Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8837210, 22 pages
https://doi.org/10.1155/2020/8837210

Research Article

Modified Password Guessing Methods Based on TarGuess-I

Zhijie Xie,[1] Min Zhang,[1] Yuqi Guo,[2] Zhenhan Li,[1] and Hongjun Wang[1]

[1]*College of Electronic Engineering, National University of Defense Technology, Hefei 230037, China*
[2]*School of Electronic and Computer Science, Peking University, Beijing 100871, China*

Correspondence should be addressed to Min Zhang; zhangmindy@nudt.edu.cn

Received 3 September 2020; Revised 29 September 2020; Accepted 17 October 2020; Published 2 December 2020

Academic Editor: Ding Wang

TarGuess – I is a leading online targeted password guessing model using users' personally identifiable information (PII) proposed at ACM CCS 2016 by Wang et al. It has attracted widespread attention in password security owing to its superior guessing performance. Yet, after analyzing the users' vulnerable behaviors of using popular passwords and constructing passwords with users' PII, we find that this model does not take into account popular passwords, keyboard patterns, and the special strings. The special strings are the strings related to users but do not appear in the users' demographic information. Thus, we propose TarGuess – I^{+KPX}, a modified password guessing model with three semantic methods, including (1) identifying popular passwords by generating top-300 lists from similar websites, (2) recognizing keyboard patterns by relative position, and (3) catching the special strings by extracting continuous characters from user-generated PII. We conduct a series of evaluations on six large-scale real-world leaked password datasets. The experimental results show that our modified model outperforms TarGuess – I by 2.62% within 100 guesses.

1. Introduction

Password-based authentication is still an essential method in cybersecurity [1]. To understand password security, people have gone through several stages, from some heuristic methods that lack theoretical foundations to those algorithms that conform to strict probability models [2]. Since the emergence of Markov-based [3, 4] and probabilistic context-free grammar- (PCFG-) based [5, 6] password guessing algorithms, trawling password guessing has been intensively studied [7–10]. Recently, several large-scale personal information database leakage events have caused widespread concern in the security community [11–14]. With the development of related researches, the targeted password guessing algorithms using users' personally identifiable information (PII) have emerged [15–17].

Das et al. [15] studied the threat posed by password reuse and proposed a cross-site cracking algorithm for the first time. However, without considering common popular passwords, this algorithm is not optimal. Li et al. [16] studied to what extent a user's PII would affect password security and proposed a semantics-rich model, Personal-PCFG, which adopted with length-based PII matching and substitution. But it could not accurately capture the usage of users' PII, thus greatly hindering the cracking efficiency. As a seminal work of password guessing, Wang et al. [17] put forward a framework, TarGuess, which systematically characterizes typical targeted guessing scenarios. It contains a type-based PII semantics-aware PCFG and recognitions of password reuse behaviors, both of which significantly outperform the former cracking algorithms. Their work has motivated successive new studies on password security [18–20] and even led the revision of the NIST SP800-63-3 [21, 22].

TarGuess framework is proposed after an in-depth analysis of users' vulnerable behaviors. The framework includes four password guessing models, TarGuess – I ~ IV, for four attacking scenarios #1~#4. TarGuess – I caters for scenario #1, where the attacker is equipped with the users' explicit PII, such as name, birthday, and phone number. The users' explicit PII can be easily obtained from the Internet [23] and can be the building blocks of passwords [17]. The rest three models required user information such as PII attributes that play an implicit role in passwords (e.g., gender and profession) and/or sister passwords that were leaked from

Table 1: Statistics of the experimental result of TarGuess – I^{+KPX} model.

Training set	Testing set	Guess number				R_n			
		10	100	10^3	10^4	10~100	100 ~ 10^3	10^3 ~ 10^4	Avg
12306	**Aipai**	0.078	0.153	0.214	0.270	-0.65%	0.92%	2.91%	1.41%
	Dodon	0.125	0.209	0.270	0.332	2.62%	6.06%	3.99%	4.51%
	Senda	0.109	0.195	0.259	0.320	0.72%	4.25%	2.11%	2.68%
	Tinya	0.130	0.198	0.248	0.316	1.52%	3.47%	1.90%	2.46%
Youku	**Aipai**	0.075	0.149	0.211	0.259	-0.75%	0.47%	3.39%	1.42%
	Dodon	0.126	0.206	0.269	0.321	2.16%	5.73%	4.56%	4.54%
	Senda	0.107	0.191	0.256	0.307	0.00%	3.61%	2.07%	2.28%
	Tinya	0.130	0.195	0.241	0.300	0.85%	2.80%	1.87%	2.04%

the user's other accounts. This work mainly focuses on scenario #1. As more users' PII is being leaked these days, attacking scenario #1 becomes more practical.

Wang et al. [17] showed that their TarGuess – I model is more efficient than previous algorithms using users' PII to crack passwords, which can gain success rates over 20% with just 100 guesses. Whether it can continue to improve the success rate of guessing models has become a peer research hotspot [24]. After analyzing the users' vulnerable behaviors in constructing passwords based on TarGuess – I, we find that there are some missing attributes in TarGuess – I. Therefore, based on TarGuess – I, we put forward three modified methods and conduct a series of experiments to examine their feasibility. In the end, we proposed a modified model: TarGuess – I^{+KPX}, which includes our three improvements. Extensive evaluations have shown that, TarGuess – I^{+KPX} outperforms its original model by 2.62% within 100 guesses.

1.1. Our Triple Contributions. This article is an extended version of the paper [25]. In this work, we make the following key contributions.

1.1.1. A Modified Password Guessing Model. After analyses of the users' vulnerable behaviors in constructing passwords on a total of 163,041,192 public leaked data based on TarGuess – I, we find that some effective semantic tags have not been testified and employed in TarGuess – I. To fill the gap, we make use of the adaptiveness of TarGuess – I PII tags and define three new tags: the popular password tag P, the keyboard pattern tag K, and the special string tag X. This gives rise to a variant of TarGuess-I, we call it TarGuess – I^{+KPX}.

1.1.2. An Extensive Evaluation. To demonstrate the feasibility of these incremental tags, we perform a series of experiments on six large-scale real-world leaked datasets. The experimental results show that our single-tag-modified models (TarGuess – I with each tag we defined individually) outperform TarGuess – I by 0.75% in optimal and 0.33% on average within 100 guesses. Particularly, our modified model TarGuess – I^{+KPX} works best among the 10 models we experimented with. It can successfully crack a target user's password with an optimal chance of 20.9% within 100 guesses when it gets the same users' PII as TarGuess – I gets, which outperforms TarGuess – I by 2.62% (the target user is come from the four sites, see Table 1).

1.1.3. A New Insight. We propose a new method to modify the password guessing model: parsing the passwords into the special strings X tag, such as anniversary dates or someone's name, that do not appear in users' demographic PII. It can be identified by adding incremental information to the model or refining the model recognitions of user-generated PII (such as e-mail addresses and user names). This method gives a new insight into targeted password guessing.

2. Preliminaries

TarGuess – I is a targeted guessing model using users' PII and builds on the PCFG-based algorithm. This section explicates what kinds of users' vulnerable behaviors are considered in this work and gives a brief introduction to the PCFG-based algorithm and TarGuess – I.

2.1. Explication of Users' Vulnerable Behaviors. Users' vulnerable behaviors are the key influence factor of password crackability [26]. A series of related studies have been conducted since the pioneering work of Morris and Thompson in 1979 [27]. Part of the studies based on data analyses, such as [3, 12, 14, 28–31], the others based on user surveys, such as [15, 32–35]. In summary, the discovered users' vulnerable behaviors can be classified into the following three categories.

2.1.1. Popular Passwords. A large number of studies (such as [3, 14, 30]) have shown that users often choose simple words as passwords or make simple transformed strings to meet the requirements of the website password setting strategy, such as "123456a" meets the "alphanumeric" strategy. These strings, which are frequently used by users, called popular passwords. Furthermore, Wang et al. [36] have found that the Zipf distribution is the main cause of the aggregation of popular passwords.

2.1.2. Password Reuse. After a series of interviews to investigate how users cope with keeping track of many accounts and passwords, Stobert and Biddle [32] point out that users have more than 20 accounts on average. It is fairly impossible for them to create a unique password for each account, so

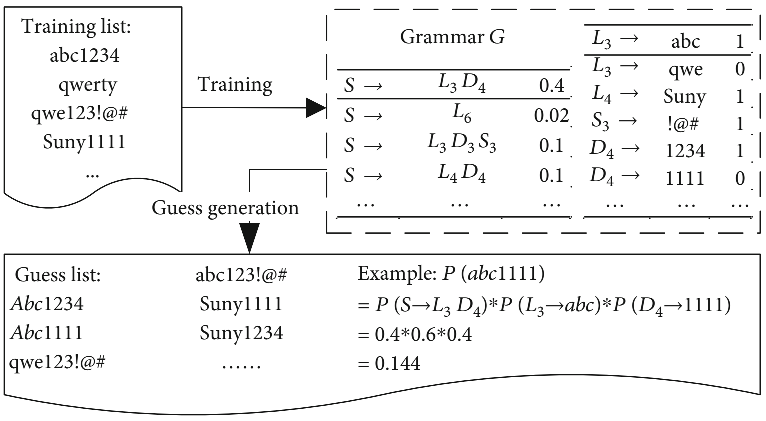


Figure 1: An illustration of PCFG-based algorithm.

reusing passwords is a rational approach. At the same time, password reuse is a vulnerable behavior; the key is how to reuse.

2.1.3. Passwords Containing Personal Information. Wang et al. [37] note that Chinese users tend to construct passwords with their pinyin name and relevant digits, such as phone number and birthdate, which are quite different from English users. They revealed a new insight into what extent users' native languages influence their passwords and what extent users' personal information plays a role in their passwords.

Considering that TarGuess – I caters for scenario #1, we only analyze the two categories of users' vulnerable behaviors (i.e., popular passwords and password containing personal information).

2.2. PCFG-Based Password Guessing Algorithm. Weir et al.'s PCFG-based algorithm [5] has shown a great success in dealing with trawling guessing scenarios [17]. The context-free grammar in [5] is defined as $\mathscr{G} = (\mathscr{V}, \Sigma, \mathscr{S}, \mathscr{R})$, where

(i) $\mathscr{V}$ is a finite set of variables

(ii) Σ is a finite set disjoint from $\mathscr{V}$ and contains all the terminals of $\mathscr{G}$

(iii) $\mathscr{S}$ is the start symbol and $\mathscr{S} \in \mathscr{V}$

(iv) $\mathscr{R}$ is a finite set of productions of the form: $\alpha \longrightarrow \beta$, where $\alpha \& \beta \in \mathscr{V} \cup \Sigma$

The core assumption of the algorithm is the segments of letters, numbers, and symbols in a password which are independent with each other. Thus, in $\mathscr{V}$ set, except for $\mathscr{S}$ start symbol, there are only L_n letters, D_n digits, and S_n symbols tag sets, where n represents the segment length, such as L_3 represents three-letter segments, and D_4 represents four-digit segments.

There are two phases in the algorithm, the training phase and the guess generation phase, as shown in Figure 1. In the training phase, the password is parsed into the *LDS* segments based on the length and type to generate the password base structure (the start symbol $\mathscr{S}$). Then, it counts the segment frequency table in each tag set and outputs the context-free grammar $\mathscr{G}$. In the guess generation phase, passwords are derived by the grammar $\mathscr{G}$ and the segment frequency tables. The final guess candidates are arranged based on the probability multiplied by all the frequency of segments in the password.

2.3. TarGuess-I Model. TarGuess – I is a semantics-aware PCFG model built by the type-based PII tags, which are firstly proposed by Wang et al. Besides the three basic tags *LDS* in the PCFG-based algorithm, the grammar $\mathscr{G}_{\mathscr{I}}$ in TarGuess – I includes six PII tags (such as N_n name, U_n user name, B_n birthday, T_n phone number, I_n id card, and E_n e-mail address). For each PII tag, its index number n is different from the *LDS* tags, which represents the type of generation rule for this PII. For example, N stands for name usage, while N_1 stands for the full name, and N_2 stands for the abbreviation of the full name (such as "Zhang San" abbreviated as "zs"). See Figure 2 for a specific description. The grammar $\mathscr{G}_{\mathscr{I}}$ is highly adaptive. It can be modified simply by adding incremental tags without changing the whole structure to confirm the function, which brings great convenience to our research.

As shown in Figure 3, for each user, the segment frequency table of each PII tag is generated through the user's PII. In the training phase, the password is firstly parsed with the PII segments into PII tags, and the rest of the segments are parsed into *LDS* tags. The guessing phase is similar to the PCFG-based algorithm, but a part of products are intermediate candidates consisting of PII tags (e.g., N_3B_5 and N_3 1234). These intermediate candidates will be matched by the segments from users' PII before be added to the final guess candidates.

3. Users' Vulnerable Behaviors in Constructing Passwords

In this section, we analyze the users' vulnerable behaviors based on real-world leaked data for inspiration to improve TarGuess – I. Because of the lack of studies on how Chinese users select passwords, we only focus on Chinese users. We dissect 163,041,192 leaked user passwords from the six websites (see Table 2) for analyzing. Hereinafter, the bold

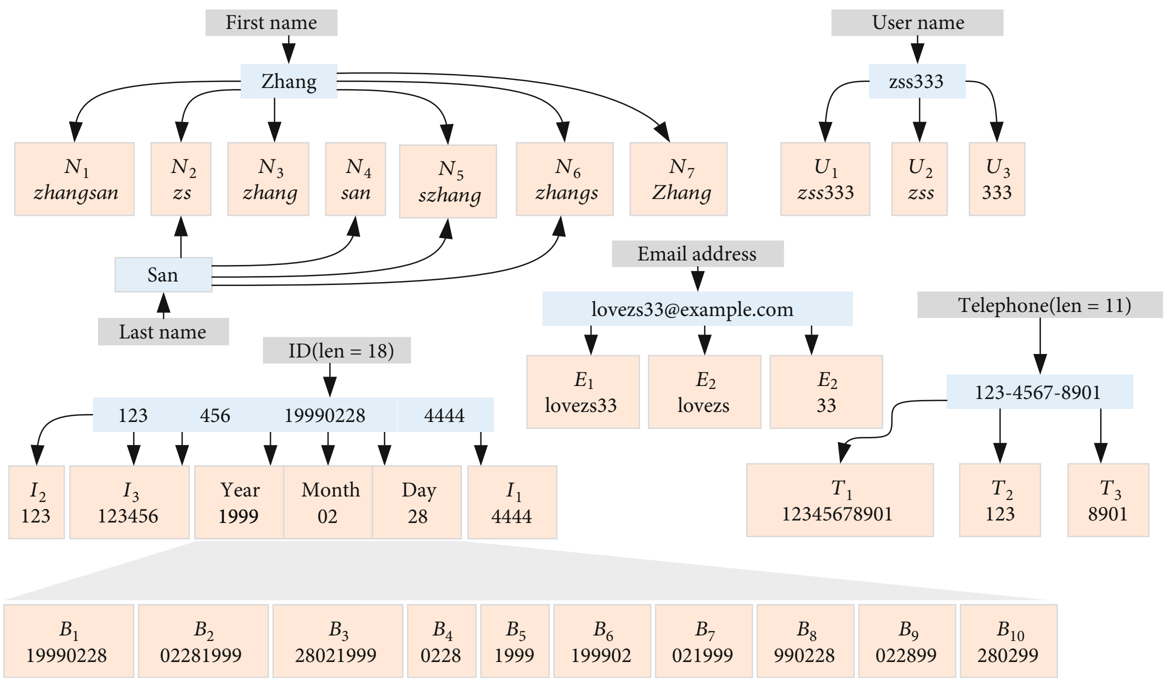


Figure 2: An illustration of TarGuess-I's PII tags generation.

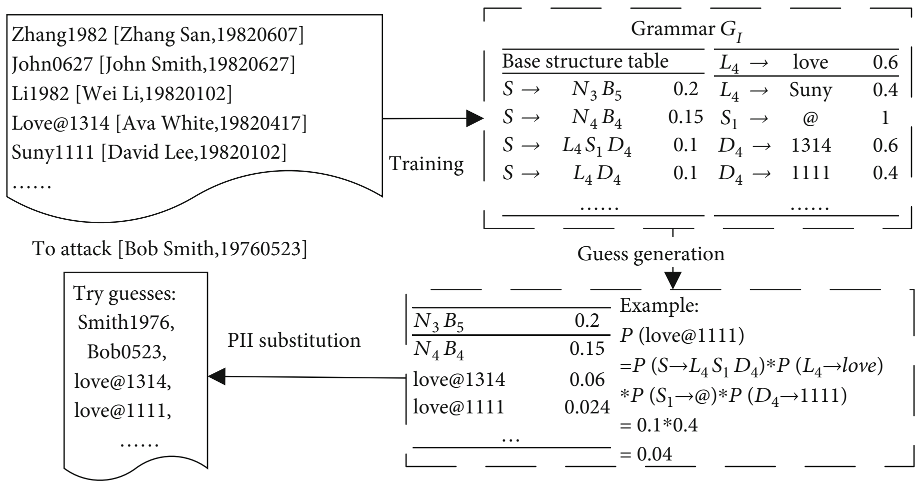


Figure 3: An illustration of TarGuess-I.

Table 2: Basic information about our datasets.

Dataset*	Web service	When leaked	Total	With PII	Content	Be used
Tianya (**Tinya**)	Social forum	2011	31,006,590	28,158	User name, PW, E-mail	[12, 14, 38, 39]
Dodonew (**Dodon**)	E-commerce	2011	16,258,891	21,854	User name, PW, E-mail	[12, 14, 17, 39]
Shengda (**Senda**)	Game	2011	15,313,334	38,203	User name, PW, E-mail	None
12306 (**12306**)	Train ticketing	2014	232,884	232,884	PW, E-mail, PII	[16, 17]
Aipai (**Aipai**)	Video blogs	2016	7,682,232	27,917	User name, PW, E-mail	None
Youku (**youku**)	Audio visual	2016	92,547,261	134,863	PW, E-mail	None

*Hereinafter, each data source is referred as the bold shorthand notations in the brackets.

shorthand notation in the brackets of Table 2 represents the source of each dataset. The datasets were hacked by attackers or leaked by insiders and disclosed publicly on the Internet, and some of them have been used in the former research (as shown in Table 2). Due to the lack of datasets containing users' PII, we choose the unique PII (e.g., e-mail address) in

Table 3: Ranking and proportion of top-10 popular passwords.

Rank	Aipai	Tinya	Senda	Dodon	Youku	12306
1	1q2w3e4r	123456	123456	123456	123456	123456
2	5201314	111111	111111	111111	a123456	a123456
3	1qaz2wsx	000000	5201314	123456789	111111	123456a
4	qq123456	123456789	123456789	5201314	123456789	woaini1314
5	1314520	5201314	123123	123123	5201314	5201314
6	aptx4869	123123	1q2w3e4r	a123456	123456a	111111
7	1q2w3e4r5t	1qaz2wsx	000000	woaini1314	123123	qq123456
8	123456789a	7758521	a123456	1314520	123456789a	1qaz2wsx
9	a123456789	1314520	qq123456	123456a	000000	1q2w3e4r
10	abc123456	666666	woaini1314	qq123456	woaini1314	123qwe
%	0.51%	3.40%	2.00%	1.96%	2.33%	1.17%

Table 4: Component form distribution of the top-10^4 popular passwords (*Compos*: composite).

Form	Aipai	Tinya	Senda	Dodon	Youku	12306
Alpha	14.73%	10.89%	15.24%	11.46%	13.49%	7.21%
Digit	59.52%	63.12%	66.07%	39.17%	65.9%	46.93%
Symbol	0%	0.05%	0.02%	0.01%	0.08%	0.05%
Compos	25.75%	25.94%	18.67%	49.33%	20.53%	45.81%

12306 datasets to match passwords in other datasets. The sizes of matched datasets with PII from each dataset are shown in Table 2.

3.1. Analysis of Popular Passwords. According to the occurrence frequency, the top-10 popular passwords in six filtered databases with the proportion of them are calculated, and the results are shown in Table 3. It shows that 0.51% to 3.40% of users' passwords can be cracked successfully by just using the top-10 popular passwords. Chinese users prefer simple combinations of numbers (such as "123456," "111111," "000000") and the strings with the meaning of love (such as "5201314" and "woaini1314"). There are also some unique passwords in the top-10 list (such as "aptx4869" in **Aipai** and "7758521" in **Youku**). These passwords may come from the site's name or culture or maybe come from a large number of "ghost accounts" held by a particular user of the website. Besides, the passwords constructed with the QWERTY keyboard pattern (such as "1q2w3e4r" and "1qaz2wsx") also account for a certain proportion in the popular passwords.

The statistical results of the component form of the top-10^4 popular passwords, which are analyzed by the PCFG-based algorithm, are shown in Table 4. It illustrates that, though the majority of popular passwords are composed of pure numbers, composite passwords (e.g., a password included multiple types of characters) also account for a considerable part, especially 45.81% in **12306** and 49.33% in **Dodon**.

Since the grammar $\mathscr{G}_{\mathscr{T}}$ of TarGuess – I does not contain tags related to popular passwords, it could lower the success rate if the targeted users are likely to choose composite passwords. Because TarGuess – I is based on PCFG, which generates passwords according to the existing base structures generated from data and the elements in each set of tags. Thus, in the training phase, the model parses the composite password into *LDS* segments, and it might generate many invalid outputs at last, an illusion is shown in Figure 1. For instance, "1qaz2wsx," a password constructed with keyboard patterns, which is the 3rd popular password in **Aipai**, will be parsed into $D_1L_3D_1L_3$ by TarGuess – I. Meanwhile, "1" ranks the first in the set of D_1 tag, and "qaz" ranks the first in L_3. Therefore, in the guessing phase, the first password output with the base structure $D_1L_3D_1L_3$ is "1qaz1qaz." This password occupies a relatively small proportion in actual password distribution but ranks much higher in the output list, thus reducing the overall password guessing success rate. From this perspective, we come up with an idea to take popular passwords and the keyboard pattern into consideration for the training phase in TarGuess – I.

By analyzing popular passwords, we find that there are two missing attributes in the grammar $\mathscr{G}_{\mathscr{T}}$: the popular password tag P and the keyboard pattern tag K.

3.2. Analysis of Passwords Containing Personal Information. We adopt the TarGuess – I^{+PK} model to analyze the datasets. TarGuess – I^{+PK} is improved with the popular password tag P containing top-10^4 list and the keyboard pattern tag K. The rank of top-10 password base structures and the proportion of passwords containing PII tag or P tag are shown in Table 5. It indicates that about half of Chinese users generally construct passwords using PII or just choose popular ones. We speculate that the top-10 base structures of passwords should be related to the strings that are easy for users to remember. And we also find that some password base

Table 5: Top-10 base structures and proportion of the passwords containing PII and P tag.

Rank	Aipai	Tinya	Senda	Dodon	Youku	12306
1	P_1	P_1	P_1	P_1	P_1	P_1
2	D_7	D_6	D_7	E_1	D_6	D_6
3	N_2P_1	D_7	D_8	D_7	D_7	D_7
4	N_2D_7	D_8	D_6	D_6	D_8	N_2D_6
5	D_8	E_1	N_2P_1	D_8	N_2P_1	U_1
6	L_1D_7	D_{10}	E_1	L_1D_7	U_1	E_1
7	N_2D_6	B_8	N_2D_7	N_2D_7	U_3	D_8
8	D_6	D_9	L_1D_7	N_2D_6	E_1	N_2D_7
9	E_1	U_3	N_2D_6	U_1	P_1D_3	N_2P_1
10	U_1	B_1	U_3	U_3	N_2D_6	U_3
% of PII	34.24%	35.47%	33.80%	38.83%	32.97%	35.98%
% of P	12.72%	13.94%	18.14%	10.62%	21.07%	19.07%

structures in the top-10 list are not relevant to users' PII or containing P tag.

The strings which are accessible to memorize include users' PII conversions, keyboard patterns, and popular passwords. They also include the user-created strings that have special meaning for the user but are no equal importance to other users, we call them "the special strings." To give an example, we assume that $\mathscr{A}$ user creates a string "080405" as his password, then "080405" shall be special for him. But for another user $\mathscr{B}$, "080405" is nothing other than a common string; then, the probability of $\mathscr{A}$'s password containing this string shall be different from $\mathscr{B}$'s. Meanwhile, we cannot find the string "080405" in $\mathscr{A}$'s or $\mathscr{B}$'s demographic information (such as name, ID number, and telephone number). The special string cannot be extracted from the user's demographic information but may appear in some like the prefix of e-mail addresses and user names that the user-created strings, or in user's passwords from other servers.

The parse of users' PII in TarGuess – I also includes the user-generated strings, such as e-mail address E_n and user name U_n. However, the analyses of these two user-generated strings are not sophisticated enough. Only three parse type (Entire E_1&U_1, the first letter segments E_2&U_2 and the first digit segments E_3&U_3) are proposed. The special strings for each user, as the above said, the probability distribution is different. If we use the original TarGuess – I model, because of the lack of recognition of the special string, most of these segments will be parsed into typical LDS tags, merging the user behavior characteristics, thus hinder the effectiveness of password cracking. Therefore, we consider adding the special string tag X to the grammar $\mathscr{G}_{\mathscr{J}}$ of TarGuess – I.

We analyze the coverage of consecutive substrings of the e-mail address and user name in the password. The result is shown in Figure 4. It reveals that a significant number of user passwords do overlap user-created strings; thus, it gives us a new hint that when an attacker obtains information about a user that is not public or very useful, they may turn that information into a special string to participate in password guessing. This idea may serve as a new direction for further research.

4. Implementations of Modified Methods

After analyzing users' vulnerable behaviors in constructing passwords in Section 3, we find that TarGuess – I does not take into account three attributes, including popular passwords, keyboard patterns, and the special strings. Thus, we come up with three ideas for modifying TarGuess – I.

(1) Add the popular password tag P to the grammar $\mathscr{G}_{\mathscr{J}}$ and apply the popular password list generated from a dataset similar to the target website

(2) Add the keyboard pattern tag K to the grammar $\mathscr{G}_{\mathscr{J}}$ and identify password segments with physical location sequence in QWERTY keyboard

(3) Add the special string tag X to the grammar $\mathscr{G}_{\mathscr{J}}$ and extracted continuous characters from the user-generated PII

Figure 5 gives a brief explanation on how we try to modified TarGuess – I and generate TarGuess – I^{+KPX}. In this section, we will study for the implementations of these modified methods.

4.1. Popular Passwords. Add the popular password tag P_1 to the grammar $\mathscr{G}_{\mathscr{J}}$, and the set of elements in P_1 tag is a top-N popular password list based on the data statistics of similar websites. The number 1 in P_1 has no meaning but to conform to the grammar format. The parse of P_1 tag is shown in Figure 6.

In the training phase, the top-N list is matched with the password data by a regular expression. If the match occurs, the occurrence of the corresponding password in P_1 set is increased by 1. In the guess generation phase, the probabilities of containing P_1 password structures are multiplied by the frequencies of the corresponding password in the element set of P_1 as the final probabilities of output passwords.

To find out which N is the optimal parameter in setting the improved guessing model, we conduct a series of evaluations with top-N popular passwords based on **12306** data. Figure 7 is a contour plot displaying how top-N popular passwords influence the success rate. It can be seen that, within 100 guesses, the success rate increased slightly with the growing N, and the success rate is a bit higher with $N \subseteq (100 \sim 10^3)$ under 100 guesses. Figure 8 shows the similarities of top-N popular passwords between two different services. The similarity fluctuates greatly within top-100, and it tends to a stable peak when N is around at 300, then gradually reduces as N continues to grow. With the exception of **Aipai** and **12306**, the top-300 popular password list of each dataset has a similar top – 300 list from another dataset (their similarity >60%). The dispersion of shared password fractions implies that different types of services do impact on top popular passwords.

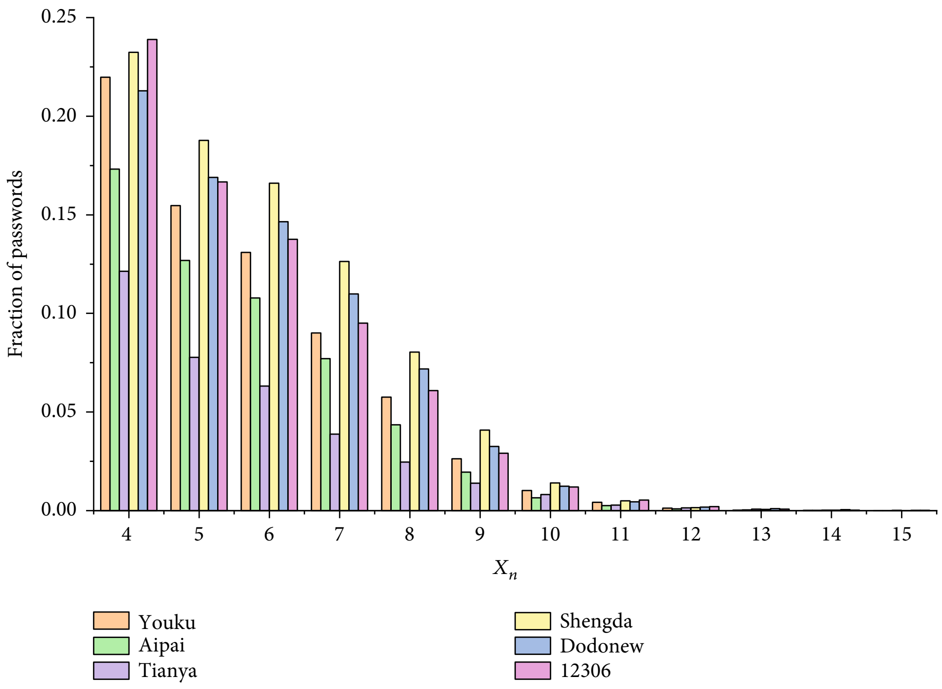


FIGURE 4: Fraction of the special string X_n in the password.

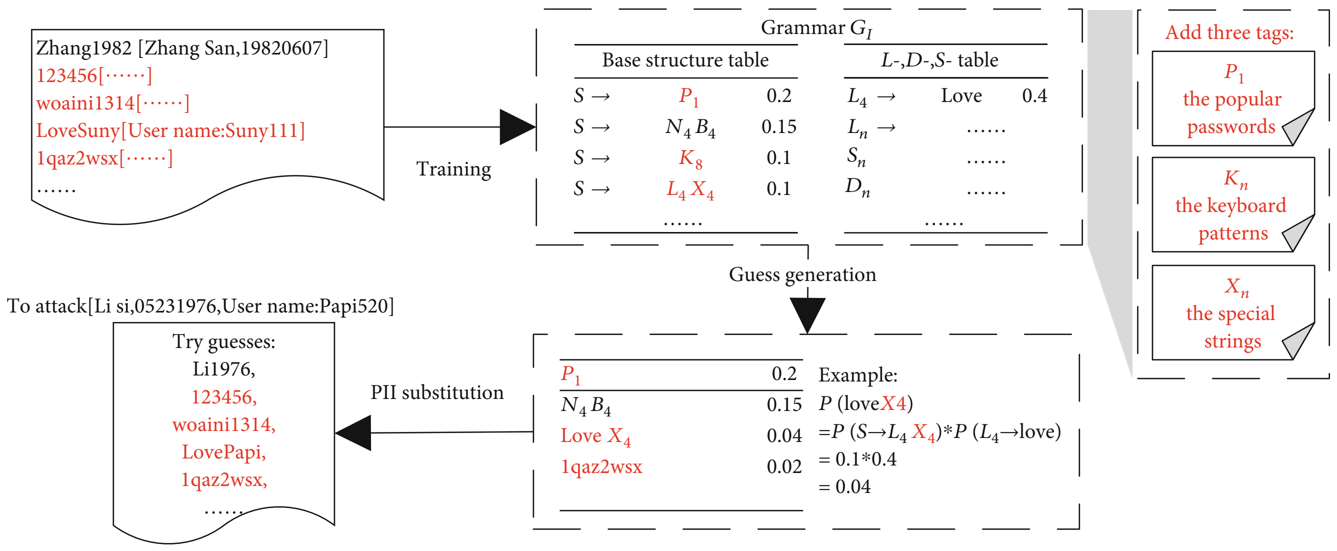

Zhang1982 [Zhang San,19820607]
123456[……]
woaini1314[……]
LoveSuny[User name:Suny111]
1qaz2wsx[……]
……
Training
Grammar G_I
Base structure table
$S \rightarrow$ P_1 0.2
$S \rightarrow$ $N_4 B_4$ 0.15
$S \rightarrow$ K_8 0.1
$S \rightarrow$ $L_4 X_4$ 0.1
……
L-,D-,S- table
$L_4 \rightarrow$ Love 0.4
$L_n \rightarrow$ ……
S_n ……
D_n ……
……
Add three tags:
P_1 the popular passwords
K_n the keyboard patterns
X_n the special strings
Guess generation
P_1 0.2
$N_4 B_4$ 0.15
Love X_4 0.04
1qaz2wsx 0.02
Example:
P (loveX4)
=P (S→$L_4 X_4$)*P (L_4→love)
= 0.1*0.4
= 0.04
PII substitution
To attack[Li si,05231976,User name:Papi520]
Try guesses:
Li1976,
123456,
woaini1314,
LovePapi,
1qaz2wsx,
……

FIGURE 5: A testing case of TarGuess – I^{+KPX} and our modified methods for it. The parts highlighted in red are the semantic tags we added and the additional password structures the model recognized after adding the semantic tags.

Based on the above experimental results, we set $N = 300$ in the cross-site password guessing scenario.

4.2. Keyboard Patterns. The process of keyboard pattern K_n is compliant with the left-hand side (*LHS*) principle. First, get the relative position of the character on the keyboard and then determine whether the latter character is adjacent to the previous character position. If the length of the string with adjacent characters len ≥ 4, then divide the segment into the keyboard order K_n variable, where $n = \text{len}$. An illusion of the K_n process is shown in Figure 9.

See Table 6 for the password base structure ranking and proportion with keyboard pattern K_n tags in our experimental datasets. K_8 represents the password generated by 8-length of keyboard patterns (such as "1qaz2wsx"). It should note that K_4K_4 is a password composed of 2 nonadjacent keyboard pattern strings, such as "1234asdf." Table 6 shows that the proportion of the passwords containing keyboard patterns is 0.88% to 1.37% in our experimental datasets.

4.3. The Special Strings. Considering that only two user-generated PII is needed in TarGuess – I, e-mail addresses

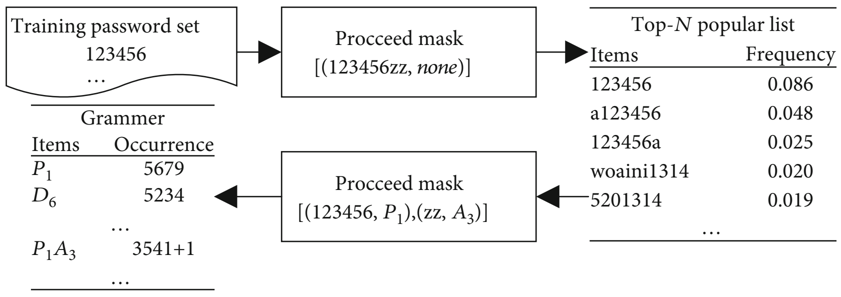


FIGURE 6: An illustration of P_1 tag parse.

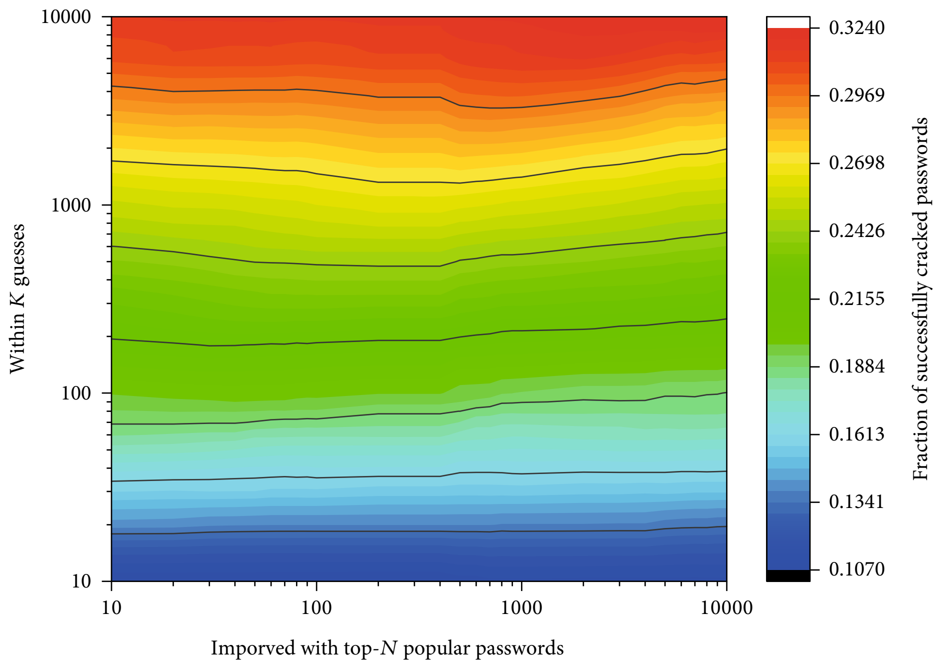


FIGURE 7: Experimental results of improved models with top-N popular passwords based on **12306** dataset. K represents the guess number. We use 60% of 12306 data for training and 40% for testing. Top-N popular password list is generated from the same service.

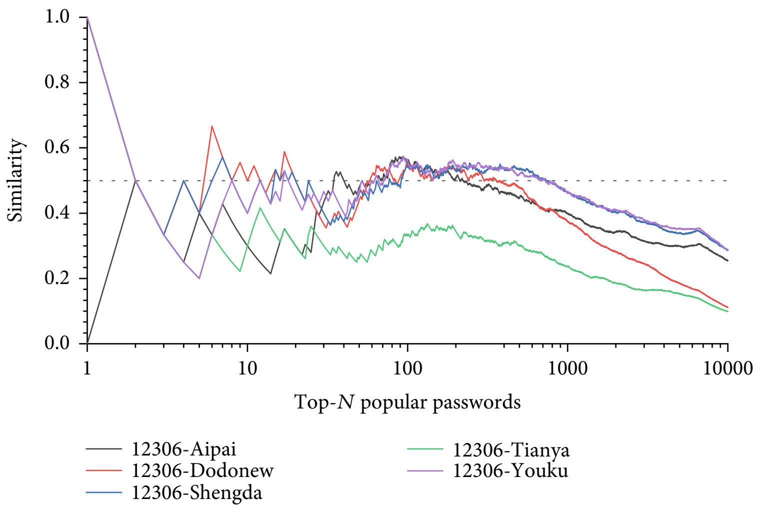
and user names, and the limitation of experimental resources, we only generated the elements of the special string X_n from these two PII. Since there are various and different ways for each user to generate special strings, it is difficult to categorize the generation methods of special strings uniformly and may cause sparse data. Therefore, n and m only classified according to the length of special strings and the position where the special string occurs. An illustration of the special string X_n process is shown in Figure 10.

Figure 11 shows the success rate of the X-tag-modified models compared to TarGuess – I. Each modified model has a different threshold number of the identification length for X tag. Unfortunately, we find that no matter how we change the threshold, the model with X tags does not work well. The reasons for the poor performance of the X tag may be as follows:

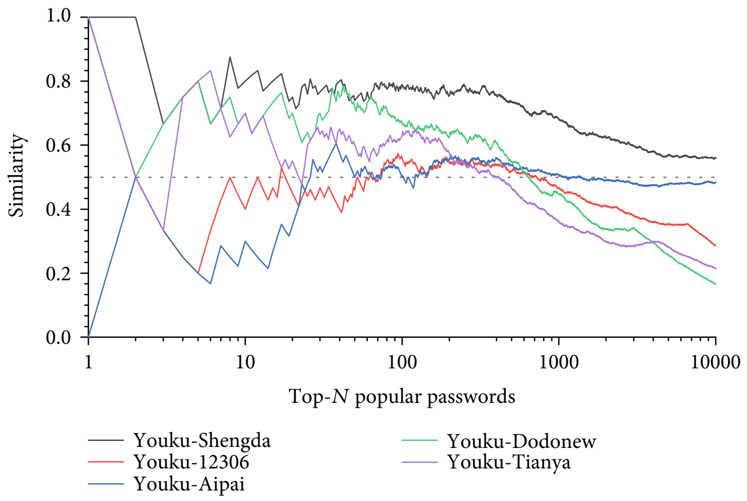
(1) This semantic tag is originally based on incremental information to improve the targeted password guessing model, that is, the user's other information besides users' demographic PII (such as work number, home address, and lover's name). However, due to the limitation of experimental conditions, we cannot obtain more incremental information, but can only make a finer segmentation from the user-generated PII (such as e-mail address and user name, which have been already analyzed in TarGuess – I)

(2) Our improved method of the special strings X is not in line with the habit of users setting passwords. We divide the user-generated PII according to length and calculate the relative position of substrings (as shown in Figure 10), which is a length-based method. That was also confirmed by Wang et al. to be insufficient for the analysis of users' behavioral characteristics. For a long user-generated PII string, this method generates too many invalid substrings

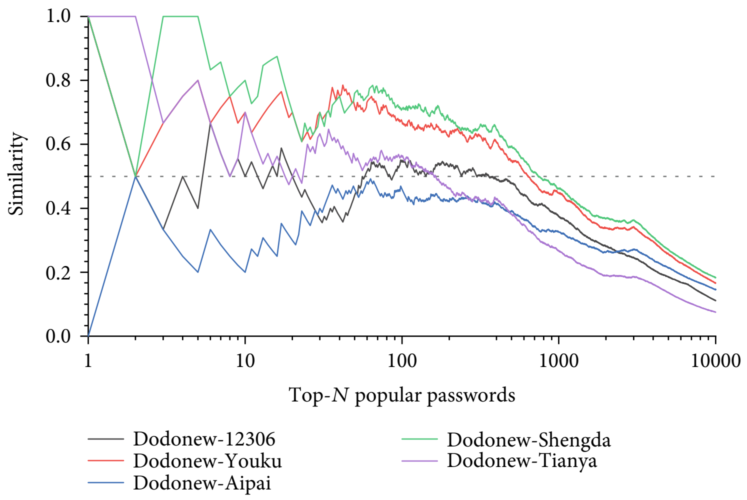
In a real situation, if an attacker is about to attack a targeted user, he/she will do his/her best to obtain the information required for attacking. Therefore, the problem caused by



(a) With **12306**

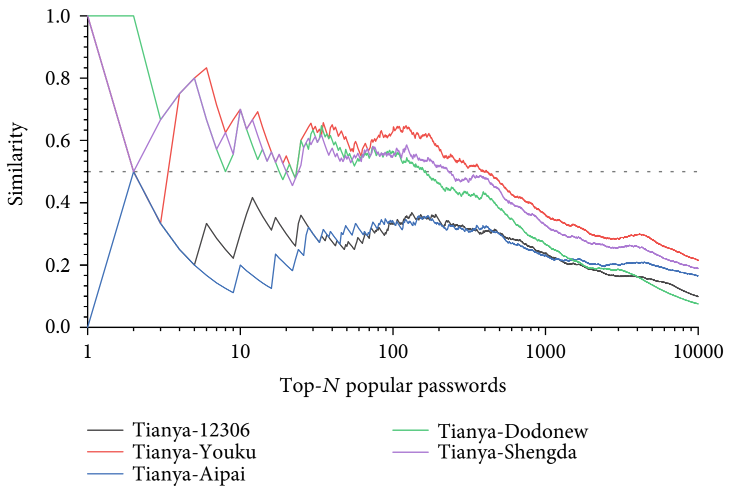


(b) With **Youku**

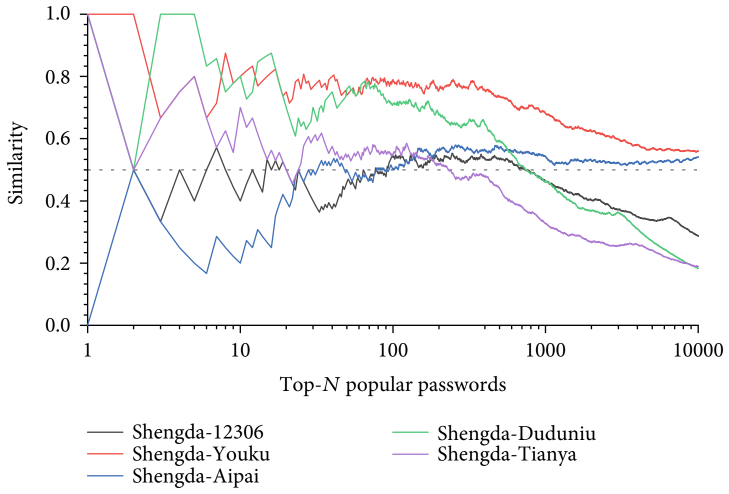


(c) With **Dodon**

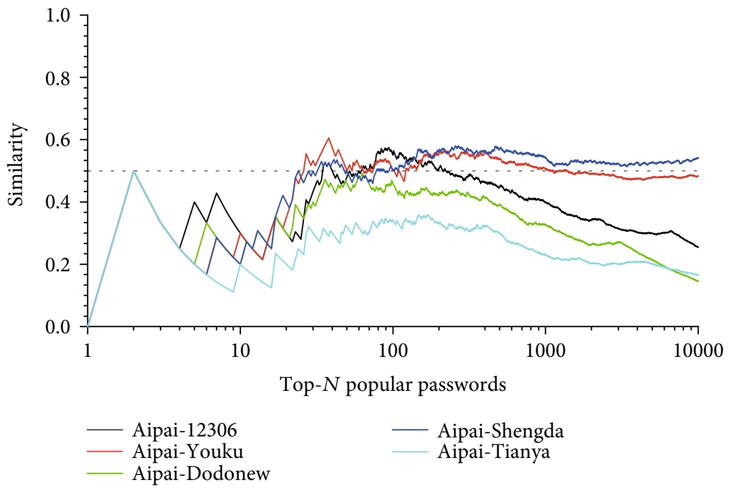
FIGURE 8: Continued.



(d) With **Tinya**



(e) With **Senda**



(f) With **Aipai**

FIGURE 8: Fraction of top-N passwords shared between two services. We use **difflflib** function in Python to calculate the similarity of the top-N popular passwords between each site.

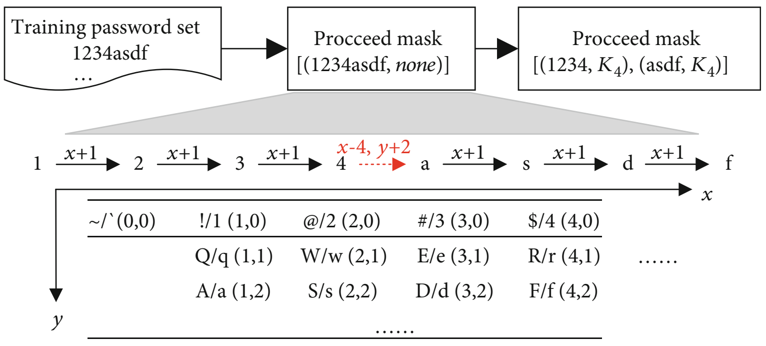


Figure 9: An illustration of K_n tags process.

Table 6: Ranking and proportion of the passwords containing keyboard pattern. *struct*: structure.

Source	Aipai		Tinya		Senda		Dodon		Youku		12306	
Rank	Struct	Rank	Struct	Rank	Struct	Rank	Struct	Rank	Struct	Rank	Struct	Rank
1	K_8	80	K_8	65	K_8	57	K_8	98	K_8	81	K_8	71
2	K_4K_4	148	K_4K_4	150	K_4K_4	141	K_7	184	K_6	143	K_4K_4	152
3	K_{10}	208	K_6	200	K_{10}	170	K_4K_4	190	K_4K_4	165	K_6	175
4	K_9	229	K_{10}	204	K_6	196	K_6	228	K_7	175	K_{10}	216
5	K_5D_3	264	L_4K_4	341	K_9	233	K_9	273	K_{10}	223	K_7	235
6	K_4D_4	327	K_4D_3	386	K_7	309	K_{10}	343	K_9	244	K_9	264
7	D_5K_4	335	K_9	391	K_4D_4	326	K_4D_3	353	K_5D_3	322	K_5D_3	411
8	K_7	360	K_7	424	L_4K_4	401	K_4D_5	423	K_4D_5	388	K_4D_5	493
9	K_6	368	D_5K_4	450	D_5K_4	403	D_5K_4	489	K_4D_4	515	K_4D_4	495
10	K_4L_5	381	D_3K_5	663	K_5D_3	430	K_4D_4	523	L_4K_4	600	L_5K_4	620
%	1.37%		0.97%		1.21%		0.88%		1.19%		1.22%	

reason 1 does not exist. What we do is highlight the threat that incremental information from users will help for targeted password guessing. And for reason 2, we deleted the relative position m with low frequency in the training results, leaving the one with the largest proportion. Thus, we regenerate the implementation of X tag, and the experimental result is shown as the success rate of $+X$ tag (filtered) in Figure 11. We will further study the implementation of X tag in the future.

5. Experiments

TarGuess – I is mainly used in online guessing scenarios, where the guess number allowed is the scarcest resource, while computational power and bandwidth are not essential. Therefore, we mainly evaluate the availability of the modified guessing models by the success rates r with guess number N.

5.1. Experiment Setup. To make our experiments as scientific as possible, we follow 3 rules.

(1) Training sets and testing sets are strictly separated

(2) The comparison experiments of the two models are based on the same training sets and testing sets

(3) The training sets shall be as large as possible

To abide by rule 1 and rule 3, we chose the largest 10^5 sized datasets **12306** and **Youku** as training sets and remove it from the testing list. Particularly, users' passwords in each dataset are highly heterogeneous. The distribution of password semantics may greatly different between two sites even in distinct parts of the same dataset. Thus, the fraction of successfully cracked passwords in each dataset evaluated by different password guessing model may fluctuate greatly. To avoid the heterogeneity of datasets that may hinder our observation of the feasibility of improved methods, we use Monte Carlo method to randomly extract data and generate 10 testing sets from each dataset. The size of each testing set is 10^3. We applied these sets for every evaluation.

Table 7 shows the four-dimensional variables of the experiment setup. We build nine models by adding three improved tags individually or in combination to TarGuess – I (hereinafter referred to as "TG – I"). The four single-

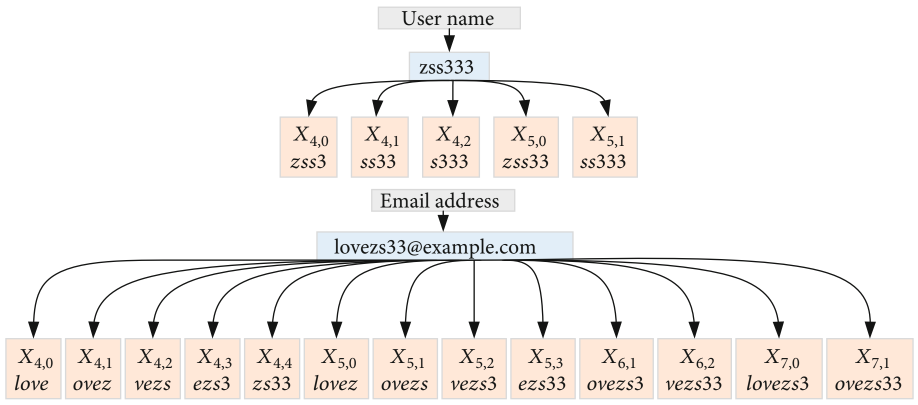


Figure 10: An illustration of X_n tags process.

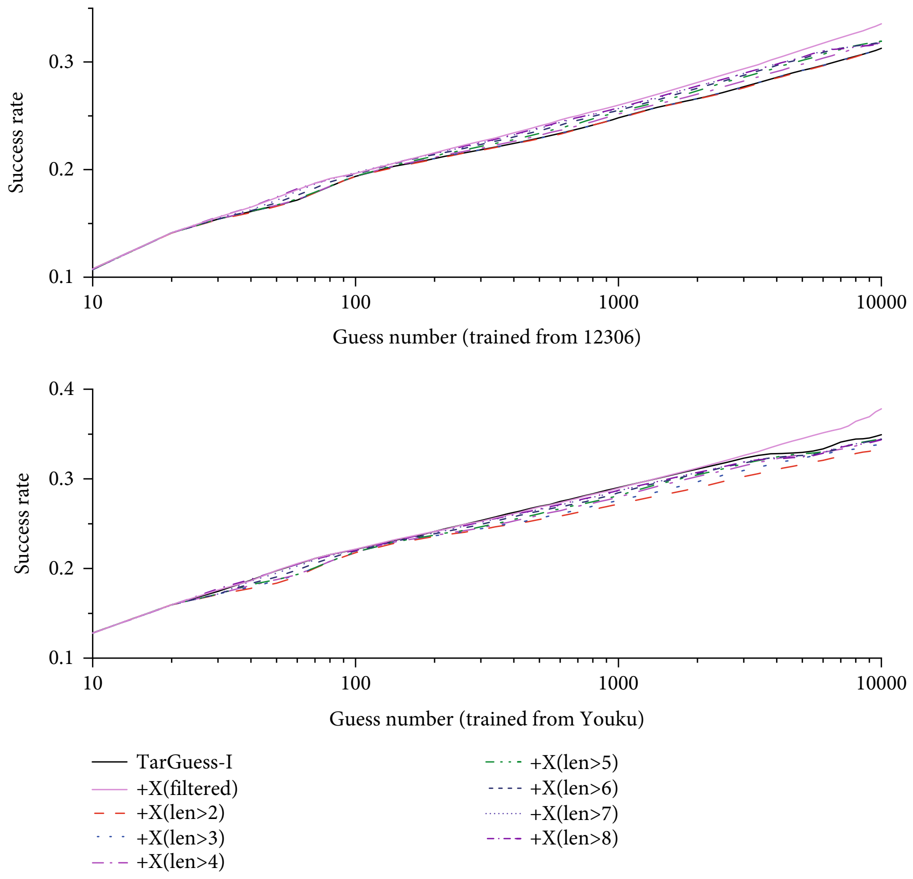


Figure 11: A brief comparison of the length n that considered in X tag.

tag-modified models (e.g., $\mathrm{TG-I}^{+K}$, $\mathrm{TG-I}^{+X}$, $\mathrm{TG-I}^{+P}$, and $\mathrm{TG-I}^{+P'}$) are built to evaluate the validation of our three modified methods (e.g., the popular password tag P, the keyboard pattern tag K, and the special string tag X). To make our evaluations more realistic, we define two kinds of scenarios with P tags. In one optimal scenario (e.g., with P tag), attackers have got the target site's top-300 popular password list for cracking. In the other scenario (e.g., with P' tag), which is more realistic, attackers are only able to crack the target site with similar top-300, from which we choose based on the analysis in Figure 8. The four combined-tag-modified models (e.g., $\mathrm{TG-I}^{+KP}$, $\mathrm{TG-I}^{+PX}$, $\mathrm{TG-I}^{+KX}$, $\mathrm{TG-I}^{+KP'X}$, and $\mathrm{TG-I}^{+KPX}$) are built to find out the optimal model and whether incremental attributes can improve the efficiency of password guessing. We set a total of 80 attacking scenarios based on these four-dimensional variables and conducted 10 experiments on each one.

Figure 12 shows the average of cracking success rate with guess number n evaluated by nine models trained from two sites and tested from four sites. As shown in

Table 7: Training and test settings for each attacking scenario under 9 models.

Password guessing model	$\mathrm{TG-I}^*$, $\mathrm{TG-I}^{+K}$, $\mathrm{TG-I}^{+P}$, $\mathrm{TG-I}^{+P'}$, $\mathrm{TG-I}^{+X}$, $\mathrm{TG-I}^{(+KP')}$, $\mathrm{TG-I}^{(+P'X)}$, $\mathrm{TG-I}^{+KX}$, $\mathrm{TG-I}^{+KPX}$, $\mathrm{TG-I}^{+KP'X}$			
Training sets	**12306, Youku**			
Testing sets	**Aipai**	**Tinya**	**Senda**	**Dodon**
P' tag with top-300[#]	**Senda**'s	**Youku**'s	**Youku**'s	**Senda**'s
P tag with top-300	**Aipai**'s	**Tinya**'s	**Senda**'s	**Dodon**'s

[*]TG-I: TarGuess – I. The superscript of each model represents the improved tags it has added; [#]P' tag is in the optimal condition that attackers have got the target site's top-300 popular password list, while P tag is in the normal condition that attackers only have the similar site's top-300 with target one.

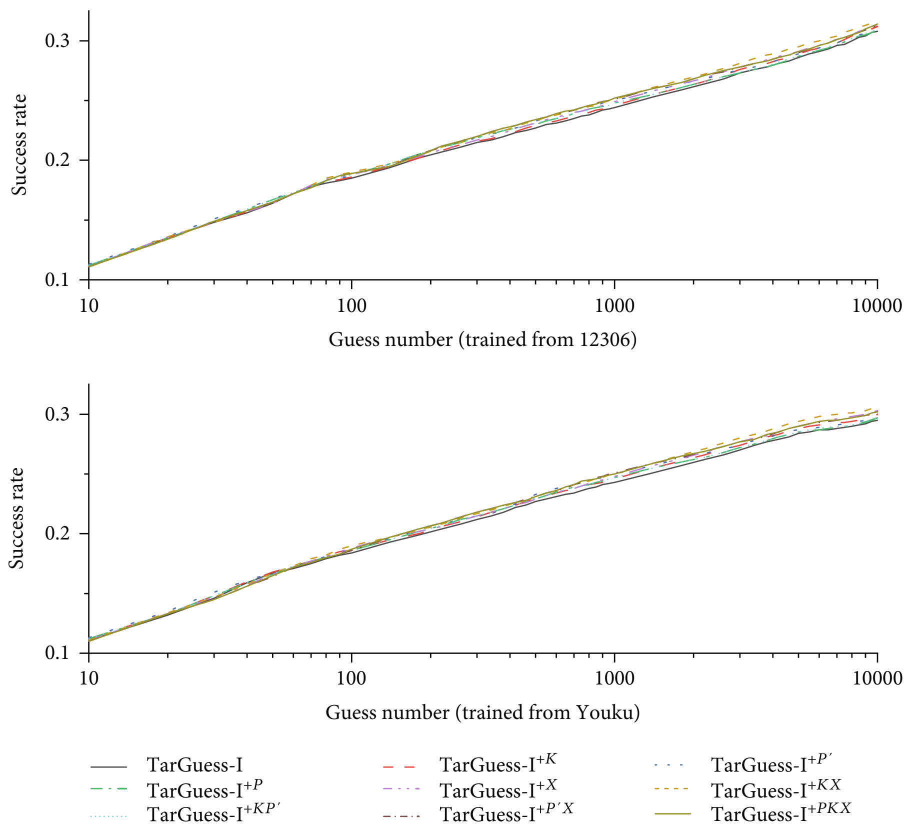


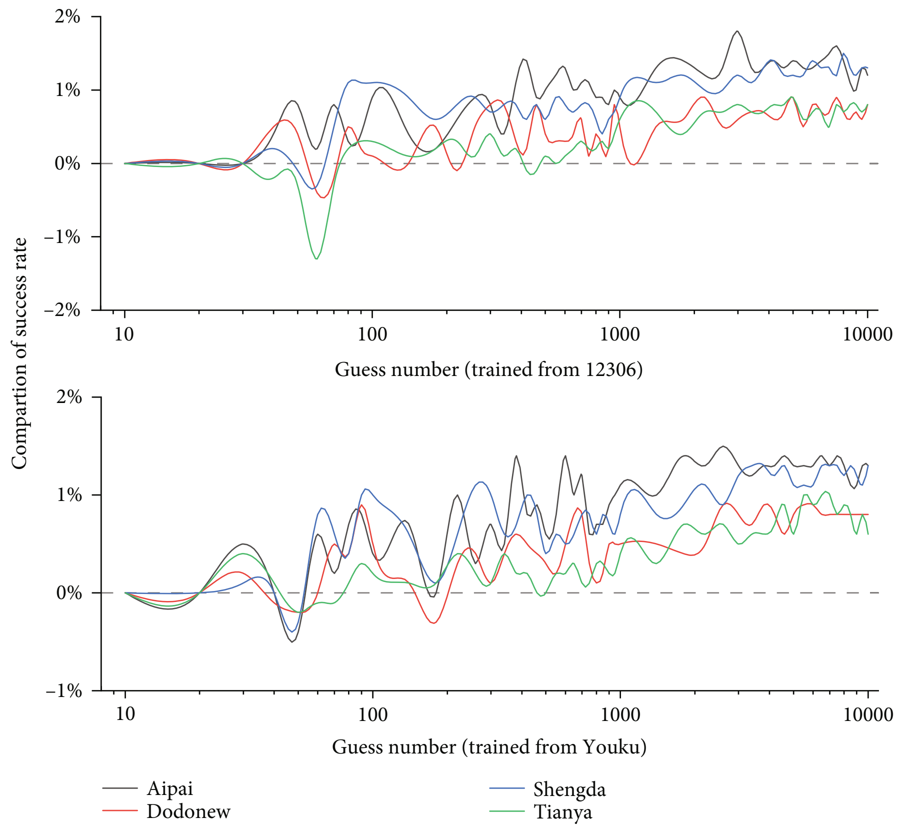
Figure 12: An average guess-number graph including nine models.

the figure, the differences of cracking success rate in each model are not obvious if we just compare them with guess-number-graph like this. Thus, to make the experimental results easier to analyze, we calculate the relative values R_n between each model and original $\mathrm{TG-I}$ with guess number n, which:

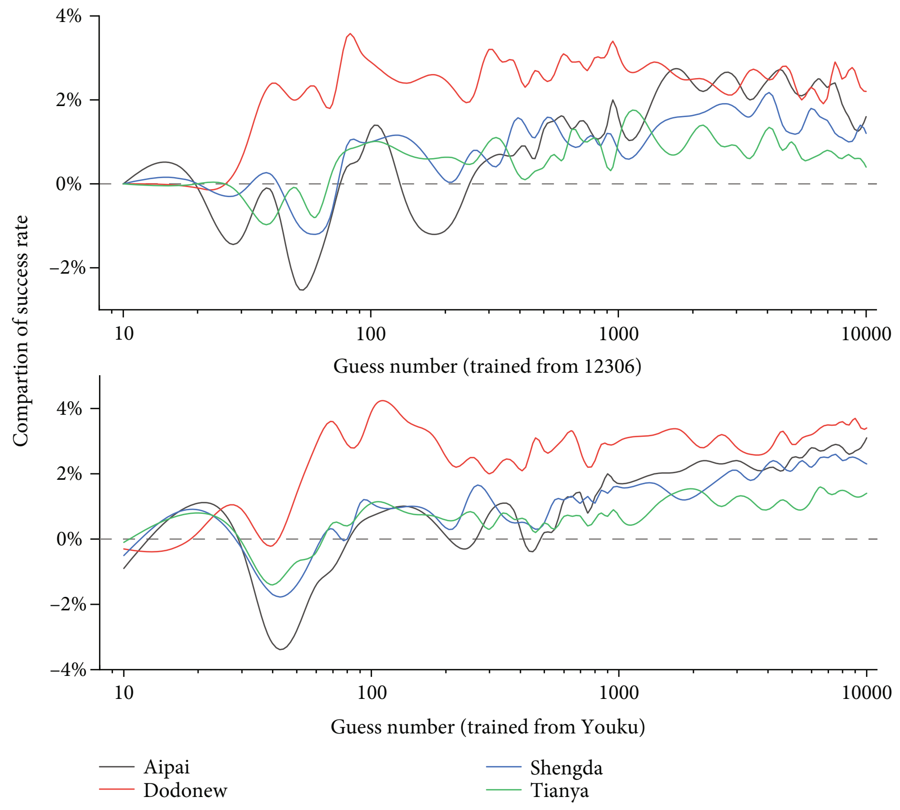
$$R_n = \mathrm{Mean}\left(\frac{{r_n}^{TG_i^+} - {r_n}^{TG_i}}{{r_n}^{TG_i}}\right) \times 100\%, \tag{1}$$

where ${r_n}^{TG_i^+}$ is the success rate of improved model tested from (i)th testing set with guess number n, and ${r_n}^{TG_i}$ is that of TG-I.

5.2. Experiment 1: Validation of the Modified Methods. To demonstrate the effectiveness of our modified methods, we compare the cracking success rate of the four single-tag-modified models (e.g., $\mathrm{TG-I}^{+K}$, $\mathrm{TG-I}^{+X}$, $\mathrm{TG-I}^{+P}$, and $\mathrm{TG-I}^{+P'}$) with that of $\mathrm{TG-I}$ based on the testing data. The R_n-number-graphs of the four single-tag improved models are shown in Figure 13, and the average R_n statistics of them are shown in Table 8. As shown in the Table 8, except that the $\mathrm{TG-I}^{+P'}$ trained by **Youku** has an average R_n of 0.03% lower than $\mathrm{TG-I}$ within 100 guesses, the rest of the single-tag-modified models outperformed $\mathrm{TG-I}$. They outperformed $\mathrm{TG-I}$ by 0.18% ~ 0.75% on average within 100 guesses. It proves the effectiveness of our three modified methods.

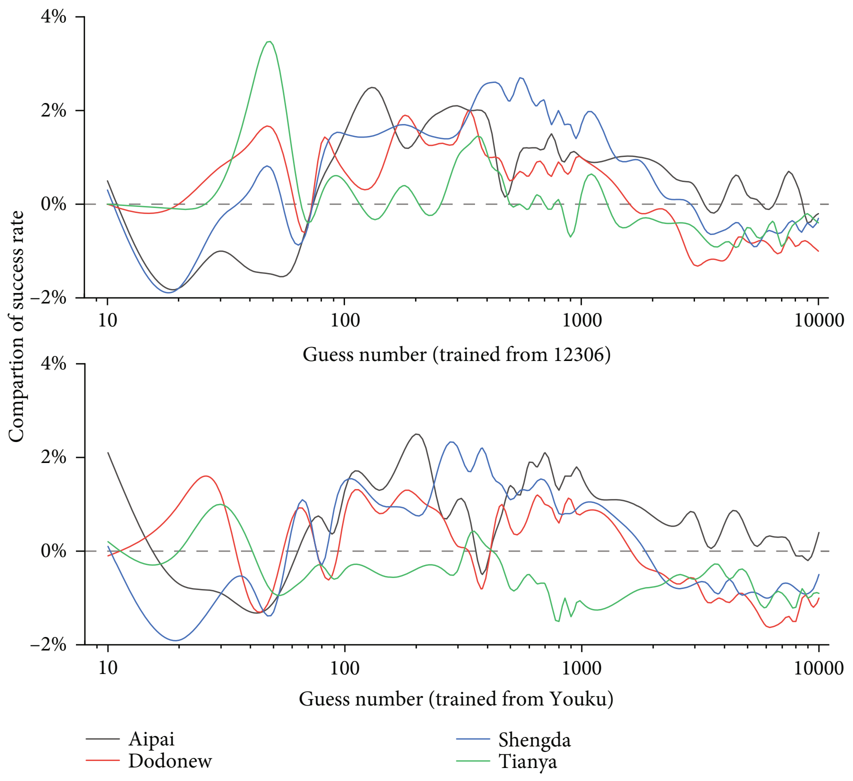


(a) TG – I^{+K}

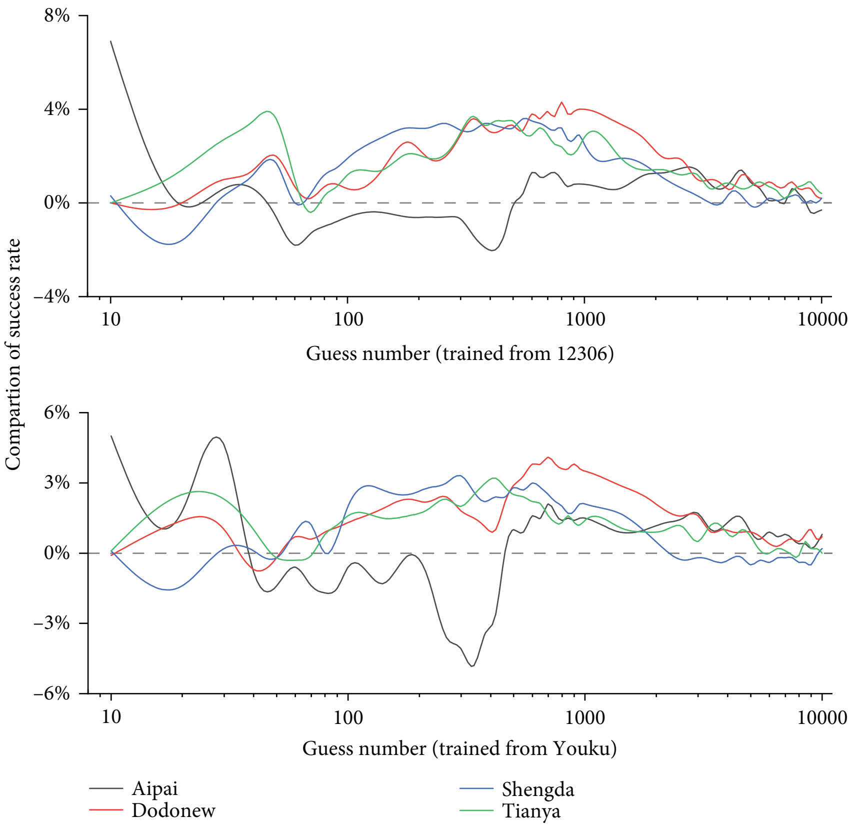


(b) TG – I^{+X}

Figure 13: Continued.



(c) TG − $I^{+P'}$



(d) TG − I^{+P}

FIGURE 13: Experimental results of four single-tag-modified models. Figures 8(a)–8(d) represent the R_n of the four single-tag-modified models. The dotted line at 0% on the y-axis represents our reference baseline (i.e., the cracking success rate of TG-I).

Table 8: Average of R_n statistics of Figure 13.

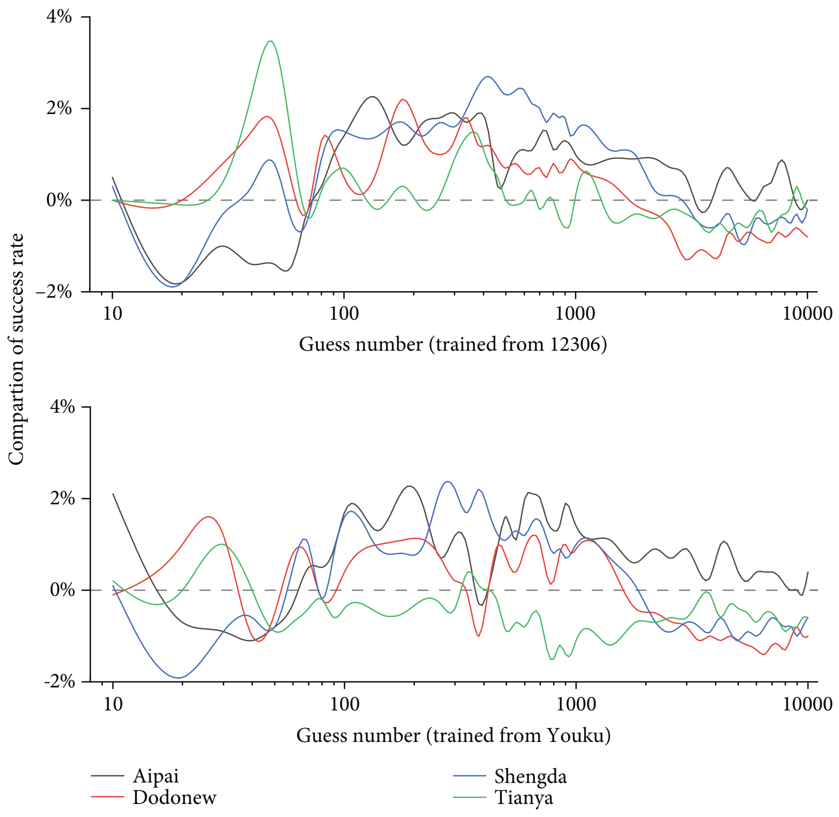
Training Set	Improved Model	Guess number range 10 – 100		100 – 10^3		10^3 – 10^4	
12306	$TG-I^{+K}$	0.61%~-0.43%	0.18%	0.72%~0.35%	0.56%	1.14%~0.88%	0.98%
	$TG-I^{+X}$	1.53%~-0.39%	0.38%	1.74%~0.55%	1.34%	2.11%~1.37%	1.72%
	$TG-I^{+P'}$	1.07%~-0.92%	0.31%	1.80%~0.70%	1.12%	0.50%~-0.70%	-0.27%
	$TG-I^{+P}$	1.79%~-0.31%	0.75%	2.88%~1.31%	2.20%	1.96%~0.11%	0.80%
Youku	$TG-I^{+K}$	0.75%~-0.27%	0.24%	0.72%~-0.02%	0.51%	1.10%~0.66%	0.97%
	$TG-I^{+X}$	1.70%~-1.63%	0.27%	1.80%~0.75%	1.33%	2.62%~1.92%	2.25%
	$TG-I^{+P'}$	0.78%~-1.06%	-0.03%	0.99%~0.28%	0.68%	0.30%~-0.90%	-0.47%
	$TG-I^{+P}$	2.05%~-0.49%	0.55%	2.65%~0.46%	1.73%	1.64%~0.09%	0.67%

Table 9: Statistics of R_n of Figure 14.

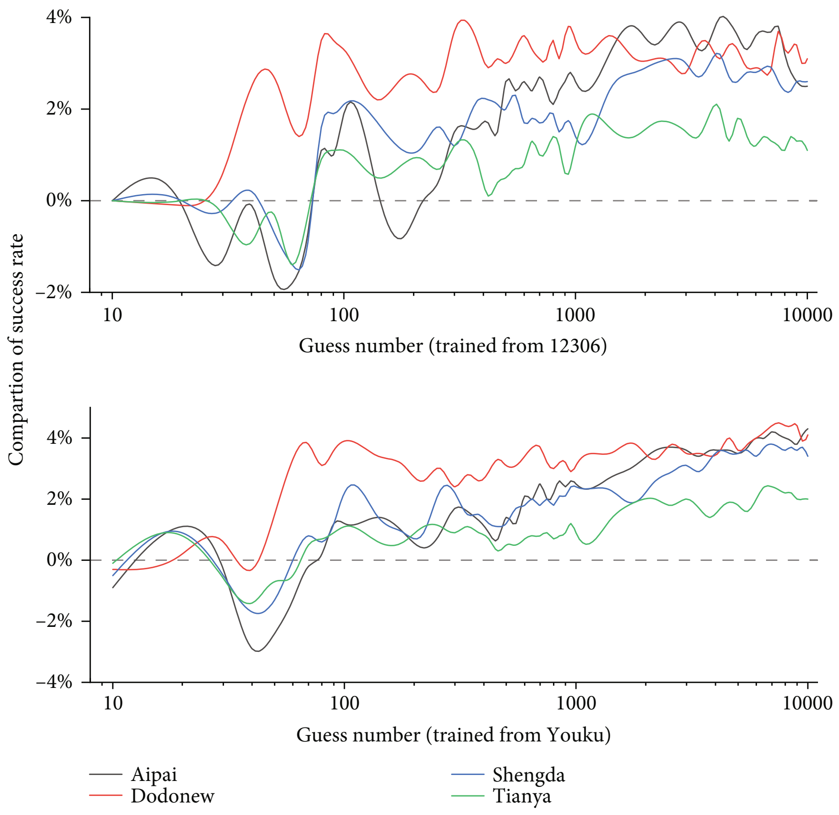
Training set	Improved model	Guess number range 10 – 100		100 – 10^3		10^3 – 10^4	
12306	$TG-I^{+KP}$	1.14%~-0.92%	0.33%	1.75%~0.77%	1.10%	0.51%~-0.62%	-0.19%
	$TG-I^{+PX}$	2.58%~-1.01%	0.59%	3.41%~1.21%	2.70%	2.60%~1.04%	1.56%
	$TG-I^{+KX}$	2.10%~-0.77%	0.50%	2.23%~0.93%	1.90%	3.07%~2.32%	2.67%
	$TG-I^{+KP'X}$	2.60%~-1.01%	0.62%	3.33%~1.20%	2.69%	2.57%~1.15%	1.61%
	$TG-I^{+KPX}$	2.27%~-0.49%	1.05%	4.65%~1.34%	3.67%	3.77%~2.63%	2.73%
Youku	$TG-I^{+KP}$	0.82%~-0.78%	0.04%	1.06%~0.19%	0.65%	0.31%~-0.71%	-0.39%
	$TG-I^{+PX}$	1.37%~-1.90%	0.00%	2.65%~1.69%	2.21%	2.24%~1.38%	1.83%
	$TG-I^{+KX}$	2.11%~-1.55%	0.44%	2.25%~1.24%	1.83%	3.68%~2.44%	3.16%
	$TG-I^{+KP'X}$	1.54%~-1.86%	0.02%	2.67%~1.73%	2.18%	2.25%~1.50%	1.82%
	$TG-I^{+KPX}$	1.85%~-1.13%	0.56%	4.44%~1.96%	3.15%	3.77%~2.63%	2.97%

Figures 13(a) and 13(b) show that, compared to $TG-I$, the guess performances of $TG-I^{+K}$ and $TG-I^{+X}$ models are magnified as the number of guesses increases. In the range of $10 \sim 100$ guesses, our modified models have no better performances than $TG-I$, and the cracking success rates of our models are even lower than $TG-I$ under some guess numbers. For instance, when $TG-I^{+X}$ is evaluated using the **Aipai** datasets, the cracking success rate of $TG-I^{+X}$ is −2.4% (trained from **12306**) lower than $TG-I$ at 50 guesses and −3.2% (trained from **Youku**) lower than $TG-I$ at 40 guesses. This is because the passwords containing K or X tag are relatively small in the overall password distribution, and the addition of the corresponding tag will only affect the lower-ranked candidate passwords. In addition, for $TG-I^{+X}$, it also may be because the implementation of our X tag is still not consistent with the users' behaviors. It causes the modified model to incorrectly generate the candidate passwords with a higher ranking. Nevertheless, as the number of guesses increases, the advantages of our model are reflected. In the guess range of $100 \sim 10^3$, the R_n of our models are slightly increased. And in the guess range of $10^3 \sim 10^4$, the cracking success rates of our models are significantly and stably better than $TG-I$. At the number of 10^4 guesses, compared with $TG-I$, the cracking success rates of $TG-I^{+K}$ are increased by 1.3% at most and 0.6% at least, and the cracking success rates of $TG-I^{+X}$ are increased by 3.4% at most and 0.4% at least. We find that these methods worked for trawling scenarios because it does increase the success rate of the modified models over 100 guesses.

As shown in Figures 13(c) and 13(d), the models with P tag significantly outperform $TG-I$ between 100 and 10^3 guesses. In this range, the cracking success rates of $TG-I^{+P'}$ are at most 2.7% (trained from **12306**) and 2.2% (trained from **Youku**) higher than $TG-I$, and the cracking success rates of $TG-I^{+P}$ are at most 4.3% (trained from **12306**) and 4.1% (trained from **Youku**) higher than $TG-I$. The reason for this is that popular passwords P_1 rank first in the grammar $\mathcal{G}_{\mathcal{J}}$, while the composite-form popular passwords are in the bottom half of the top-300 list. As the above said, composite-form popular passwords cause $TG-I$ to produce invalid output.

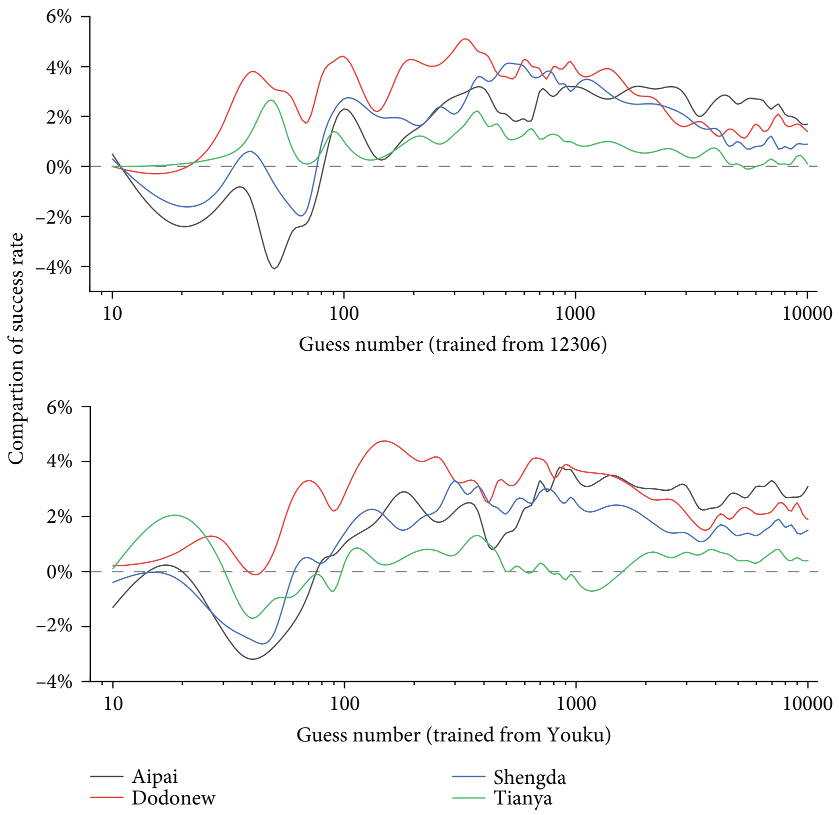


(a) TG − I$^{+KP'}$

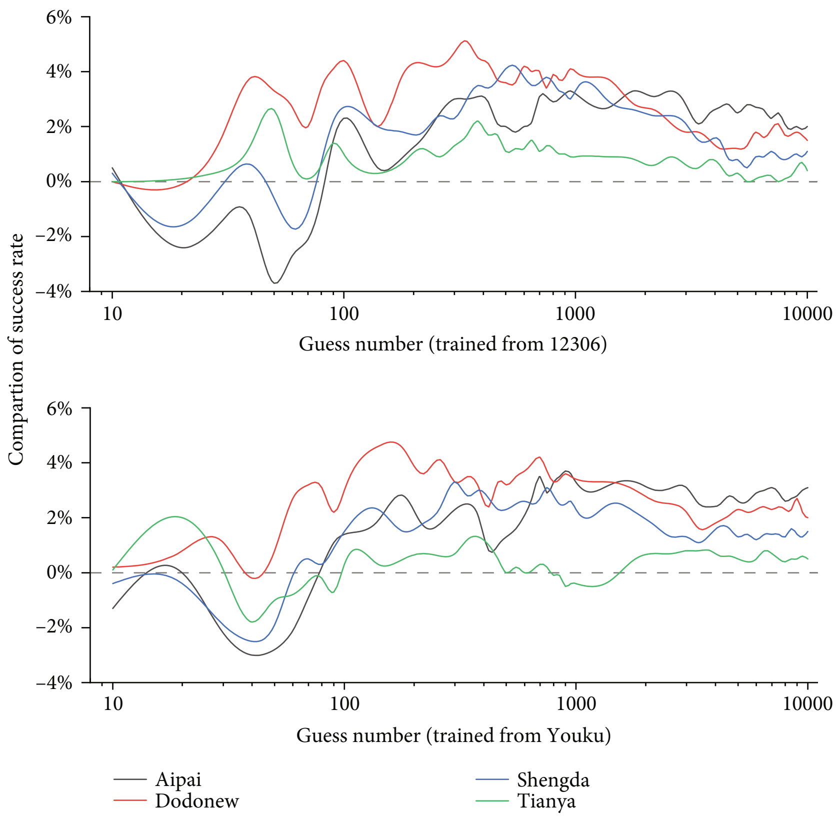


(b) TG − I^{+KX}

FIGURE 14: Continued.

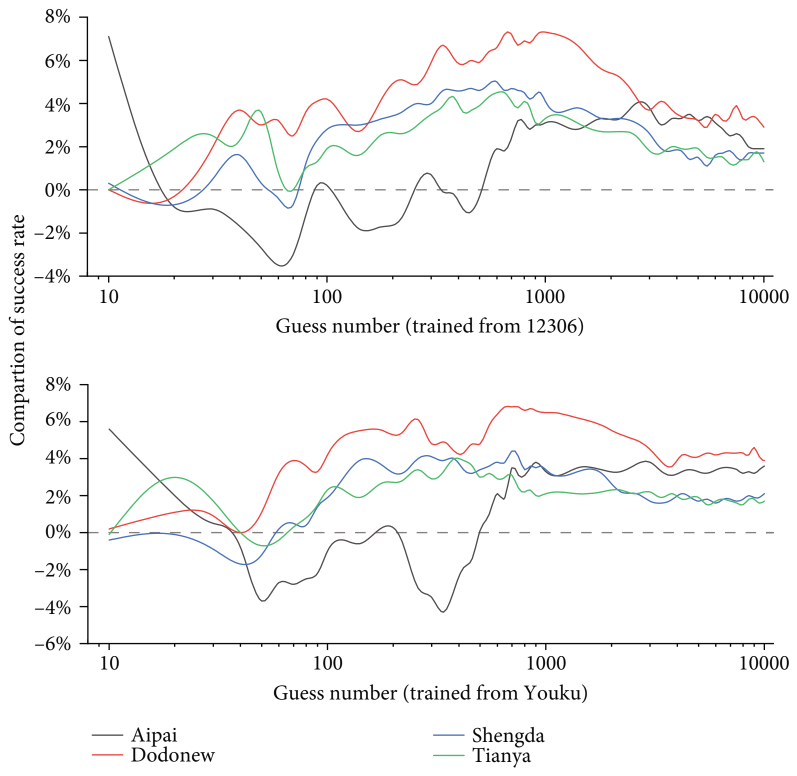


(c) TG − $I^{+P'X}$

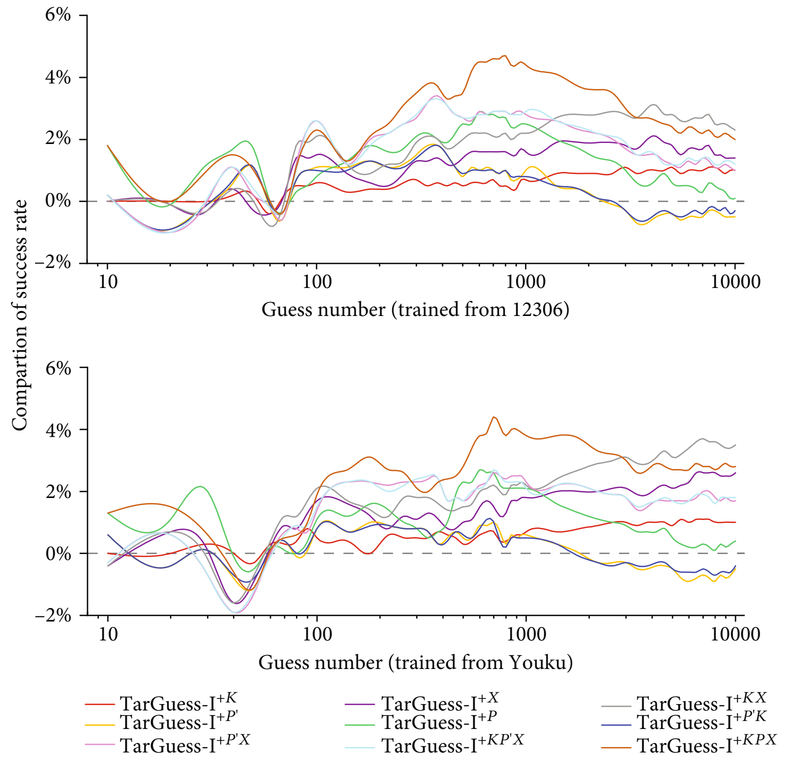


(d) TG − $I^{+KP'X}$

FIGURE 14: Continued.



(e) TG − I^{+KPX}



(f) Avg R_n of the modified models

Figure 14: Experimental results of five combined-tag-modified models and comparison of nine modified models. (a–e) The R_n of the five combined-tag-modified models; (f) the R_n comparison of ten modified models. The dotted line at 0% on the y-axis represents our reference baseline (i.e., the cracking success rate of TG-I).

Table 10: Top-10 base structures of candidate passwords.

Rank	12306				Youku			
	TG – I		TG – I^{+KPX}		TG – I		TG – I^{+KPX}	
1	D_6	4.603%	P_1	4.151%	D_6	7.411%	P_1	6.209%
2	D_7	3.358%	D_6	3.747%	D_7	4.941%	D_6	5.387%
3	N_2D_6	2.519%	D_7	3.112%	D_8	2.498%	D_7	4.519%
4	U_1	1.995%	U_1	1.995%	N_2D_6	2.266%	D_8	2.343%
5	D_8	1.959%	D_8	1.874%	U_1	2.169%	U_1	2.169%
6	E_1	1.776%	N_2D_6	1.845%	U_3	2.039%	U_3	1.985%
7	N_2D_7	1.719%	E_1	1.776%	E_1	1.733%	E_1	1.733%
8	A_2D_6	1.502%	N_2D_7	1.457%	A_2D_6	1.436%	N_2D_6	1.580%
9	U_3	1.466%	U_3	1.431%	B_1	1.350%	B_1	1.347%
10	N_1D_3	1.322%	N_1D_3	1.203%	D_9	1.335%	N_1	1.217%
%*	57.422%		65.516%		53.132%		63.557%	

*The proportions of the password base structures containing incremental tags (such as PII tags, K tag, and P tag) in candidate passwords.

Particularly, TG – I$^{+P'}$ has a lower guess success rate than TG – I within 100 guesses. It is because the tops of the top-300 popular password lists included in the P' tag are different from these in the testing site. As a result, there are a few invalid outputs in the top 100 candidate passwords. It can be seen that there is no such phenomenon in the results of the TG – I^{+P} models, which show that the models also have improvements within 100 guesses.

Interestingly, there is an outlier curve in each R_n-number-graph at Figures 13(b)–13(d). Some curves are significantly higher than others, and some are significantly lower than others. The TG – I^{+X} model has an average guess success rate of 2.6% higher than TG – I on the **Dodon** testing data, while it has an average of 1.0% higher than TG – I on the rest of testing data. The TG – I^{+P} model (trained from **Youku**) shows a −4.8% lower success rate than TG – I on the **Aipai** testing data, while the worst guess performance for other testing data is −1.6% lower than TG – I. This phenomenon may be due to the different distribution of each password dataset. We find that there are a few "uncleaned" passwords data in **Aipai**'s popular password list and datasets, such as "0a2cb03c4dc29cfc0d56afa46ae8fd2e" ranked 20th in the top-300 list. Thus, these "uncleaned" popular password data may cause a reduction in the models' success rate.

5.3. Experiment 2: Comparison and Evaluation of Modified Models. We evaluate each combined-tag-modified model to find out the optimal skim. Table 9 calculates the average R_n compared each modified model with TG – I. TG – I^{+KPX} modified with our three incremental tags has the best improvement effect (see Figure 14(f)).

It can be seen in Figure 14(f) that, by comparing TG – I^{+K}, TG – I^{+X}, and TG – I^{+KX} models, the improvement effects of the modified models are magnified as the number of incremental tags increase. This phenomenon can also be seen by comparing TG – I$^{+P'}$, TG – I$^{+P'X}$, TG – I$^{+KP'}$, and TG – I$^{+KP'X}$ models. However, we also find that the improvement effects of the combined-tag-modified models with P tag are not strongly correlated with the single-tag-modified models with X or K tag. This may because popular passwords account for a large proportion of the password distribution, and there is an obvious gap to the proportion of passwords containing the keyboard patterns or the special strings. Therefore, in terms of the degree of influence on the success rate of guessing, the influence of adding P tag is far greater than that of adding K or X tag.

Table 1 shows the guessing performance of TG – I^{+KPX} evaluated based on each testing dataset. It can be seen that, except that the success rate of TG – I^{+KPX} based on **Aipai** dataset is weaker than that of TG – I within 100 guesses, TG – I^{+KPX} outperforms TG – I based on the other three testing datasets. The reason for the poor results of **Aipai** dataset has been mentioned in the analysis of TG – I$^{+P'}$ in experiment 5.2. TG – I^{+KPX} outperforms TG – I by 0.72% ~ 2.62% (trained from **12306**) and 0.00% ~ 2.16% (trained from **Youku**) within 100 guesses.

Table 10 shows the top-10 base structure of candidate passwords generated by TG – I and TG – I^{+KPX} and the proportions of base structures containing incremental tags in candidate passwords. It can be seen that TG – I^{+KPX} generates nearly 10% more candidate passwords containing incremental tags than TG – I does. Meanwhile, the passwords with the top-10 base structure in Table 10 are very easy to be cracked by the targeted password guessing models. Therefore, we recommend users to avoid setting similar passwords.

In all, the modified methods we proposed are effective. Our results reiterate the threat posed by users using popular passwords and keyboard mode passwords and highlight the threat of targeted password guessing. When an attacker gets more information about a user, the user's password is more likely to be cracked. Our work implies that for important applications, a multifactor authentication scheme (e.g., [40–42]) is necessary.

6. Conclusion

Based on the well-known password guessing model TarGuess – I and six real-world leaked password datasets, we conduct an in-depth analysis of users' vulnerable password setting behavior and targeted password guessing. We find three missing elements in TarGuess – I and propose an improved model: TarGuess – I^{+KPX}, which is capable of identifying popular passwords, keyboard patterns, and the special strings. Experimental results show that our improved model outperforms TarGuess – I by 2.62% within 100 guesses. We highlight the threat posed by targeted password guessing. Our modified idea of the special strings sheds new light on password guessing, but the implementation of this idea is not optimal. We will further study in this direction.

Data Availability

The experimental datasets were disclosed publicly on the Internet (https://breachalarm.com/all-sources). And the probabilistic context-free grammar- (PCFG-) based algorithm code can be found in https://github.com/lakiw/pcfg_cracker.

Disclosure

This article is an extended version of the paper [25].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors are very grateful to the anonymous reviewers for their valuable advice that improves the completeness of this paper. This work was supported by the National Natural Science Foundation of China under grant no. 61971473.

References

[1] J. Bonneau, C. Herley, P. C. Van Oorschot, and F. Stajano, "Passwords and the evolution of imperfect authentication," *Communications of the ACM*, vol. 58, no. 7, pp. 78–87, 2015.

[2] D. Wang, *Research on Key Issues in Password Security, [Ph.D. thesis]*, Peking University, 2017, http://wangdingg.weebly.com/uploads/2/0/3/6/20366987/phd_thesis0103.pdf.

[3] J. Ma, W. Yang, M. Luo, and N. Li, "A study of probabilistic password models," in *2014 IEEE Symposium on Security and Privacy*, pp. 689–704, San Jose, CA, USA, 2014.

[4] A. Narayanan and V. Shmatikov, "Fast dictionary attacks on passwords using time-space tradeoff," in *Proceedings of the 12th ACM conference on Computer and communications security - CCS '05*, pp. 364–372, 2005.

[5] M. Weir, S. Aggarwal, B. De Medeiros, and B. Glodek, "Password cracking using probabilistic context-free grammars," in *2009 30th IEEE Symposium on Security and Privacy*, pp. 391–405, Berkeley, CA, USA, 2009.

[6] R. Veras, C. Collins, and J. Thorpe, "On semantic patterns of passwords and their security impact," in *Proceedings 2014 Network and Distributed System Security Symposium*, San Diego, CA, USA, 2014.

[7] W. Melicher, B. Ur, S. M. Segreti et al., "Fast, lean, and accurate: modeling password guessability using neural networks," in *25th USENIX Security Symposium (USENIX Security 16)*, pp. 175–191, Austin, TX, USA, 2016.

[8] S. Aggarwal, S. Houshmand, and M. Weir, "New technologies in password cracking techniques," in *Cyber Security: Power and Technology*, pp. 179–198, Springer, 2018.

[9] E. Tirado, B. Turpin, C. Beltz, P. Roshon, R. Judge, and K. Gagneja, "A new distributed brute-force password cracking technique," in *Future Network Systems and Security. FNSS 2018*, pp. 117–127, Springer, 2018.

[10] B. Hitaj, P. Gasti, G. Ateniese, and F. Perez-Cruz, "Passgan: a deep learning approach for password guessing," in *Applied Cryptography and Network Security. ACNS 2019*, pp. 217–237, Springer, 2019.

[11] S. Ji, S. Yang, X. Hu, W. Han, Z. Li, and R. Beyah, "Zero-sum password cracking game: a large-scale empirical study on the crackability, correlation, and security of passwords," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 5, pp. 550–564, 2017.

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

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

[14] M. K. Liu Gong-Shen, Q. Wei-Dong, and L. Jian-Hua, "Password vulnerability assessment and recovery based on rules mined from large-scale real data," *Chinese Journal of Computers*, vol. 39, no. 3, pp. 454–467, 2016.

[15] A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang, "The tangled web of password reuse," in *Proceedings 2014 Network and Distributed System Security Symposium*, p. 7, San Diego, CA, USA, 2014.

[16] Y. Li, H. Wang, and K. Sun, "A study of personal information in human-chosen passwords and its security implications.," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9, San Francisco, CA, USA, 2016.

[17] D. Wang, Z. Zhang, P. Wang, J. Yan, and X. Huang, "Targeted online password guessing: an underestimated threat," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1242–1254, Vienna Austria, 2016.

[18] K. C. Wang and M. K. Reiter, "How to end password reuse on the web," in *Proceedings 2019 Network and Distributed System Security Symposium*, San Diego, CA, USA, 2019.

[19] B. Lu, X. Zhang, Z. Ling, Y. Zhang, and Z. Lin, "A measurement study of authentication rate-limiting mechanisms of modern websites," in *Proceedings of the 34th Annual Computer Security Applications Conference*, pp. 89–100, San Juan, PR, USA, 2018.

[20] B. Pal, T. Daniel, R. Chatterjee, and T. Ristenpart, "Beyond credential stuffing: password similarity models using neural networks," in *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 417–434, San Francisco, CA, USA, 2019.

[21] P. A. Grassi, J. L. Fenton, E. Newton et al.et al., "Nist special publication 800-63b: digital identity guidelines," *Enrollment*

and Identity Proofing Requirements, 2017, https://pages.nist.gov/800-63-3/sp800-63b.html.

[22] A. D. Jaggard and P. Syverson, "Oft target," in *Proceedings of the PET*, Barcelona, Spain, 2018.

[23] M. Guri, E. Shemer, D. Shirtz, and Y. Elovici, "Personal information leakage during password recovery of internet services," in *2016 European Intelligence and Security Informatics Conference (EISIC)*, pp. 136–139, Uppsala, Sweden, 2016.

[24] 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*, pp. 196–203, New York, NY, USA, 2018.

[25] Z. Xie, M. Zhang, A. Yin, and Z. Li, "A new targeted password guessing model," in *Information Security and Privacy. ACISP 2020*, pp. 350–368, Springer, 2020.

[26] A. Adams and M. A. Sasse, "Users are not the enemy," *Communications of the ACM*, vol. 42, no. 12, pp. 40–46, 1999.

[27] R. Morris and K. Thompson, "Password security," *Communications of the ACM*, vol. 22, no. 11, pp. 594–597, 1979.

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

[29] M. L. Mazurek, S. Komanduri, T. Vidas et al., "Measuring password guessability for an entire university," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security - CCS '13*, pp. 173–186, Berlin Germany, 2013.

[30] D. V. Bailey, M. Durmuth, and C. Paar, "Statistics on password re-use and adaptive strength for financial accounts," in *Security and Cryptography for Networks. SCN 2014*, pp. 218–235, Springer, 2014.

[31] E. I. Tatlı, "Cracking more password hashes with patterns," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 8, pp. 1656–1665, 2015.

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

[33] P. G. Kelley, S. Komanduri, M. L. Mazurek et al., "Guess again (and again and again): measuring password strength by simulating password-cracking algorithms," in *2012 IEEE Symposium on Security and Privacy*, pp. 523–537, San Francisco, CA, USA, 2012.

[34] B. Ur, F. Noma, J. Bees et al., ""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)*, pp. 123–140, Ottawa, Canada, 2015.

[35] R. Shay, L. Bauer, N. Christin et al., "A spoonful of sugar? The impact of guidance and feedback on passwordcreation behavior," in *2015 in Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*, pp. 2903–2912, Seoul Republic of Korea, 2015.

[36] D. Wang, H. Cheng, P. Wang, X. Huang, and G. Jian, "Zipf's law in passwords," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2776–2791, 2017.

[37] D. Wang, P. Wang, D. He, and Y. Tian, "Birthday, name and bifacial-security: understanding passwords of chinese web users," in *28th USENIX Security Symposium (USENIX Security 19)*, pp. 1537–1555, Santa Clara, CA, USA, 2019.

[38] D. Wang, H. Cheng, Q. Gu, and P. Wang, "Understanding passwords of Chinese users: characteristics, security and implications," in *CACR Report, Presented at ChinaCrypt*, Shanghai, China, 2015.

[39] D. Wang, D. He, H. Cheng, and P. Wang, "fuzzyPSM: a new password strength meter using fuzzy probabilistic context-free grammars," in *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 595–606, Toulouse, France, 2016.

[40] C. Wang, D. Wang, Y. Tu, G. Xu, and H. Wang, "Understanding node capture attacks in user authentication schemes for wireless sensor networks," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[41] Q. Jiang, N. Zhang, J. Ni, J. Ma, X. Ma, and K. K. R. Choo, "Unified biometric privacy preserving threefactor authentication and key agreement for cloudassisted autonomous vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 9390–9401, 2020.

[42] D. Wang, W. Li, and P. Wang, "Measuring two-factor authentication schemes for real-time data access in industrial wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 9, pp. 4081–4092, 2018.