Personal Information in Passwords and Its Security Implications

Yue Li, Haining Wang, and Kun Sun

Abstract—**While it is not recommended, Internet users tend to include personal information in their passwords for easy memorization. However, the use of personal information in passwords and its security implications have yet to be studied. In this paper, we dissect user passwords from several leaked data sets to investigate the extent to which a user's personal information resides in a password. Then, we introduce a new metric called coverage to quantify the correlation between passwords and personal information. Afterward, based on our analysis, we extend the probabilistic context-free grammars (PCFGs) method to be semantics-rich and propose personal-PCFG to crack passwords by generating personalized guesses. Through offline and online attack scenarios, we demonstrate that personal-PCFG cracks passwords much faster than PCFG and makes online attacks much more likely to succeed. To defend against such semantics-aware attacks, we examine the use of simple distortion functions that are chosen by users to mitigate unwanted correlation between personal information and passwords.**

Index Terms—**Password measurement, password attack and protection, personal information.**

I. Introduction

The text-based password is still believed to remain a dominating and irreplaceable authentication method in the foreseeable future. Although researchers have proposed different authentication mechanisms, no alternative can bring all the benefits of passwords without introducing extra burdens to users [3]. However, passwords have long been criticized as being one of the weakest links in authentication. Due to the human memorability limitation, user passwords are usually far from truly random [2], [28], [33], [42], [46]. For instance, "secret" is more likely a human-chosen password than "zjorqpe. In other words, human users are prone to choose weak passwords simply because they are easier to remember. As a result, most passwords are chosen within a small portion of the entire password space, leaving them vulnerable to brute-force or dictionary attacks.

To increase password security, online authentication systems have started to enforce stricter password policies. Meanwhile, many websites provide password strength meters to help users create secure passwords. However, these meters are proven to be ad-hoc and inconsistent [13], [14]. To better assess the strength of passwords, one needs to have a deeper understanding of how users construct their passwords. Knowing the exact tactics to create passwords also helps an attacker to crack passwords. Meanwhile, if a user is aware of the potential vulnerability induced by a commonly used password creation method, the user can avoid using the same method for creating passwords.

Toward this end, researchers have made significant efforts to unveil the structures of passwords. Traditional dictionary attacks on passwords have shown that users tend to use simple dictionary words to construct their passwords [17], [32]. The first language of a user is also preferred when constructing passwords [2]. Besides, passwords are commonly phonetically memorable [33] even though they are not simple dictionary words. It is also indicated that users may use keyboard strings such as "qwerty" and "qweasdzxc," trivial strings such as "password" and "123456," and date strings such as "19951225" in their passwords [27], [39], [42]. However, most studies reveal only superficial password patterns, and the semantic-rich composition of passwords remains to be fully uncovered. Fortunately, an enlightening work investigates how users generate their passwords by learning the semantic patterns [41].

This paper studies password semantics from a different perspective: the use of personal information. We utilize a leaked password dataset, which contains personal information, from a Chinese website for this study. We first measure the usage of personal information in password creation and present interesting observations. We are able to obtain the most popular password structures with personal information embedded. It is also observed that males and females behave differently when using personal information in password creation. In addition, we assess the usage of names in other leaked password datasets that do not come with any personal information.

Next, we introduce a new metric called Coverage to accurately quantify the correlation between personal information and user passwords. Since it considers both the length and continuation of personal information, Coverage is a useful metric to measure the strength of a password. Our quantification results using the Coverage metric confirm our direct measurement results on the dataset, showing the efficacy of Coverage. Moreover, Coverage is easy to integrate with

Manuscript received November 8, 2016; revised February 23, 2017 and April 23, 2017; accepted May 1, 2017. Date of publication May 18, 2017; date of current version June 23, 2017. A preliminary version of this paper was presented at the IEEE INFOCOM 2016 [26]. This work was supported in part by U.S. ONR under Grant N00014-16-1-3216 and Grant N00014-16-1-3214. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Karen Renaud. *(Corresponding author: Yue Li.)*

Y. Li is with the College of William & Mary, Williamsburg, VA 23185 USA (e-mail: yli@cs.wm.edu)

H. Wang is with the University of Delaware, Newark, DE 19716 USA (e-mail: hnw@udel.edu)

K. Sun is with George Mason University, Fairfax, VA 22030 USA (e-mail: ksun3@gmu.edu)

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2017.2705627

existing tools, such as password strength meters. To demonstrate the security vulnerability induced by using personal information in passwords, we propose a semantics-rich Probabilistic Context-Free Grammars (PCFG) method called Personal-PCFG, which extends PCFG [45] by considering personal information symbols in password structures. Personal-PCFG takes a user's personal information as another input vector and generates highly personalized guesses for a specific account. With the assistance of such knowledge, Personal-PCFG is able to crack passwords much faster than original PCFG. It also makes an online attack more feasible.

Finally, we present simple distortion functions to defend against these semantics-aware attacks such as Personal-PCFG or [41]. Our evaluation results demonstrate that distortion functions can effectively protect passwords by significantly reducing the unwanted correlation between personal information and passwords.

Though our study is primarily based on a dataset collected from a Chinese website, we also try to extend our measurement study to English-based websites. We conclude that as long as memorability plays an important role in password creation, the correlation between personal information and user passwords remains, regardless of the language. We believe that our work on personal information quantification, password cracking, and password protection could be applicable to any other text-based password datasets from different websites.

The remainder of this paper is organized as follows. Section II measures how personal information resides in user passwords and shows the gender difference in password creation, based on a leaked dataset from an online ticket reservation website. Section III extends analysis to more datasets. Section IV introduces the new metric, Coverage, to accurately quantify the correlation between personal information and user passwords. Section V details Personal-PCFG and compares cracking results with the original PCFG. Section VI presents a simple but effective defense against semantics-aware attacks by applying distortion functions on password creation. Section VII discusses the limitation of this work. Section VIII surveys related work, and finally Section IX concludes this paper.

II. Personal Information in Passwords

Intuitively, people tend to create passwords based on their personal information because human beings are limited by their memory capacities. We show that personal information plays an important role in a human-chosen password by dissecting passwords in a mid-sized password dataset. Understanding the usage of personal information in passwords and its security implications can help us further enhance password security. To start, we introduce the dataset used throughout this study.

A. 12306 Dataset

A number of password datasets have been exposed to the public in recent years, usually containing several thousand to millions of real passwords. As a result, there are several password-cracking studies based on analyzing these datasets [2], [27]. In this paper, a dataset called 12306 is used to illustrate how personal information is involved in password creation.

TABLE I
Most Frequent Passwords

Rank	Password	Amount	Percentage
1	123456	389	0.296%
2	a123456	280	0.213%
3	123456a	165	0.125%
4	5201314	160	0.121%
5	111111	156	0.118%
6	woaini1314	134	0.101%
7	qq123456	98	0.074%
8	123123	97	0.073%
9	000000	96	0.073%
10	1qaz2wsx	92	0.070%

1) Introduction to Dataset: At the end of 2014, a Chinese dataset was leaked to the public by anonymous attackers. It is reported that the dataset was collected by trying passwords from other leaked passwords in an online attack. hereafter, it is called 12306 dataset because all passwords are from a website www.12306.cn, which is the official site of the online railway ticket booking system in China. There is no data available on the exact number of users of the 12306 website; however, we infer at least tens of millions of registered users in the system, since it is the official website for the entire Chinese railway system.

The 12306 dataset contains more than 130,000 Chinese passwords. Having witnessed so many leaked large datasets, its size is considered medium. What makes it special is that together with plaintext passwords, the dataset also includes several types of personal information, such as a user's name and the government-issued unique ID number (similar to the U.S. Social Security Number). As the website requires a real ID number to register and people must provide accurate personal information to book a ticket, the information in this dataset is considered reliable.

2) Basic Analysis: We first conduct a simple analysis to reveal general characteristics of the 12306 dataset. For data consistency, users whose ID number is not 18-digit long are removed. These users may have used other forms of ID (e.g., passport number) for registration and account for 0.2% of the dataset size. The dataset contains 131,389 passwords for analysis after being cleansed. Note that different websites may have different password creation policies. From the leaked dataset, we can infer that the password policy is quite simple—all passwords cannot be shorter than six characters. Besides, there is no restriction on types of characters used.

The average length of passwords in the dataset is 8.44. The most common passwords are listed in Table I. The dominating passwords are trivial passwords (e.g., 123456, a123456, etc.), keyboard passwords (e.g., 1qaz2wsx, 1q2w3e4r, etc.), and phrasal passwords (e.g., "I love you). Both "5201314" and "woaini1314" mean "I love you forever" in Chinese. The most

TABLE II

RESISTANCE TO GUESSING

H_∞	$\tilde{G}$	λ_5	λ_{10}	$\tilde{G}_{0.25}$	$\tilde{G}_{0.5}$
8.4	16.85	0.25%	0.44%	16.65	16.8

TABLE III

MOST FREQUENT PASSWORD STRUCTURES

Rank	Structure	Amount	Percentage
1	D_7	10,893	8.290%
2	D_8	9,442	7.186%
3	D_6	9,084	6.913%
4	L_2D_7	5,065	3.854%
5	L_3D_6	4,820	3.668%
6	L_1D_7	4,770	3.630%
7	L_2D_6	4,261	3.243%
8	L_3D_7	3,883	2.955%
9	D_9	3,590	2.732%
10	L_2D_8	3,362	2.558%

"D" represents digits and "L" represents English letters. The number indicates the segment length. For example, L_2D_7 means the password contains 2 letters followed by 7 digits.

commonly used passwords in 12306 dataset are similar to those are found in a previous study [27]; however, popular 12306 passwords distribute more sparsely in the password space. The most popular password, "123456," accounts for less than 0.3% of all passwords while the number being 2.17% in [27]. The password sparsity may be due to the importance of the website service nature, where users are less prone to use very trivial passwords like "123456" and there are fewer Sybil accounts as a real ID number is mandatory for registration.

Similar to [27], the resistance to guessing of the 12306 dataset is measured in terms of several useful metrics, including the worst-case security bit representation (H_∞), the guesswork bit representation ($\tilde{G}$), the α-guesswork bit representations ($\tilde{G}_{0.25}$ and $\tilde{G}_{0.5}$), and the β-success rates (λ_5 and λ_{10}). The results are listed in Table II, showing that 12306 has a substantially higher worst-case security and β-success rates than the previously studied datasets. This is mainly because the users of 12306 avoid using extremely guessable passwords such as "123456." It implies that users have certain password security concerns when creating passwords for critical services like 12306. However, their security concern is limited to the avoidance of using extremely guessable passwords. As indicated by the values of α-guesswork, the overall password sparsity of the 12306 dataset is not higher than that of the previously studied datasets.

We also study the composition structures of passwords in 12306. The most popular password structures are listed in Table III. Similar to a previous study [27], our results again show that Chinese users prefer digits in their passwords as opposed to letters that are favored by English-speaking users. Specifically, the top five structures all have a significant portion of digits. The reason behind this may be that Chinese characters are logogram-based, and digits seem to be the best alternative when creating a password.

TABLE IV

PERSONAL INFORMATION

Type	Description
Name	User's Chinese name
Email address	User's registered email address
Cell phone	User's registered cell phone number
Account name	The username used to log in to the system
ID number	Government-issued ID number

In summary, the 12306 dataset is a Chinese password dataset that has general Chinese password characteristics. Users have certain level of security concern by choosing less trivial passwords. However, in terms of the overall sparsity, the 12306 dataset is no higher than previously studied datasets.

B. Personal Information

Other than passwords, 12306 dataset also includes multiple types of personal information, as listed in Table IV.

Note that the government-issued ID number is a unique 18-digit number, which intrinsically includes the owner's personal information. Specifically, digits 1-6 represent the birthplace, digits 7-14 represent the birthdate, and digit 17 represents the gender—odd being male and even being female. We take out the 8-digit birthdate and treat it separately since birthdate is a critical piece of personal information in password creation. Thereby, six types of personal information are considered: name, birthdate, email address, cell phone number, account name, and ID number (birthdate excluded).

1) New Password Representation: To better illustrate how personal information correlates to user passwords, we develop a new representation of a password by adding more semantic symbols besides the conventional "D," "L," and "S" symbols, which stand for digit, letter, and special character, respectively.

The password is first matched to the six types of personal information under this new representation. For example, a password "alice1987abc" can be represented as $[Name][Birthdate]L_3$, instead of $L_3D_4L_3$ as in the traditional representation. The matched personal information is denoted by corresponding tags—[Name] and [Birthdate] in this example; for segments that are not matched, we still use "D," "L," and "S" to describe the symbol types.

Representations like $[Name][Birthdate]L_3$ are more accurate than $L_5D_4L_3$ in describing the composition of a user password by including more detailed semantic information. Using this representation, the following matching method is applied to the entire 12306 dataset to uncover the way personal information appears in password structures.

2) Matching Method: We propose a matching method to locate personal information in a user password, which is shown in Algorithm 1. The high-level idea is that we first generate all substrings of the password and sort them in descending-length order. Then we match these substrings, from the longest to the shortest, to all types of personal information. If a match is found, the match function is recursively applied over the remaining password segments until no further matches are found. The segments that are not matched to any personal information will still be labeled using the traditional "LDS" tags.

Algorithm 1 Personal Information Matching

```
 1: procedure MATCH($pwd$,$infolist$)
 2:     $newform \leftarrow$ empty_string
 3:     if len($pwd$) == 0 then
 4:         return empty_string
 5:     end if
 6:     $substring \leftarrow$ get_all_substring($pwd$)
 7:     reverse_length_sort($substring$)
 8:     for $eachstring \in substring$ do
 9:         if len($eachstring$) $\geq 2$ then
10:             if matchbd($eachstring$,$infolist$) then
11:                 $tag \leftarrow$ "[BD]"
12:                 $leftover \leftarrow pwd$.split($eachstring$)
13:                 break
14:             end if
15:             ...
16:             if matchID($eachstring$,$infolist$) then
17:                 if tag != None then
18:                     $tag \leftarrow tag +$ "&[$ID$]"
19:                 else
20:                     $tag \leftarrow$ "[ID]"
21:                 end if
22:                 $leftover \leftarrow pwd$.split($eachstring$)
23:                 break
24:             end if
25:         else
26:             break
27:         end if
28:     end for
29:     if $leftover$.size() $\geq 2$ then
30:         for i $\leftarrow$ 0 to $leftover$.size()-2 do
31:             $newform \leftarrow$ MATCH($leftover[i]$,$infolist$) + $tag$
32:         end for
33:         $newform \leftarrow$ MATCH($leftover[leftover.size() - 1]$)+$newform$
34:     else
35:         $newform \leftarrow$ seg($pwd$)
36:     end if
37:     $results \leftarrow$extract_ambiguous_structures($newform$)
38:     return $results$
39: end procedure
```

In Algorithm 1, we first ensure that the length of a password segment is at least 2. Then we try to match eligible segments to each kind of the personal information (line 10 and line 16). Note that personal information may not always appear as it is. Instead people sometimes may mangle them a bit or use abbreviations. As each case is different, we do not present the specific algorithms used for each type of the personal information. Instead, we describe the methods as follows. For the Chinese names, we convert them into Pinyin form, which is the alphabetic representation of Chinese characters. Then we compare password segments to 10 possible permutations of a name, such as *lastname+firstname* and *last_initial+firstname*. If the segment is exactly the same as one of the permutations, we consider it a match. For birthdate, we list 17 possible permutations, such as YYYYMMDD, and compare them with a password segment. If the segment is the same as any permutation, we consider it a match. For account name, email address, cell phone number, and ID number, we further constrain the length of a segment to be at least 3 to avoid mismatching by coincidence.

TABLE V
MOST FREQUENT PASSWORD STRUCTURES

Rank	Structure	Amount	Percentage
1	[ACCT]	6,820	5.190%
2	D7	6,224	4.737%
3	[NAME][BD]	5,410	4.117%
4	[BD]	4,470	3.402%
5	D6	4,326	3.292%
6	[EMAIL]	3,807	2.897%
7	D8	3,745	2.850%
8	L1D7	2,829	2.153%
9	[NAME]D7	2,504	1.905%
10	[ACCT][BD]	2,191	1.667%

TABLE VI
PERSONAL INFORMATION USAGE

Rank	Information Type	Amount	Percentage
1	Birthdate	31,674	24.10%
2	Account Name	31,017	23.60%
3	Name	29,377	22.35%
4	Email	16,642	12.66%
5	ID Number	3,937	2.996%
6	Cell Phone	3,582	2.726%

Note that for one password segment, it may result in matches of multiple types of personal information. In such cases, all matches are counted. Thus, the results of Algorithm 1 contain all possible matches.

3) Matching Results: After applying Algorithm 1 to 12306 dataset, we find that 78,975 out of 131,389 (60.1%) of the passwords contain at least one of the six types of personal information. Apparently, personal information is frequently used in password creation. The ratio could be even higher if we know more personal information about users. We present the top 10 password structures in Table V and the usage of personal information in passwords in Table VI. As mentioned above, a password segment may match multiple types of personal information, and we count all of these matches. Therefore, the sum of the percentages is greater than 60.1%. Within 131,389 passwords, we obtain 1,895 password structures. Based on Tables V and VI, we can see that people largely rely on personal information when creating passwords. Among the 6 types of personal information, birthdate, account name, and name are most popular with a more than 20% occurrence rate, and 12.66% of users include email in their passwords. However, only small

TABLE VII
MOST FREQUENT STRUCTURES IN DIFFERENT GENDERS

Rank	Male		Female	
	Structure	Percentage	Structure	Percentage
1	[ACCT]	4.647%	D6	3.909%
2	D7	4.325%	[ACCT]	3.729%
3	[NAME][BD]	3.594%	D7	3.172%
4	[BD]	3.080%	D8	2.453%
5	D6	2.645%	[EMAIL]	2.372%
6	[EMAIL]	2.541%	[NAME][BD]	2.309%
7	D8	2.158%	[BD]	1.968%
8	L1D7	2.088%	L2D6	1.518%
9	[NAME]D7	1.749%	L1D7	1.267%
10	[ACCT][BD]	1.557%	L2D7	1.240%
NA	TOTAL	28.384%	TOTAL	23.937%

TABLE VIII
MOST FREQUENT PERSONAL INFORMATION IN DIFFERENT GENDERS

Rank	Male		Female	
	Information Type	Percentage	Information Type	Percentage
1	[BD]	24.56%	[ACCT]	22.59%
2	[ACCT]	23.70%	[BD]	20.56%
3	[NAME]	23.31%	[NAME]	12.94%
4	[EMAIL]	12.10%	[EMAIL]	13.62%
5	[ID]	2.698%	[CELL]	2.982%
6	[CELL]	2.506%	[ID]	2.739%

percentage of people include their cellphone and ID number in their passwords (less than 3%).

4) Gender Password Preference: As the user ID number in our dataset actually contains gender information (i.e., the second-to-last digit in the ID number representing gender), we compare the password structures between males and females to see if there is any difference in password preference.

The average password lengths for males and females are 8.41 and 8.51 characters, respectively, which suggests that gender does not greatly affect the length of passwords. Applying the matching method to each gender, we then observe that 61.0% of male passwords contain personal information while only 54.1% of female passwords contain personal information. The top 10 structures for each gender are listed in Table VII, and personal information usage is shown in Table VIII. These results demonstrate that male users are more likely to include personal information in their passwords than female users.

Additionally, we have two other interesting observations. First, the total number of password structures for females is 1,756, which is 10.3% more than that of males. Besides, 28.38% of males' passwords fall into the top 10 structures while only 23.94% of females' passwords fall into the top 10 structures. Thus, passwords created by males seem denser and possibly more predictable. Second, males and females vary significantly in the use of name information. While 23.32% of males' passwords contain their names, only 12.94% of females' passwords contain their names. We conclude that the use of name is the main difference in personal information usage between males and females.

In summary, passwords of males are generally composed of more personal information, especially the name of a user. In addition, the password diversity for males is lower. Our analysis indicates that the passwords of males are more vulnerable to cracking than those of females. At least from the perspective of personal-information-related attacks, our observations are different from the conclusion drawn in [30] that males have slightly stronger passwords than females.

TABLE IX
DOMAIN INFORMATION IN PASSWORDS

Dataset	Password Amount	Domain Info Amount	Percentage
Rockyou	14,344,391	44,025	0.3%
Tianya	26,832,592	29,430 [1]	0.11%
PHPBB	184,389	2,209	1.2%
12306	131,389	490	0.4%
MySpace	37,144	72	0.2%

C. Domain Information

Cao *et al.* [8] proposed using domain information to crack user passwords. It draws our attention because we have shown the involvement of personal information in a user's password, so naturally we are also interested in the involvement of the domain information as another aspect of semantic information in password creation. By domain information, we mean the information of an Internet domain (e.g., a web service). For example, the famous "Rockyou" dataset is leaked from *www.rockyou.com*, and the domain information here is "rockyou." In our personal information study, the domain information is "12306." It is reasonable for users to include domain information in their passwords to keep their passwords different from site to site but still easy to remember. This approach may be promising to balance password security and memorability; however, the idea has not been validated with a large-scale experiment. Therefore, we attempt to verify whether domain information is involved in password creation. In addition to the medium-sized 12306 dataset, we study more password datasets, including Tianya, Rockyou, PHPBB, and MySpace datasets. In each dataset, we search the domain information and its meaningful substrings in the passwords. The results are shown in Table IX.

From Table IX, we can see that some users indeed include domain information in their passwords. Our results indicate that all the datasets examined have 0.11% to 1.2% of passwords that contain domain information. However, the small percentage indicates that while the inclusion of domain information in a password helps users to create different passwords for different websites, not many users are currently using such a method.

III. PERSONAL INFORMATION IN ENGLISH-BASED DATASETS

So far, we have only discussed personal information in 12306 dataset. However, due to cultural or language differences, analyzing a single dataset may result in bias conclusions. To gain a better understanding of how personal information generally resides in passwords, we try to extend the analysis to English-based password datasets. Unfortunately, up to this point, there is no available English-based

TABLE X
NAMESETS

Set Name	Total Number	Unique Number	Common	LongCommon4	LongCommon5
Firstname	138,797,749	4,347,667	1,652	1,519	1,232
Lastname	138,797,749	5,369,437	1,497	1,390	1,179

"Common" means common names, where the name has more than 10,000 occurrences in the dataset, "LongCommon4" means the common names with length no less than 4, and "LongCommon5" means the common names with length no less than 5. We will use these filtered data in our experiments.

TABLE XI
MATCHING RESULTS

Set Name	Total Number	Match4	Exact Match4	Match5	Exact Match5
Rockyou	14,344,391	3,540,629 (24.7%)	6,919 (0.05%)	1,750,702 (12.2%)	7,153 (0.05%)
PHPBB	184,389	32,180 (17.5%)	1,709 (0.9%)	14,418 (7.8%)	1,661 (0.9%)
MySpace	37,144	11,521 (31.0%)	221 (0.6%)	5,965 (16.1%)	206 (0.5%)

"Match4" means the name is at least 4-characters long. "Exact Match" indicates the password is exactly the same as the password.

password dataset that incorporates personal information. Due to the lack of personal information, English-based datasets cannot provide us with an accurate correlation as we had in the 12306 dataset. However, we can still make use of some easily inferable personal information from the password itself for a pilot study. One type of inferable information is the user's name. Specifically, we examine the name usage in user passwords by matching commonly used names to the passwords. A name used in the password is a strong indicator that the user has included personal information in the password. Though the name included in the password may have nothing to do with the account owner, the probability of such cases should be fairly low.

A. Methodology

In this study, we use the three English-based leaked password datasets (Rockyou, Phpbb, and Myspace) as in Section II-C and two name datasets (firstname set and lastname set) collected from Facebook. While the three password datasets have been used extensively in many works, the two name datasets are not as prevalent in the research. We show the basic statistics on these two sets in Table X.

For each of the leaked password datasets, we match the password to a first name or last name from the name datasets. Specifically, we try to find exact occurrences of the name in the passwords. For instance, "mary" can be matched to a password "maryspassword." However, this method inevitably results in wrong matches since some 2-grams or 3-grams are widely shared in English words, and people are known to use words from their first language in their passwords. Thus, it may not be clear whether the user is using a name or an English word that coincidentally contains the name. To mitigate this problem, we deliberately ignore names that are less than 4 or 5 characters long in 2 separate experiments. Furthermore, we only consider common names with more than 10,000 occurrences in the datasets to mitigate the effect of too many wrong matches from less commonly used names. We show the filtered results in Table XI. A match is considered found when a first name or a last name is in the password.

Note that there could be both under-matching and over-matching in this study. On the under-matching side, short names such as "Mary" and "Dave" are ignored under "LongCommon5" criteria. Furthermore, some users may use their name initials in passwords, which makes accurate matching much harder. Our study does not consider such cases. On the over-matching side, the problem of commonly shared n-grams still persists. With stricter filtering (ignoring short names), there should be much fewer wrong matches, which in turn results in under-matching.

B. Results

From Table XI, we can see that websites in English-speaking cultures still have a significant use of names in their passwords. For instance, the largest dataset "Rockyou" has more than 24.7% of passwords containing a name of length at least 4, and 12.2% of passwords containing a name of at least length 5. Furthermore, the exact match is not negligible. Exact matches are a strong indication of the use of personal information. In the 12306 dataset, the exact match has lower than a 1% probability, which is basically consistent with this study. Similarly, PHPBB and MySpace also indicate a large name use in their password sets. Our findings are consistent with the expectation that people from different cultures and language backgrounds tend to include their personal information in their passwords. Furthermore, the extent of name use in their passwords does not largely differ—while 12306 has roughly 23% name use in a passwords, the three English-based datasets have 17.5% to 31.0% name use when only names of at least length 4 are considered.

Although our experiments shed light on general personal name use in English-based passwords, we cannot use these datasets in our study on personal information correlation and password cracking since all personal information is inferred from passwords.

IV. Correlation Quantification

While the analysis above show the correlation between each type of personal information and passwords, they cannot accurately measure the degree of personal information involvement in an individual password. Thus, we introduce a novel metric—Coverage—to quantify the involvement of personal information in the creation of an individual password in an accurate and systematic fashion.

A. Coverage

The value of Coverage ranges from 0 to 1. A larger Coverage implies a stronger correlation. Coverage "0" means no personal information is included in a password, and Coverage "1" means the entire password is perfectly matched with one type of personal information. Though Coverage is mainly used for measuring an individual password, the average Coverage also reflects the degree of correlation in a set of passwords. In the following, we describe the algorithm to compute Coverage, presents a detailed example to illustrate how Coverage works, and elaborates the key features of Coverage.

1) Computation Method: To compute Coverage, we take the password and personal information in terms of strings as input and adopt a sliding window approach. To conduct the computation, a dynamic-sized window sliding from the beginning to the end of the password is maintained. The initial size of the window is 2. If the segment behind the window matches to a certain type of personal information, the window size grows by 1. Then we try again to match the new segment to the personal information. If a match is found, the window size is further enlarged until a mismatch happens. At this point, the window resets to the initial size and continues sliding from where the mismatch happens.

Meanwhile, an array called *tag array* with the same length as the password is used to record the length of each matched password segment. For example, assuming a password with a length of 8, and its tag array is [4,4,4,4,0,0,2,2]. The first four elements in the array (i.e., {4,4,4,4}), indicate that the first 4 password symbols match a certain type of personal information. The following two elements in the array ({0,0} indicate that the 5th and 6th symbols have no match. The last two elements in the array ({2,2}) imply that the 7th and 8th symbols again match a certain type of personal information. The personal information types matched with different password segments may or may not be the same. After eventually sliding window through the entire password string, the tag array is used to compute the value of Coverage—the sum of squares of the matched password segment length divided by the square of the password length. Mathematically, we have

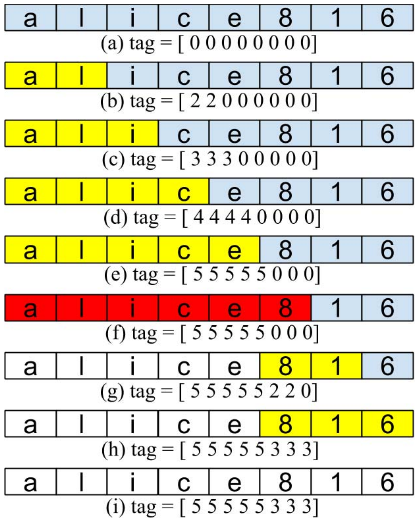
$$CVG = \sum_{i=1}^{n} (\frac{l_i^2}{L^2}), \tag{1}$$

where n denotes the number of matched password segments, l_i denotes the length of the corresponding matched password segment, and L is the length of the password. We show the algorithm of computing Coverage in Algorithm 2. A match is found if at least a 2-symbol long password segment matches to a substring of certain personal information.

Algorithm 2 Compute Coverage

```
procedure CVG($pwd$,$infolist$)
    $windowsize \leftarrow 2$
    $pwdlen \leftarrow$ len($pwd$)
    $matchtag \leftarrow [0]*pwdlen$
    $matchmark \leftarrow 0$
    $cvg \leftarrow 0$
    while $windowsize \leq$len($pwd$) do
        $passseg \leftarrow pwd[0 : windowsize]$
        if $passseg$ = substring of $anyinfo$ in $infolist$ then
            for $j \leftarrow matchmark$ to $matchmark$+$windowsize$ do
                $matchtag[j] \leftarrow windowsize$
            end for
            if $windowsize$ != len($pwd$) then
                $windowsize \leftarrow windowsize$+1
            end if
        else
            $matchmark \leftarrow matchmark$+$windowsize$
            $pwd \leftarrow pwd[windowsize :]$
            $windowsize \leftarrow 2$
        end if
    end while
    for $eachitem$ in $matchtag$ do
        $cvg \leftarrow cvg + eachitem$
    end for
    return $cvg/(pwdlen * pwdlen)$
end procedure
```

To better illustrate how Coverage is computed, we show a simple example in Figure 1. Here we assume a user named Alice, who was born on August 16, 1988. One password of Alice happens to be "alice816." If applying the matching algorithm in Section II-B.2, the structure of this password will be [NAME][BD]. Apparently, her password is highly related to her personal information. To quantify this correlation, we follow the Algorithm 2 to compute Coverage for her. The computation steps are shown in Figure 1, and each step is detailed as follows. In step (a), the tag array is initialized as [0,0,0,0,0,0,0,0]. Note the personal information includes "alice" as the name and "19880816" as the birthdate. In step (b), the window size is initialized to 2 so that the first two symbols in the password are covered. As "al" is a substring of Alice's name, a match is found. Therefore, we extend the window size by 1, and the tag array is updated as [2,2,0,0,0,0,0,0]. From step (c) to step (e), the window keeps growing since matches are continuously found. The tag array also keeps being updated. Until step (f), the window now covers "alice8," which is not a match of "alice" or "19880816." Therefore, the window size is reset to 2 and the window slides to the position of the symbol of the previous window that causes the mismatch (i.e., the position of "8"). The tag array remains unchanged. In step (g), the window of size 2 now covers "81," which is a substring of her birthdate, so again we extend the window by 1 and update the tag array to [5,5,5,5,5,2,2,0]. After the window grows by 1 in step (h),



(a) tag = [0 0 0 0 0 0 0 0]

(b) tag = [2 2 0 0 0 0 0 0]

(c) tag = [3 3 3 0 0 0 0 0]

(d) tag = [4 4 4 4 0 0 0 0]

(e) tag = [5 5 5 5 5 0 0 0]

(f) tag = [5 5 5 5 5 0 0 0]

(g) tag = [5 5 5 5 5 2 2 0]

(h) tag = [5 5 5 5 5 3 3 3]

(i) tag = [5 5 5 5 5 3 3 3]

Fig. 1. An Example of Coverage Computing. Gray boxes hold unvisited password symbols. Yellow and red boxes denote that the symbols inside are covered by the sliding window. White boxes denote that the symbols inside have been settled (i.e. the window stops extending).

"816" is again found as a match. The tag array is updated to [5,5,5,5,5,3,3,3]. The window does not grow or slide anymore since it has reached the end of the password. In the last step (i), the tag array is ready to be used in computing the Coverage value. Based on Equation 1, the coverage is computed as

$$CVG = \sum_{i=1}^{2} \frac{l_i^2}{L^2} = \frac{5^2 + 3^2}{8^2} = 0.52.$$

Coverage is independent of password datasets. As long as we can build a complete string list of personal information, Coverage can accurately quantify the correlation between a user's password and its personal information. For personal information segments with the same length, Coverage stresses the continuation of matching. A continuous match is stronger than fragmented matches. That is to say, for a given password of length L, a matched segment of length l ($l \leq L$) has a stronger correlation to personal information than two matched segments of length l_1 and l_2 with $l = l_1 + l_2$. For example, a matched segment of length 6 is expected to have a stronger correlation than 2 matched segments of length 3. This feature of Coverage is desirable because multiple shorter segments (i.e., originated from different types of personal information) are usually harder to guess and more likely to involve a wrong match due to coincidence. Since it is difficult to differentiate a real match from a coincidental match, we would like to minimize the effect of false matches by taking squares of the matched segments to compute Coverage in favor of a continuous match. Coverage is independent of password datasets. As long as we can build a complete string list of personal information, Coverage can quantify the correlation between the password and these information.

B. Coverage Results on 12306

We compute the Coverage value for each user in the 12306 dataset and show the result as a cumulative distribution function in Figure 2. To easily understand the value of Coverage, we discuss a few examples to illustrate the implication of a roughly 0.2 Coverage. Suppose we have a 10-symbol-long password. One matched segment with length 5 will yield 0.25 Coverage. Two matched segments with length 3 (i.e., in total 6 symbols are matched to personal information) yield 0.18 Coverage. Moreover, 5 matched segments with length 2 (i.e., all symbols are matched but in a fragmented fashion) yield 0.2 Coverage. Apparently, Coverage of 0.2 indicates a fairly high correlation between personal information and a password.

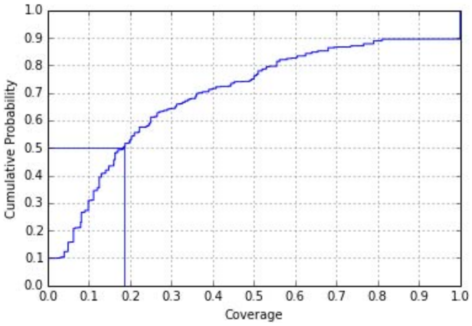


Fig. 2. Coverage distribution - 12306.

The median value for a user's Coverage is 0.186, which implies that a significant portion of user passwords have relatively high correlation to personal information. Furthermore, around 10.5% of users have Coverage of 1, which means that 10.5% of passwords are perfectly matched to exactly one type of personal information. However, around 9.9% of users have zero Coverage, implying no use of personal information in their passwords.

The average Coverage for the entire 12306 dataset is 0.309. We also compute the average Coverages for male and female groups, since we observe that male users are more likely to include personal information in their passwords in Section II-B.4. The average Coverage for the male group is 0.314, and the average Coverage for the female group is 0.269. This complies with our previous observation and indicates that the correlation for male users is higher than that of female users. Conversely, it also shows that Coverage works very well to quantify the correlation between passwords and personal information.

C. Coverage Usage

Coverage could be very useful for constructing password strength meters, which have been reported as mostly ad-hoc [13]. Most meters give scores based on password structure and length or blacklist commonly used passwords (e.g., the notorious "password"). There are also meters that perform simple social profile analysis, such as that a password cannot contain the user's names or the password cannot be the same as the account name. However, these simple analysis mechanisms can be easily manipulated by slightly mangling a password, while the password remains weak. Using the metric of Coverage, password strength meters can be improved to more accurately measure the strength of a password. Moreover, it is straightforward to implement Coverage as a part of the strength measurement (only a few lines of Javascript should do). More importantly, since users cannot easily defeat the Coverage measurement through simple mangling methods, they are forced to select more secure passwords.

Coverage can also be integrated into existing tools to enhance their capabilities. There are several Markov model-based tools that predict the next symbol when a user creates a password [23], [44]. These tools rank the probability of the next symbol based on the Markov model learned from dictionaries or leaked datasets, and then show the most probable predictions. Since most users would be surprised to find that the next symbol in their mind matches the tool's output exactly, they may choose a less predictable symbol. Coverage helps to determine whether personal information prediction ranks high enough in probability to remind a user of avoiding the use of personal information in password creation.

V. Personal-PCFG

After investigating the correlation between personal information and user passwords through measurement and quantification, we further study their potential usage to crack passwords from an attacker's point of view. Based on the PCFG approach [45], we develop Personal-PCFG as an individual-oriented password cracker that can generate personalized guesses towards a targeted user by exploiting the already known personal information.

A. Attack Scenarios

We assume that the attacker knows a certain amount of personal information about the targets. The attacker can be an evil neighbor, a curious friend, a jealous husband, a blackmailer, or even a company that buys personal information from other companies. Under these conditions, targeted personal information is rather easy to obtain by knowing the victim personally or searching online, especially on social networking sites (SNS) [16], [24]. Personal-PCFG can be used in both offline and online attacks.

In traditional offline password attacks, attackers usually steal hashed passwords from victim systems and then try to find out the unhashed values of these passwords. As a secure hash function cannot be simply reversed, the most popular attacking strategy is to guess and verify passwords by brute force. Each guess is verified by hashing a password (salt needs to be added) from a password dictionary and comparing the result to the hashed values in the leaked password database. High-probability password guesses can usually match many hashed values in the password database and thus are expected to be tried first for efficiency purpose. For offline attacks, Personal-PCFG is much faster in guessing the correct password than conventional methods, since it can generate high-probability personalized passwords and verify them first.

For an online attack, the attacker does not even have a hashed password database, so he or she instead tries to log in directly to the real systems by guessing the passwords. It is more difficult to succeed in online attacks than offline attacks because online service systems usually throttle login attempts in a given period of time. If the attempt quota has been reached without inputting a correct password, the account may be locked temporarily or even permanently unless certain actions are taken (e.g., call the service provider). Therefore, online attacks require accurate guesses, which can be achieved by integrating personal information. Personal-PCFG can crack around 1 out of 20 passwords within only 5 guesses.

B. A Revisit of PCFG

Personal-PCFG is based on the basic idea of PCFG method [45] and provides an extension to further improve its efficiency. Before we introduce Personal-PCFG, we briefly revisit principles of PCFG. PCFG pre-processes passwords and generates base password structures such as "$L_5D_3S_1$" for each of the passwords. Starting from high-probability structures, the PCFG method substitutes the "D" and "S" segments using segments of the same length learned from the training set. These substitute segments are ranked by probability of occurrence learned from the training set. Therefore, high-probability segments will be tried first. One base structure may have a number of substitutions; for example, "$L_5D_3S_1$" can have "$L_5$123!" and "$L_5$691!" as its substitutions. These new representations are called pre-terminal structures. No "L" segment is currently substituted since the space of alpha strings is too large to learn from the training set. Next, these pre-terminals are ranked from high probability to low probability. Finally "L" segments are substituted using a dictionary to generate actual guesses. Besides, PCFG method also carries an efficient algorithm to enumerate passwords from high probability to low probability on the fly. These guesses are hashed to compare with the values in password databases. Since PCFG can generate statistically high-probability passwords first, it can significantly reduce the guessing number of traditional dictionary attacks.

C. Personal-PCFG

Personal-PCFG leverages the basic idea of PCFG. Besides "L," "D," and "S" symbols, it features more semantic symbols, including "B" for birthdate, "N" for name, "E" for email address, "A" for account name, "C" for cellphone number, and "I" for ID number. Richer semantics make Personal-PCFG more accurate in guessing passwords. To make Personal-PCFG work, an additional personal information matching phase and an adaptive-substitution phase are added to the original PCFG method. Therefore, Personal-PCFG has 4 phases in total, and the output of each phase will be fed to the next phase as input. The output of the last phase is the actual guesses. We now describe each phase in detail along with simple examples.

1) Personal Information Matching: Given a password, we first match the entire password or a substring of the password to its personal information. The detailed algorithm is similar to Algorithm 1. However, this time we also record the length of the matching segment. We replace the matched segments in a password with corresponding symbols and mark each symbol with its length. Unmatched segments remain unchanged. For instance, we assume Alice was born on August 16, 1988, and her password is "helloalice816!." The matching phase will replace "alice" with "N_5" and "816" with "B_3." The leftover "hello" is kept unchanged. Therefore the outcome of this phase is "$helloN_5B_3$!."

2) Password Pre-Processing: This phase is similar to the pre-processing routine of the original PCFG; however, based on the output of the personal information matching phase, the segments already matched to personal information will not be processed. For instance, the sample structure "$helloN_5B_3$!" will be updated to "$L_5N_5B_3S_1$" in this phase. Now the password is fully described by semantic symbols of Personal-PCFG, and the output in this phase provides base structures for Personal-PCFG.

3) Guess Generation: Similar to the original PCFG, we replace "D" and "S" symbols with actual strings learned from the training set in descending probability order. "L" symbols are replaced with words from a dictionary. Similar to PCFG [45], we output the results on the fly, so we do not need to wait for all of the possible guesses being calculated and sorted. The guesses keep being generated for next step. Note that we have not replaced any symbols for personal information, so the guesses are still not actual guesses. We do not handle personal information in this step, since personal information of each user is different. Thus, the personal information symbols can only be substituted until the target user is specified. Therefore, in this phase, the base structures only generate pre-terminals, which are partial guesses that contain part of actual guesses and part of Personal-PCFG semantic symbols. For instance, the example "$L_5N_5B_3S_1$" is instantiated to "$helloN_5B_3$!" if "hello" is the first 5-symbol-long string in the input dictionary and "!" has the highest probability of occurrence among 1-symbol special characters in the training set. Note that for "L" segments, each word of the same length has the same probability. The probability of "hello" is simply $\frac{1}{N}$, in which N is the total number of words of length 5 in the input dictionary.

4) Adaptive Substitution: In the original PCFG, the output of guess generation can be applied to any target user. However, in Personal-PCFG, the guess will be further instantiated with personal information, which is specific to only one target user. Each personal information symbol is replaced by corresponding personal information of the same length. If there are multiple candidates of the same length, all of them will be included for trial. In our example "$helloN_5B_3$!," "N_5" will be directly replaced by "alice." However, since "B_3" has many candidate segments and any length 3 substring of "19880816" may be a candidate, the guesses include all substrings, such as "helloalice198!," "helloalice988!," ... , "helloalice816!" We then try these candidate guesses one by one until we find one candidate that matches exactly the password of Alice. Note that on the contrary of having multiple candidates, not all personal information segments can be replaced because same-length segments may not always be available. For instance, a pre-terminal structure "$helloN_6B_3$!" is not suitable for Alice since her name contains only 5 characters. In this case, no guess from this structure should be generated for Alice.

D. Cracking Results

We compare the performance of Personal-PCFG and the original PCFG using the 12306 dataset, which has 131,389 users. We use half of the dataset as the training set, and the other half as the testing set. For the "L" segments, both methods need to use a dictionary, which is a critical part of password cracking. To eliminate the effect of an unfair dictionary selection, we use "perfect" dictionaries in both methods. Perfect dictionaries are dictionaries we collected directly from the testing set, so that any string in the dictionary is useful and any letter segments in the passwords must appear in the dictionary. Thus, a perfect dictionary is guaranteed to find correct string segments efficiently. In our study, both the PCFG perfect dictionary and the Personal-PCFG perfect dictionary contain 15,000 to 17,000 entries.

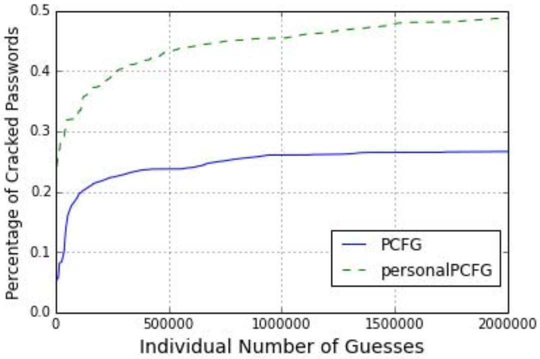


Fig. 3. PCFG vs. Personal-PCFG (Offline).

We use *individual number of guesses* to measure the effectiveness of Personal-PCFG compared with PCFG. The individual number of guesses is defined as the number of password guesses generated for cracking each individual account (e.g., 10 guess trials for each individual account), which is independent of the password dataset size. In Personal-PCFG, the aggregated individual number of guesses (i.e., the total number of guesses) is linearly dependent on the password dataset size. By contrast, in a conventional cracking strategies like PCFG, each guess is applied to the entire user base, and thus the individual number of guesses equals the total number of guesses. Regardless of such discrepancies between Personal-PCFG and conventional cracking methods, the bottleneck of password cracking lies in the number of hashing operations. Due to the salting mechanism, the total number of hashes is bounded by $G \cdot N$ for both Personal-PCFG and other password crackers, where G is the individual number of guesses and N is the size of the dataset.

Given the different number of guesses, we compute the percentage of those cracked passwords in the entire password trial set. Figure 3 shows the comparison result of the original PCFG and Personal-PCFG in an offline attack. Both methods have a quick start because they always try high probability guesses first. Figure 3 clearly indicates that Personal-PCFG can crack passwords much faster than PCFG does. For example, with a moderate size of 500,000 guesses, Personal-PCFG achieves a similar success rate that can be reached with more than 200 million guesses by the original PCFG. Moreover, Personal-PCFG is able to cover a larger password space than PCFG because personal information provides rich personalized strings that may not appear in the dictionaries or training set.

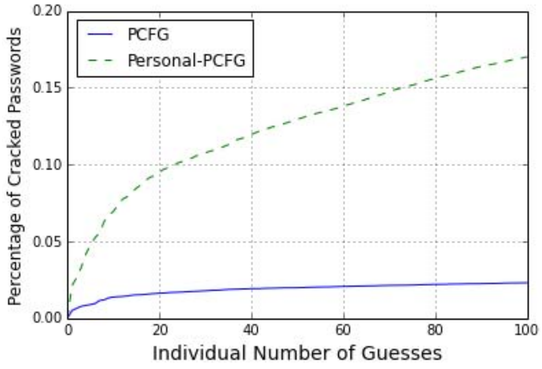


Fig. 4. PCFG vs. Personal-PCFG (Online).

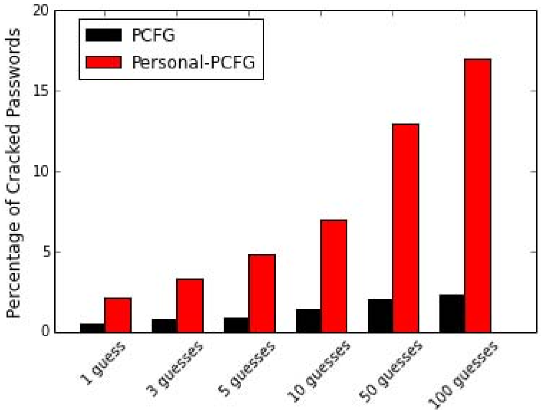


Fig. 5. Representative Points – Online attacks.

Personal-PCFG not only improves the cracking efficiency in offline attacks but also increases the guessing success rate in online attacks. Online attacks are only able to try a small number of guesses in a certain time period due to the system constraint on the login attempts. Thus, we limit the number of guesses to be at most 100 for each target account. We present the results in Figure 4, from which it can be seen that Personal-PCFG is able to crack 309% to 634% more passwords than the original PCFG. We then show several representative guessing numbers in Figure 5. For a typical system that allows 5 attempts to input the correct passwords, Personal-PCFG is able to crack 4.8% of passwords within only 5 guesses. Meanwhile, the percentage is just 0.9% for the original PCFG, and it takes around 2,000 more guesses for PCFG to reach a success rate of 4.8%. Thus, Personal-PCFG is more efficient to crack the passwords within a small number of guesses.

Therefore, Personal-PCFG substantially outperforms PCFG in both online and offline attacks, due to the integration of personal information into password guessing. The extra requirement of Personal-PCFG on personal information can be satisfied by knowing the victim personally or searching on social networking sites (SNS).

VI. Password Protection

In almost all systems, users are able to choose and update their passwords. However, they may sacrifice password security for usability since a long and random secure password is less memorable. As user passwords are easier to be compromised when their personal information is available to the attacker, we investigate how users can protect their passwords against such attacks.

To increase the password security while retaining good memorability, we suggest the *Distortion Function*, which performs a transformation on user passwords. A distortion function converts user passwords to be more secure by breaking password semantics. Therefore, the user only needs to remember the original password and apply a simple function on it to create a stronger password. This distortion function can be chosen by users, so it could be either linear or non-linear.

We conduct a proof-of-concept study to show the effectiveness of a distortion function on password security. In this study, two types of distortion functions are introduced. The first type maps each password character to another character. For instance, add_1 function simply replaces each letter with the one 1 letter later in the alphabet and replaces a single digital number i with $(i + 1)$ mod 10. It is similar to the Caesar Cipher [21]. In another example, add_{pi} is a non-linear function that shifts password letters by a corresponding position specified by π, which is 314159 . . .. It shifts a letter to N positions later in the alphabet with wraparound, where N is the corresponding digit of π. For instance, "abc" becomes "dcg" after applying this distortion function. The second type of distortion function adds an extra fixed character between any pair of characters in passwords. The length of a password becomes $2l - 1$, where l is the length of the original password. We call this distortion function gap_x, in which "x" represents the extra symbol. For example, when x = "a", Alice's password "alice816" will be extended to "aalaiacaea8a1a6" after the distortion function is applied.

The distortion function must be simple enough for users to remember and generate the passwords. We apply a number of easy-to-remember distortion functions on each of the passwords in the 12306 dataset individually and calculate the Coverage for the converted passwords. As Figure 6 shows, the distortion functions are effective in increasing password security by greatly reducing the correlation between user passwords and personal information. Moreover, we notice that the impacts of various distortion functions are also different. For example, $add_!$ performs the best (Coverage is 0 for all users, so it is not shown in Figure 6) since users rarely have special characters in their personal information. Surprisingly, the non-linear add_{pi} function does not produce a better result than other linear functions such as add_1 because digits preferred by Chinese users are more likely to have coincidentally wrong matches due to its low entropy.

We conclude that distortion functions can mitigate the problem of including personal information in user passwords without significantly sacrificing password usability. Moreover, distortion functions are also a cure for semantics-aware password cracking methods [41], which leverage semantic patterns in passwords to crack other passwords. After applying a distortion function, the semantic pattern is no longer available. Distortion functions are also effective against PCFG [45], since it generates unrecognizable letter segments, which are not likely to be covered in commonly-used password dictionaries.

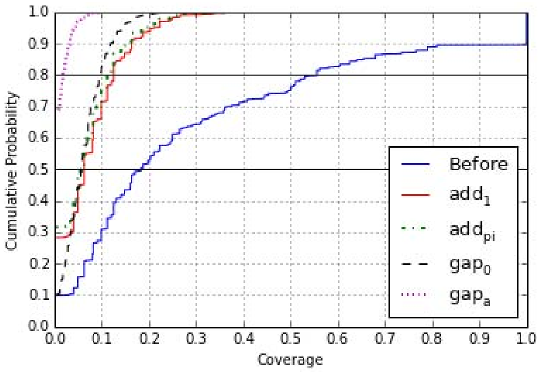


Fig. 6. Coverage distribution.

What differentiates the use of a distortion function from using an extra piece of user-chosen secret is that it breaks password semantics, which makes it much harder for an attacker to interpret since there could be many possible (password, function) pairs that produce the same final password. Thus, it becomes increasingly difficult for an attacker to learn how users construct their passwords, and the inaccurate training cripples the efficiency of training-based attack. Furthermore, we do not claim that a distortion function is able to eliminate the trade-off between security and usability. Instead, the distortion function is only used to effectively mitigate the problem of personal information residing in passwords.

VII. Discussion

A. Limitation

In most of our study, only a single dataset is used. Most users of the 12306 website are Chinese, and the number of males and females is not balanced. Consequently, there might be cultural, language, and gender biases on the analytical results. Moreover, the effectiveness of the Coverage metric and Personal-PCFG is only validated on a single website. Publicly available password datasets leaked with personal information are very rare. We have done an estimation on English-based datasets, which shows that no matter the language or culture, people include personal information in their passwords, though to slightly different extents. However, although we have tried to extend the analytical work to more datasets, it is infeasible to test the effectiveness of Coverage and Personal-PCFG since personal information is directly derived from passwords.

B. Ethical Considerations

Though using leaked password datasets for more accurate and convincing study has been a mainstream method on password studies, we do realize that studying leaked datasets involves ethical concerns. We only use the datasets for researching purpose. All data are carefully stored and used. We will not expose any personal information or password or use this information in any way other than for research use.

VIII. Related Work

A. Password Study

Many works on password have been done through decades of password-based authentication. In one of the earliest works [32], Morris and Thompson found that passwords are quite simple and thus are vulnerable to guessing attacks. Nowadays, passwords studies are done on a much larger scale, and thus they reveal more insightful and accurate characteristics of passwords. For example, Mazurek *et al.* [30] measured 25,000 passwords from a university and revealed correlation between demographics or other factors, such as gender and field of study. Li *et al.* [27] conducted a large-scale measurement study on Chinese passwords, in which more than 100 million real-life passwords are studied and differences between passwords in Chinese and other languages are presented. Bonneau [2] studied the language effect on user passwords from more than 70 million passwords. Bonneau et al. [4] also found that a user's birthdate appears extensively in 4-digit PINs. Malone *et al.* [28] studied the distribution of passwords on several large leaked datasets and found that user passwords fit Zipf distribution well.

There are also many works investigating more specific aspects of passwords. For instance, Yan *et al.* [46] and Kuo *et al.* [25] investigated the mnemonic passwords. Veras *et al.* [42] showed the importance of a date in passwords. Das *et al.* [11] studied how users mangle one password for different sites. Schweitzer *et al.* [39] studied the keyboard pattern in passwords. Apart from the password itself, human habits and psychology toward password security are also being extensively investigated [15], [18].

B. Strength Measurement

Password Strength measurement still remains a challenge. It has been shown that Shannon entropy can hardly accurately describe the security level of passwords [7], [22], [36], [44]. Thus, a number of metrics to measure passwords is introduced. Massey [29] proposed guessing entropy, which shows the expected number of guesses needed to make a correct guess. Several other most commonly used metrics include marginal guesswork μ_α [36], which measures the number of expected guess needed to succeed with probability α, and the marginal success rate λ_β [5], which is the probability to succeed in β guesses.

C. Password Cracking

Password-cracking methods have been discussed for more than 30 years. Attackers usually try to recover plain-text passwords from a hashed password database. While reverse hashing function is infeasible, dictionary attacks are found effective [32]. Reducing time-memory trade-off in passwords [17] even made dictionary attacks much more efficient. Rainbow table [34] further reduces the table number in [17] using multiple reduction functions. However, in recent years as the password policy has become strict, simple dictionary passwords are less common. More powerful attacks are then created. Narayanan and Shmatikov [33] used the Markov model to generate guesses based on the fact that passwords are phonetically similar to users' native languages. OMEN [9] improves [33] to crack passwords by using a more optimal guessing order. Another advanced attack is by using PCFG [45], on which Personal-PCFG is built.

Veras *et al.* [41] tried to leverage semantic patterns in passwords. Besides, while attacking a hashed password database remains the main attacking scenario, there are other attacks on different scenarios, such as video eavesdropping [1].

D. Security Enhancement

Facing the fact that passwords are a vulnerable authentication scheme, alternatives such as graphics-based [12], [20] or biometrics-based [19] authentication methods have been proposed. However, text-based passwords are expected to continue to dominate [3].

Instead of trying to replacing passwords, there are many works focusing on enhancing password security, including password strength feedback [10], [23], [44], multiple factor authentication [6], [35], [38], and security enhancement tools [31], [37], [40], [43]. However, an ideal solution to enhance password security without sacrificing usability, if existed, is yet to be found.

IX. Conclusion

In this work, we conduct a comprehensive quantitative study on how user personal information resides in human-chosen passwords. To the best of our knowledge, we are the first to systematically analyze personal information in passwords. We have some interesting and quantitative discoveries such as 3.42% of the users in the 12306 dataset use their birthdate as a password, and male users are more likely than female users to include their name in passwords. We then introduce a new metric, Coverage, to accurately quantify the correlation between personal information and a password. Our coverage-based quantification results further confirm our disclosure on the serious involvement of personal information in password creation, which makes a user password more vulnerable to a targeted password cracking. We develop Personal-PCFG based on PCFG but consider more semantic symbols for cracking a password. Personal-PCFG generates personalized password guesses by integrating personal information in the guesses. Our experimental results demonstrate that Personal-PCFG is significantly faster than PCFG in password cracking and eases the feasibility of mounting online attacks. Finally, we propose using distortion functions to protect weak passwords that include personal information. Through a proof-of-concept study, we confirm that distortion functions are effective in defending against personal-information-related and semantics-aware attacks.

Acknowledgments

We would like to thank the anonymous reviewers for their insightful comments.

References

[1] D. Balzarotti, M. Cova, and G. Vigna, "ClearShot: Eavesdropping on keyboard input from video," in *Proc. IEEE Symp. Security Privacy*, May 2008, pp. 170–183.

[2] J. Bonneau, "The science of guessing: Analyzing an anonymized corpus of 70 million passwords," in *Proc. IEEE Symp. Security Privacy*, May 2012, pp. 538–552.

[3] J. Bonneau, C. Herley, P. C. van Oorschot, and F. Stajano, "The quest to replace passwords: A framework for comparative evaluation of Web authentication schemes," in *Proc. IEEE Symp. Security Privacy*, May 2012, pp. 553–567.

[4] J. Bonneau, S. Preibusch, and R. Anderson, "A birthday present every eleven wallets? The security of customer-chosen banking pins," in *Financial Cryptography and Data Security*. Berlin, Germany: Springer, 2012.

[5] S. Boztas, "Entropies, guessing, and cryptography," Ph.D. dissertation Dept. Math., Royal Melbourne Inst. Technol., Melbourne, VIC, Australia, Tech. Rep., 1999.

[6] J. Brainard, A. Juels, R. L. Rivest, M. Szydlo, and M. Yung, "Fourth-factor authentication: Somebody you know," in *Proc. ACM CCS*, 2006, pp. 168–178.

[7] C. Cachin, "Entropy measures and unconditional security in cryptography," Ph.D. dissertation, Dept. Comput. Sci., Swiss Federal Inst. Technol. Zürich, Zürich, Switzerland, 1997.

[8] P. Cao, H. Li, K. Nahrstedt, Z. Kalbarczyk, R. Iyer, and A. J. Slagell, "Personalized password guessing: A new security threat," in *Proc. ACM Symp. Bootcamp Sci. Security*, 2014, p. 22.

[9] C. Castelluccia, A. Chaabane, M. Dürmuth, and D. Perito. (Apr. 2013). "When privacy meets security: Leveraging personal information for password cracking." [Online]. Available: https://arxiv.org/abs/1304.6584

[10] C. Castelluccia, M. Dürmuth, and D. Perito, "Adaptive password-strength meters from Markov models," in *Proc. NDSS*, 2012.

[11] A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang, "The tangled Web of password reuse," in *Proc. NDSS*, 2014, pp. 23–26.

[12] D. Davis, F. Monrose, and M. K. Reiter, "On user choice in graphical password schemes," in *Proc. USENIX Security*, 2004, p. 211.

[13] X. de C. de Carnavalet and M. Mannan, "From very weak to very strong: Analyzing password-strength meters," in *Proc. NDSS*, 2014, pp. 23–26.

[14] S. Egelman, A. Sotirakopoulos, I. Muslukhov, K. Beznosov, and C. Herley, "Does my password go up to eleven?: The impact of password meters on password selection," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 2013, pp. 2379–2388.

[15] D. Florencio and C. Herley, "A large-scale study of Web password habits," in *Proc. ACM WWW*, 2007, pp. 657–666.

[16] R. Gross and A. Acquisti, "Information revelation and privacy in online social networks," in *Proc. ACM WPES*, 2005, pp. 71–80.

[17] M. E. Hellman, "A cryptanalytic time-memory trade-off," *IEEE Trans. Inf. Theory*, vol. 26, no. 4, pp. 401–406, Jul. 1980.

[18] A. E. Howe, I. Ray, M. Roberts, M. Urbanska, and Z. Byrne, "The psychology of security for the home computer user," in *Proc. IEEE Symp. Security Privacy*, May 2012, pp. 209–223.

[19] A. K. Jain, A. Ross, and S. Pankanti, "Biometrics: A tool for information security," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 2, pp. 125–143, Jun. 2006.

[20] I. Jermyn, A. J. Mayer, F. Monrose, M. K. Reiter, and A. D. Rubin, "The design and analysis of graphical passwords," in *Proc. USENIX Security*, 1999, p. 1.

[21] C. Kaufman, R. Perlman, and M. Speciner, *Network Security: Private Communication in a Public World*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1995.

[22] P. G. Kelley *et al.*, "Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms," in *Proc. IEEE Symp. Security Privacy*, May 2012, pp. 523–537.

[23] S. Komanduri, R. Shay, L. F. Cranor, C. Herley, and S. Schechter, "Telepathwords: Preventing weak passwords by reading users' minds," in *Proc. USENIX Security*, 2014, pp. 591–606.

[24] B. Krishnamurthy and C. E. Wills, "On the leakage of personally identifiable information via online social networks," in *Proc. ACM COSN*, 2009, pp. 7–12.

[25] C. Kuo, S. Romanosky, and L. F. Cranor, "Human selection of mnemonic phrase-based passwords," in *Proc. ACM SOUPS*, 2006, pp. 67–78.

[26] Y. Li, H. Wang, and K. Sun, "A study of personal information in human-chosen passwords and its security implications," in *Proc. IEEE INFOCOM*, Apr. 2016, pp. 1–9.

[27] Z. Li, W. Han, and W. Xu, "A large-scale empirical analysis of Chinese Web passwords," in *Proc. USENIX Security*, 2014, pp. 559–574.

[28] D. Malone and K. Maher, "Investigating the distribution of password choices," in *Proc. ACM WWW*, 2012, p. 310.

[29] J. L. Massey, "Guessing and entropy," in *Proc. IEEE Int. Symp. Inf. Theory*, 1994, p. 204.

[30] M. L. Mazurek *et al.*, "Measuring password guessability for an entire University," in *Proc. ACM CCS*, 2013, pp. 173–186.

[31] D. McCarney, D. Barrera, J. Clark, S. Chiasson, and P. C. van Oorschot, "Tapas: Design, implementation, and usability evaluation of a password manager," in *Proc. ACM ACSAC*, 2012, pp. 89–98.

[32] R. Morris and K. Thompson, "Password security: A case history," *Commun. ACM*, vol. 22, no. 11, pp. 594–597, 1979.

[33] A. Narayanan and V. Shmatikov, "Fast dictionary attacks on passwords using time-space tradeoff," in *Proc. ACM CCS*, 2005, pp. 364–372.

[34] P. Oechslin, "Making a faster cryptanalytic time-memory trade-off," in *Proc. Adv. Cryptol. Conf. (CRYPTO)*, 2003, pp. 364–372.

[35] B. Pinkas and T. Sander, "Securing passwords against dictionary attacks," in *Proc. ACM CCS*, 2002, pp. 161–170.

[36] J. O. Pliam, "On the incomparability of entropy and marginal guesswork in brute-force attacks," in *Proc. Prog. Int. Conf. Cryptol.-INDOCRYPT*, 2000, pp. 67–79.

[37] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J. C. Mitchell, "Stronger password authentication using browser extensions," in *Proc. USENIX Security*, 2004, pp. 17–32.

[38] S. Schechter, A. B. Brush, and S. Egelman, "It's no secret. Measuring the security and reliability of authentication via 'secret' questions," in *Proc. IEEE Symp. Security Privacy*, May 2009, pp. 375–390.

[39] D. Schweitzer, J. Boleng, C. Hughes, and L. Murphy, "Visualizing keyboard pattern passwords," in *Proc. IEEE VizSec*, Oct. 2009, pp. 69–73.

[40] B. Strahs, C. Yue, and H. Wang, "Secure passwords through enhanced hashing," in *Proc. USENIX LISA*, 2009, p. 93.

[41] R. Veras, C. Collins, and J. Thorpe, "On the semantic patterns of passwords and their security impact," in *Proc. NDSS*, Feb. 2014.

[42] R. Veras, J. Thorpe, and C. Collins, "Visualizing semantics in passwords: The role of dates," in *Proc. IEEE VizSec*, Oct. 2012, pp. 88–95.

[43] L. Wang, Y. Li, and K. Sun, "Amnesia: A bilateral generative password manager," in *Proc. ICDCS*, Jun. 2016, pp. 313–322.

[44] M. Weir, S. Aggarwal, M. Collins, and H. Stern, "Testing metrics for password creation policies by attacking large sets of revealed passwords," in *Proc. ACM CCS*, 2010, pp. 162–175.

[45] M. Weir, S. Aggarwal, B. De Medeiros, and B. Glodek, "Password cracking using probabilistic context-free grammars," in *Proc. IEEE Symp. Security Privacy*, May 2009, pp. 391–405.

[46] J. Yan, A. Blackwell, R. Anderson, and A. Grant, "Password memorability and security: Empirical results," *IEEE Security & Privacy Mag.*, vol. 2, no. 5, pp. 25–31, Sep. 2004.

Yue Li received the B.Eng. degree from the Information Engineering Department, Chinese University of Hong Kong, Hong Kong, in 2013. He is currently pursuing the Ph.D. degree with the Computer Science Department, College of William & Mary, Williamsburg, VA, USA. He is co-advised by Dr. H. Wang and Dr. K. Sun. His research interests mainly lie in secure authentication, network security, and attack forensics analysis.

Haining Wang received the Ph.D. degree in computer science and engineering from the University of Michigan at Ann Arbor, MI, USA, in 2003. He is currently a Professor of Electrical and Computer Engineering from the University of Delaware, Newark, DE, USA. His research interests lie in the areas of security, networking system, and cloud computing.

Kun Sun received the Ph.D. degree from the Department of Computer Science, North Carolina State University. He is currently an Associate Professor with the Department of Information Sciences and Technology, George Mason University. He is also the Director of Sun Security Laboratory. He has more than ten years working experience in both industry and academia. His research focuses on systems and network security. The main thrusts of his research include moving target defense, cyber deception, and disinformation, trustworthy computing, password management, and mobile security. He has authored over 50 technical papers on security conferences and journals, including IEEE S&P, CCS, NDSS, IEEE DSN, ESORICS, ACSAC, IEEE TDSC, and IEEE TIFS, and one paper received the Best Paper Award.