength," in *Proc. APWG Symp. Electron. Crime Res. (eCrime)*, May 2018, pp. 1–9.
[31] M. Golla and M. Dürmuth, "On the accuracy of password strength meters," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2018, pp. 1567–1582.
[32] W. Melicher, B. Ur, S. M. Segreti, S. Komanduri, L. Bauer, N. Christin, and L. F. Cranor, "Fast, lean, and accurate: Modeling password guessability using neural networks," in *Proc. USENIX Secur. Symp.*, 2016, pp. 175–191.
[33] D. Pasquini, G. Ateniese, and M. Bernaschi, "Interpretable probabilistic password strength meters via deep learning," in *Proc. Eur. Symp. Res. Comput. Secur.*, Guildford, U.K., 2020, pp. 502–522.
[34] S. J. Kim and B. M. Lee, "Multi-class classification prediction model for password strength based on deep learning," *J. Multimedia Inf. Syst.*, vol. 10, no. 1, pp. 45–52, 2023.
[35] W. Ma, J. Campbell, D. Tran, and D. Kleeman, "A conceptual framework for assessing password quality," *Int. J. Comput. Sci. Netw. Secur.*, vol. 7, no. 1, pp. 179–185, 2007.
[36] Google. (2023). *Homepage*. [Online]. Available: https://www.google.com
[37] Apple. (2023). *Homepage*. [Online]. Available: https://www.apple.com
[38] Dropbox. (2023). *Homepage*. [Online]. Available: https://www.dropbox.com

BINH LE THANH THAI received the B.E. and M.E. degrees from the National Defense Academy of Japan, in 2021 and 2023, respectively, where he is currently pursuing the Ph.D. degree. His research interests include information theory, cryptography, and password security.

HIDEMA TANAKA received the B.E., M.E., and Ph.D. degrees from Tokyo University of Science, in 1995, 1997, and 2000, respectively. He was the Director of the Security Fundamentals Laboratory, National Institute of Information and Communications Technology, until 2011. Currently, he is a Professor with the Department of Computer Science, National Defense Academy of Japan.