How does personal information reside in human-chosen passwords? –A quantitative study

Yue Li
College of William & Mary
yli@cs.wm.edu

Haining Wang
University of Delaware
hnw@udel.edu

Kun Sun
College of William & Mary
ksun@cs.wm.com

ABSTRACT

Left blank.

Categories and Subject Descriptors

[**Security and privacy**]: *Human and societal aspects of security and privacy*; [**General and reference**]: *Metrics*

General Terms

Security

Keywords

passwords, password cracking, data processing, password protection

1. INTRODUCTION

Left blank

2. PERSONAL INFO IN PASSWORDS

Human-generated passwords are long criticized to be weak. Numerous works have shown that due to memorability requirement, users are more likely to use meaningful strings as their passwords. Therefore, passwords are usually very different from real random strings. For example, ”password” is more likely a password than ”ziorqpe”. As a result, most passwords are within only a small portion of the large password space, making password guessing a lot easier. A natural question is: how do users choose their passwords so that they are different from random strings? The answer to the question has great significance for it has strong security implication to both users as well as systems. If an attacker knows exactly how users construct their passwords, cracking their passwords will become an easy task. On the other hand, if a user knows how other users construct their passwords, the user can easily improve his/her password strength by avoid using these password construction methods.

To this end, researchers have done much to unveil the composition of passwords. Traditional dictionary attacks on passwords have shown that users tend to use dictionary words to construct their passwords. [][] claims that the distribution of characters in passwords is very similar to that in their native languages and people are prone to use words in their languages. [] Shows password words distribution is not the same as word distribution in the language. [Markov] shows that passwords are phonetically memorable. [PCFG] shows that using dictionary words to guess passwords is effective. [][][] indicate that users use keyboard strings such as ”qwerty” and ”qweasdzxc”, trivial strings such as ”password”, ”123456”, and date strings such as ”19951225” in their passwords.

As far as we see, most studies are done at a macro level. We now study user passwords in an individual base. We would like to show that user personal experience plays an important role when users create their passwords. Intuitively, people tend to choose their passwords based on their personal information because human beings are limited by their memory – totally unrelated passwords are much less memorable.

2.1 12306 Dataset

In recent years, many password datasets are exposed to the public. Recent works on password measurement or password cracking are usually based on these datasets. Some of these datasets, such as Rockyou, are very large such that they even constitute millions of passwords. Now we are going to use a dataset which we call 12306 dataset to illustrate how personal information is used in user passwords.

2.1.1 Introduction to dataset

At the end of year 2014, a Chinese dataset is exposed to the public by anonymous attackers. It is said that the dataset is obtained using social engineering[], in which attackers use datasets at hand to try other websites. We call this dataset 12306 dataset because all passwords are from a website www.12306.com. The website is the official website for online railway ticket booking for Chinese users.

12306 dataset contains over 130,000 Chinese passwords. Having witnessed so many large datasets been leaked out, the size of 12306 dataset is just medium. What makes it special is that together with plain text passwords, the dataset also carries several types of user personal information. For example, user’s name, ID number, etc. As the website needs real ID number to register and people need to provide real information to book a ticket, information in the dataset is considered reliable.

WOODSTOCK ’97 El Paso, Texas USA

Table 1: 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%

2.1.2 *Basic Measurement*

We do fundamental measurement to reveal some characteristics of 12306 dataset. After appropriate cleansing, we remove a minor part of passwords (0.2%), with 131,389 good passowrds left for analysis. Note that websites may have different password creation policy. With strict password policy, users may apply mangling rules (For example, $abc- > @bc$ or $abc1$)to their passwords to fulfill the policy requirement. As 12306 website has changed its password policy after the password leakage, we do not know exactly the password policy at the time the dataset is leaked. However, from the dataset, we infer the password policy is quite simple – all passwords need to be no shorter than 6 symbols. There is no restriction on what type of symbols are used. Therefore users are not forced to apply much mangling to their passwords.

The average length of passwords in 12306 dataset is 8.44. Then we show the most common passwords in 12306 dataset. They are listed in Table 1.

From Table 1 we can see that the dominating passwords are trivial passwords (123456, a123456, etc), keyboard passwords (1qaz2wsx and 1q2w3e4r), and ”I love you” passwords. Both ”5101314” and ”woaini1314” means ”I love you forever” in Chinese. The most commonly used Chinese passwords are similar to previous studies [fudan]. However, 12306 dataset is much less congregated. The most popular password ”123456” accounts less than 0.3% of all passwords while the number is 2.17% in [Fudan]. We believe that the sparsity is due to the importance of the website so that users are less prone to use trivial passwords like ”123456”, etc.

Then, we show the basic structure of passwords. The most popular password structures are shown in Table **??**. Our result again shows that Chinese users prefer to use digits in their passwords instead of letters as in English-speaking users. The 5 top structures all have significant portion of digits, in which at most 2 or 3 letters are appended in front.

We reckon that the reason behind may be Chinese users lack vocabulary because Chinese use non-ASCII character set. Digits seem to be the best choice when creating a password.

In conclusion, 12306 dataset is a Chinese password dataset that has general Chinese password characteristics. However, its passwords are more sparse than previously studied datasets.

2.2 Personal Information

As we have mentioned, 12306 dataset not only contains user passwords, it also carries multiple types of personal information. They are:

Table 2: Most Frequent Password Structures

Rank	Structure	Amount	Percentage
1	D_7	10893	8.290%
2	D_8	9442	7.186%
3	D_6	9084	6.913%
4	L_2D_7	5065	3.854%
5	L_3D_6	4820	3.668%
6	L_1D_7	4770	3.630%
7	L_2D_6	4261	3.243%
8	L_3D_7	3883	2.955%
9	D_9	3590	2.732%
10	L_2D_8	3362	2.558%

”D” represents digits and ”L” represents English letters. The number indicates the segment length. For example, D_7 means the password contains 7 digits in a row.

```
1. Name: User's Chinese name
2. Email address: User's registered email address
3. Cellphone number: User's registered cellphone number
4. Account name: the account used to log on the system,
may contain digits and letters. For example, "myacct123".
5. ID number: Government issued ID number.
```

Note that the government issued ID number is an 18-digit powerful number. These digits actually show personal information as well. Digit 1-6 represents the birth place of the owner, Digit 7-14 represents the birthday of the owner, and digit 17 represents the gender of the owner – odd number means male and even number means female. We take out the 8-digit birthday information and treat it separately because birthday information is very important in a password. Therefore, we finally have 6 types personal information - 1)Name, 2)Birthday, 3)Email, 4)Cellphone 5)Account name, and 6) ID number (birthday not included).

2.2.1 *New Password Representation*

To better illustrate how personal information correlates to user passwords, we develop a new representation of password which add more semantic symbols beside the conventional ”D”, ”L” and ”S” symbols, which means digit, letter, and special symbol accordingly. We try to match password to the 6 types of user personal information, and express the passwords with these personal information. For example, a password ”alice1987abc” may be represented as $[Name][Birthday]L_3$ instead of $L_3D_4L_3$ in a traditional measurement. We substitute personal information with corresponding tag ([Name] and [Birthday] in this case). For the segments that are not matched, we still use ”D”,”L”, and ”S” to describe the types of characters.

We believe representation like $[Name][Birthday]L_3$ is better than $L_5D_4L_3$ since it more accurately describe the composition of user passwords. We apply the matching to the whole 12306 dataset to see how these personal information tag appear in such password representations.

2.2.2 *Matching Method*

In order to make personal information password representations, an essential question will be: How do we match the personal information to user passwords? To answer this question, we show the algorithm we used in Algorithm 1.

The high level idea is that we find all substrings of the password and sort them in descending length order. Then we try to match the substrings from longest to shortest to all types of personal information. If one match is found, the leftover password segments are recursively applied the match function until no further match is found. Segments that are not matched to personal information will be processed using the traditional ”LDS” method.

Algorithm 1 Match personal information with password

```
 1: procedure MATCH(pwd,infolist)
 2:     newform ← empty_string
 3:     if l thenen(pwd) == 0
 4:         return empty_string
 5:     end if
 6:     substring ← get_all_substring(pwd)
 7:     reverse_length_sort(substring)
 8:     for eachstring ∈ substring do
 9:         if len(eachstring) ≥ 2 then
10:             if matchbd(eachstring,infolist) then
11:                 tag ← ”[BD]”
12:                 leftover ← pwd.split(eachstring)
13:                 break
14:             end if
15:             . . .
16:             if matchID(eachstring,infolist) then
17:                 tag ← ”[ID]”
18:                 leftover ← pwd.split(eachstring)
19:                 break
20:             end if
21:         else
22:             break
23:         end if
24:     end for
25:     if leftover.size() ≥ 2 then
26:         for i ← 0 to leftover.size()-2 do
27:             newform ← MATCH(leftover[i],infolist) + tag
28:         end for
29:         newform ← MATCH(leftover[leftover.size()−1])+newform
30:     else
31:         newform ← seg(pwd)
32:     end if
33:     return newform
34: end procedure
```

Note that we did not show specific matching algorithm to each type of the personal information (line 10 and line 16). To keep Algorithm 1 clean and simple, we describe the matching methods as follows.

First we make sure the password segments are at least length of 2 for matching. For segment of length 1, we directly map it to digit, letter, or special character. We try to match segments with length 2 or more to each kind of the information. For name information, we first convert Chinese names into Pinyin form, which is alphabetic representation of Chinese. Then we compare password segments to 10 possible permutations of the names, which include $lastname + firstname$, $last\_initial + firstname$, etc. If the segment is exactly same as any of the permutations, we consider a match is found. We list all the 10 permutations in the Appendices. For birthday information, we list 17 possible permutations and compare password segments to each of the permutation, if the segment is same as any permutations, we consider a match is found. We list all the birthday permutations in the Appendices. For account name, cellphone number, and ID number, we further restrain the length of segment to be at least 4 to avoid coincidence. We believe a match of length 4 is very likely to be an actual match. If the segment is a substring of any of the 3 personal information, we regard it a match to the corresponding personal information. Note that for some password segment, it may match to multiple types of personal information. In this case, we count each type that is matched once.

Table 3: Most Frequent Password Structures

Rank	Structure	Amount	Percentage
1	D7	7122	5.420%
2	[ACCT]	6820	5.190%
3	[NAME][BD]	5410	4.117%
4	D6	4886	3.718%
5	[BD]	4470	3.402%
6	D8	4245	3.230%
7	[EMAIL]	3807	2.897%
8	L1D7	3296	2.508%
9	[NAME]D7	2949	2.244%
10	[NAME]D3	2363	1.798%

Table 4: Most Popular Personal Information

Rank	Information Type	Amount	Percentage
1	[BD]	32373	24.63%
2	[NAME]	29646	22.56%
3	[ACCT]	21324	16.22%
4	[EMAIL]	11020	8.387%
5	[ID]	1210	0.920%
6	[CELL]	634	0.482%

2.2.3 Matching Result

After applying Algorithm 1 to 12306 dataset. We found that 70,892 out of 131,389 (54.0%) of the passwords contain at least one of the 6 types of personal information. Apparently, personal information is an essential part of user passwords and most users put certain personal information in their passwords. We believe the rate could be higher if we have more personal information at hand. However, this percentage has served its purpose properly. Then We present the top 10 password structures in Table 3 and most commonly used personal information in Table 4. As we have mentioned before, some password segment may match to multiple types of personal information and we count each matched type. Therefore with 131,389 passwords we obtain 143373 password structures. Based on Table 3 and Table 4, we have the following observations

1. The second and third structures are perfectly matched to personal information. 3 out of the top 10 structures are composed by pure personal information and 6 out of the top 10 structures have personal information segment. The dominating structures D_7, D_6, and D_8 in Table **??** still rank fairly high.

2. Birthday, name, and account name are most popular per-

sonal information in user passwords. Over 20% passwords in our dataset contain birthday or name information. On the other hand, much less people use email and ID number in their passwords. Further more, only few people include their cellphone number in their passwords.

3. Set aside personal information, digits are still dominating user passwords. Only one structure from the top 10 structures has one letter segment with minimum length (1). The result confirms that Chinese users prefer to use digits in their passwords.

4. An interesting observation is that although account name has merely half percentage as birthday and name information, the structure [ACCT] ranks highest among all structures that contain personal information. The reason behind may be that users tend to use their account names as their passwords instead of using them as part of their passwords.

2.2.4 Gender difference

We are also interested in the difference of password composition between males and females. Note that although the dataset does not have a gender column, user ID number actually has gender information (The second last digit in ID number represents gender). We found that the dataset is biased in gender, with 9,856 females and 121,533 males in it. To balance the number, we randomly select 9,856 males from the male pool and compare them with females. The average length of passwords for males and females are 8.41 and 8.51, which are quite similar. It shows that males and females do not differ much in the length of their passwords. We then apply the matching method to each of the genders. We found that 54.9% of male passwords contain personal information while 44.6% of female passwords contain personal information. Besides, the number of structures for females is 1,194, which is 8.7% more than the number for males. Therefore we conclude that generally males put more personal information than females in their passwords. It indicates females have wider ideas when it comes to password. It also implies that females have more complex passwords, and therefore maybe more secure. We list the top 10 structures for each gender in Table 5 and personal information usage in Table 6. From the tables we have the following observations:

1. The top 10 structures count 33.62% of males passwords and 28.31% of female passwords. Besides, the total number of password structures of males is also smaller. We can see passwords of males are more congregated, and therefore more predictable.

2. For males, 6 out of the top 10 structures contain personal information. Yet for females, only 3 out of top 10 structures contain personal information. It further implies that males are more likely to consider personal information when creating passwords.

3. For both males and females, [ACCT], [NAME][BD], and [BD] are three most frequent structures with personal information. However, The percentage of males are much higher than that of females. Averagely 47.3% more males are constructing their passwords following the 3 patterns than females.

Table 5: Most Frequent Structures in Different Gender

Rank	Male		Female	
	Structure	Percentage	Structure	Percentage
1	D_7	5.752%	D_6	4.890%
2	[ACCT]	5.418%	D_7	4.220%
3	[NAME][BD]	4.190%	[ACCT]	4.210%
4	[BD]	3.591%	D_8	3.256%
5	D_6	3.530%	[EMAIL]	2.678%
6	[EMAIL]	2.962%	[NAME][BD]	2.607%
7	D_8	2.861%	[BD]	2.221%
8	L_1D_7	2.851%	L_2D_6	2.079%
9	[NAME]D_7	2.353%	L_2D_7	1.704%
10	[NAME]D_6	1.978%	L_1D_7	1.684%

Table 6: Most Frequent Personal Information in Different Gender

Rank	Male		Female	
	Information Type	Percentage	Information Type	Percentage
1	[BD]	25.02%	[BD]	21.14%
2	[NAME]	23.43%	[ACCT]	15.35%
3	[ACCT]	16.27%	[NAME]	13.02%
4	[EMAIL]	7.96%	[EMAIL]	8.81%
5	[ID]	1.02%	[ID]	0.45%
6	[CELL]	0.30%	[CELL]	0.41%

4. From Table 6 we can see the percentage of each type of personal information in the passwords. Interestingly males and females are very different in the usage of name information. Males use their names as frequent as their birthday (23.43% passwords of males contain their names) while only 13.03% passwords of females contain names. We also notice that the name usage mostly contribute the 10% difference in personal information usage between males and females.

5. Users seem not like applying mangling rules on their passwords. We notice that the several most frequent password structures are either pure personal information (such as $[BD]$) or strings (such as D_6) that do not relate to any of the personal information. Structures like $[NAME]D_7$ are less likely to appear in user password.

In conclusion, passwords of males are generally composed of more personal information, especially names of the users. The sparsity of passwords of males is lower. We believe insightful analysis on the difference between male and female passwords is interesting since it provides insight for both password cracking and protection. An attacker may leverage personal information at hand to conduct faster attack while normal users can avoid popular password composition to secure their passwords.

2.3 Service Information

[] has an interesting idea that using service information to crack user passwords. However, the idea has not been carried out to an experiment. We have shown the importance of personal information in user password, we are also interested in how important the service information can be in user passwords. By service information we mean the information of service provider. For example, the famous "Rockyou" dataset is leaked from a website *www.rockyou.com*, the service information could be "rockyou". In our case, the service information is "12306". It is reasonable for a user to add service information to their passwords to keep his/her

Table 7: Service Information in Passwords

Dataset	Password Amount	Service info Amount	Percentage
Rockyou	28,688,782	8823	0.03%
Tianya	24,087,768	23086 [1]	0.06%
PHPBB	184,389	1411	0.8%
12306	131,389	4	0.003%
MySpace	37144	51	0.1%

passwords different in each site. The approach well balance password security and memorability. Therefore we wish to verify whether service information is as important as personal information. As 12306 is just a medium-sized Chinese dataset, the result may not be very representative. Therefore, we would like to verify more datasets, including Tianya dataset and Rockyou dataset, etc. in each of the datasets, we try to search the service information in the passwords. The result is shown in Table 7.

As shown in Table 7, we can see indeed some users are using service information in their passwords. However, the portion is quite small. such small percentage indicates that though adding service information in user passwords may maintain good memorability and security, currently few users are using such method to construct their passwords. Therefore, we concluded that only few users include service information in their passwords. the method proposed by [] that uses service information to crack passwords may not bring much improvement over state-of-art technique.

2.4 Ethical Consideration

We do realize that studying leaked datasets involves much ethical concern. Like many other works, we only use this dataset for researching purpose. We will not expose any user personal information in analyzed datasets.

3. CORRELATION QUANTIFICATION

In Section 2.2.3 we have shown the personal-information-involved password structure distributions and how much does each type of the personal information count in all passwords. These results helped us to draw interesting 1)qualitative conclusions such as users like to use their personal information in their passwords and 2)quantitative conclusions such as birthday appears in 24.63% passwords, which is the highest percentage among the 5 types of personal information. However, until this point we do not have a clear idea about how much does personal information correlate to passwords in a quantitative way. Although we have shown useful numbers between each type of personal information and passwords (such as birthday appears in 24.63% passwords), these numbers are not proper to describe the passwords. Yet we seek a more comprehensive method to quantify the correlation between passwords and personal information. Thereby we create Coverage to quantify the correlation.

3.1 Coverage

Coverage is a useful metric to describe how close are personal information and passwords. It is used on every individual user in the dataset. The average Coverage also reflects correlation for the dataset. We will show the computation of coverage and give a detailed example in the following section to illustrate how Coverage works. Besides, we also explain why Coverage is a good metric to describe the correlation between password and all personal information available.

3.1.1 Computation Method

To compute coverage, we take password and personal information in terms of strings as input. And we use a sliding window approach as in data transmission protocols to compute Coverage. We maintain a dynamic-sized window sliding from the beginning to the end of the password. The initial size of the window is 2. If the segment in the window matched to a certain type of personal information, we enlarge the window by size of 1. Then we try again to match the segment in the window to personal information. If a match is found, we further enlarge the window size until matches no longer exist. At this point we slide the window by one symbol and reset the window size to be the initial size. In the mean time of sliding, we maintain an array of same length as the password. This array is called tag array. Tag array is used to record the length of each matched password segment. For example, a tag array may be [4,4,4,4,0,0,2,2]. This array implies that the first 4 password symbols match certain type of personal information. the 5th and 6th symbols have no match and the last 2 symbols also match certain type of personal information. It does not matter which type do first 4 symbols and last 2 symbols match. When we eventually slide window through the password string thoroughly, the tag array is used to compute the Coverage metric. Coverage is computed as the sum of squares of matched password segment length divided by the square of password length. Mathematically

$$CVG = \sum_{i=1}^{n}(\frac{l_n^2}{L^2}) \qquad (1)$$

in which n means the number of matched password segment, l_n means the length of corresponding matched password segment, and L means the length of the password. We show the algorithm that is used to compute Coverage in Algorithm 2. Note that differently from the matching algorithm introduced in Section 2.2.2, we aim to keep the matching for Coverage simpler in order to keep it universally applicable. Namely we try to keep ad-hoc processing on each type of personal information minimum. Toward this end, we do not include various composition of birthday. Cases like DD+MM+YYYY are ignored and we keep only the only case of YYYY+MM+DD, which conforms to Chinese conventions. However, we keep full name and name initials to match names because name initials are very common in Chinese passwords. Besides, only little processing is acceptable in computation of Coverage. As a conclusion, only names have 2 permutations, other personal information keep their original format. A match is found if a password segment is a substring of any of the personal information.

We illustrate a simple example in Figure 1. In this example, we take a user named Alice, who was born in Aug. 16, 1988. Her password happens to be "alice816". If we apply the algorithm in Section 2.2.3, the structure of this password will be [NAME][BD]. Apparently her password is quite related to her personal information. To quantify this relation,

[1] We found 44,227 passwords in Tianya dataset is "111222tiany". It does not make much sense for so many user using such same password so we doubt they are mostly sybil accounts. Therefore the duplications are removed in our analysis

Algorithm 2 Compute Coverage

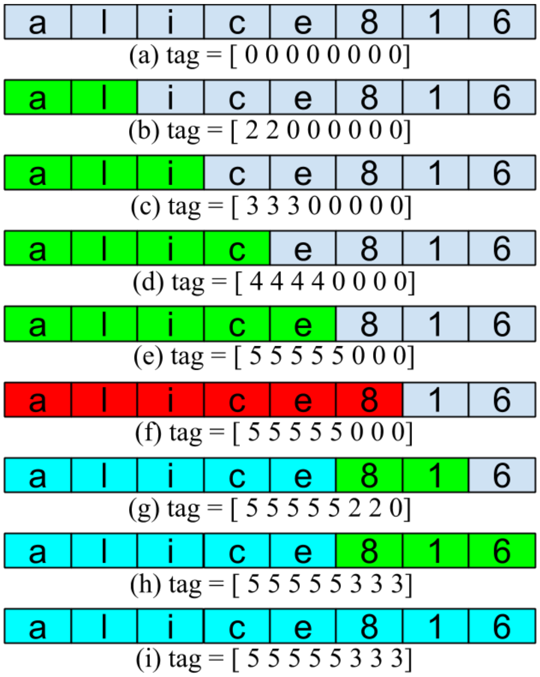
```
procedure CVG($pwd$,$infolist$)
    $windowsize \leftarrow 2$
    $pwdlen \leftarrow$ len($pwd$)
    $matchtag \leftarrow [0]*pwdlen$
    $matchmark \leftarrow 0$
    $cvg \leftarrow 0$
    while $windowsize \leq$len($pwd$) do
        $passseg \leftarrow pwd[0 : windowsize]$
        if $passseg$ = substring of $infolist$ 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
```

we follow Algorithm 2 to compute Coverage for her in each step as shown in Figure 1. (a) The password is prepared for a window to slide through. Note the personal information include ”alice” as name and ”19880816” as birthday. The tag array is initialized as [0 0 0 0 0 0 0 0]. (b) Window size is initialed to 2 so the first 2 symbols in the password are covered. ”al” is a substring of alice’s name, so 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]. (c)-(e) Window keeps growing because matches are continuously found. The tag array also keeps updating. (f) The window now covers ”alice8”, which is not a substring of ”alice” or ”19880816”. Therefore, window size is rest to 2 and tag array remains unchanged. (g) The window of size 2 now covers ”81”, which is a substring of birthday, again we extend the window by 1 and update tag array to [5 5 5 5 5 2 2 0] (h) After window grows, ”816” is also found a match. Tag array is updated to [5 5 5 5 5 3 3 3]. The window does not grow or slide any more because it has reached the end of the password. (i) All symbols have settled. Tag array can now be used to compute Coverage. Based on Equation 1, the coverage is computed as

$$\begin{aligned} CVG &= \sum_{i=1}^{2}(\frac{l_n^2}{L^2}) \\ &= \frac{5^2+3^2}{8^2} \\ &= 0.52 \end{aligned} \tag{2}$$

In order to show that Coverage is a useful and reasonable metric, we present 4 features that Coverage has and why we need these features.

Figure 1: Coverage - An Example



(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]

Grey boxes hold unvisited password symbols. green and red boxes denote that the symbols inside are covered by the sliding window. Blue boxes denote the symbols inside have been settled (window stops extending).

1. By the nature of Equation 1, Coverage ranges from 0 to 1, in which 0 means no personal information is matched to any password segment and 1 means the entire password is matched to only one type of personal information.

2. Coverage can be universally applicable for its simplicity. Any kind of personal information can be formatted as a string, check whether or not a password segment is a substring of the string is straightforward. Besides, a few variations of personal information strings may be applied to enhance the completeness and accuracy (like names in our application). Therefore, even certain types of personal information are missing or extra types of personal information are available, Coverage can consistently serve its purpose.

3. Coverage can reflect the length of personal information over the length of password. We need this feature because longer matched password segment naturally indicates stronger correlation. From Equation **??**, for a matched password segment, the longer the segment is, the larger the numerator will be. It is obvious Coverage measures the correlation strength.

4. For personal information segments of same length, Coverage distinguish the composition of the segments. The more complex the composition is, the lower the metric will be. We argue that the correlation between password and personal information is not growing linearly. That is to say, for a given password of length L, a matched segment of length l has stronger password than 2 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 stronger correlation than 2 matched segments of length 2 and 4. We desire this feature because shorter segments usually involve wrong match (coincidence). Since it is very hard to differentiate a real match and a coincidence match, we would like to minimize the effect of wrong matches by taking squares of the matched segments to compute the Coverage.

3.2 Result

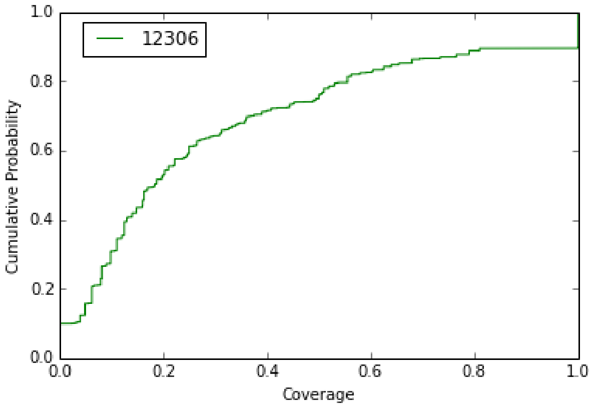
We apply Algorithm 2 on every user of 12306 dataset. The result is shown as a cumulative distribution graph in Figure 2. From the figure we have the following observation

1. Around half of the users have Coverage higher than 0.2, implying that a significant portion of user passwords have relatively high correlation to their personal information.

2. 9.9% users have 0 Coverage. 0 Coverage indicates there is no possible correlation between the users' passwords and their personal information.

3. 10.5% users have Coverage of 1, which means that 11.9% passwords are perfectly matched to exactly one type of personal information. However, Only 13.4% users have Coverage higher than 0.5 and lower than 1.0.

Figure 2: Coverage distribution



Besides, the average Coverage for 12306 dataset is 0.309. We also compute the average Coverages for male and female groups that are used in Section 2.2.4, in which we concluded that male users are more likely to include personal information in their passwords. The average Coverage for male group is 0.314 and the average Coverage for female group is 0.269. Our result indicates that the correlation between password and personal information for male users is higher. It complies with our previous conclusion. Therefore, we believe Coverage works well to determine the correlation between passwords and personal information.

4. PERSONAL-PCFG, AN INDIVIDUAL ORIENTED PASSWORD CRACKER

We have presented abundant findings regarding user personal information and their passwords, as well as a metric to quantify their correlation. Naturally we wonder how are these findings useful. And an obvious answer is that given there is significant correlation between user password and personal information, we can make password cracking much faster. Therefore we create Personal-PCFG, which relies on the idea of PCFG approach created by M. Weir[]. However, unlike original PCFG method, Personal-PCFG is an individual-oriented password cracker. It aims to crack the password of one or a small group of victims.

4.1 Attack Scenarios

In traditional offline password attacks, attackers usually steal hashed passwords from victim systems, and try to find out the unhashed values of these passwords. As hash process cannot be simply reversed, the most usual solution is by brute force, namely guessing and verifying passwords. A significant amount of works focus on finding efficient ways to enumerate password guesses[][][][], among which PCFG method is one of the best. One guess is verified by hashing the guess and comparing it to each of the hashed values. Statistically high-probability password guesses can usually match many hashed values and thus are usually tried first.

However, Personal-PCFG does not try to break the whole dataset. It Instead aims to break only a small number of person's passwords quickly. Attackers need to know certain amount of personal information about the targets. The attackers 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, target personal information is rather easy to obtain – The attackers might either know some victim personal information or can easily get personal information by searching online social networks.

Our method can benefit both online or offline attacks. For offline attacks, the attacker has hashed database in hand. Personal-PCFG is much faster in guessing out the correct password than conventional methods, therefore saving time for attacks. As for online attack, the attacker does not have a hashed database. Instead the attacker tries to directly log in the real systems. It is considered more difficult than offline attacks because websites usually have restrictions on log in attempts for a given period of time. If the quota has been reached without inputting the correct password, the target account may be locked either for some time or permanently unless certain actions are taken (For example, call the service provider). Therefore, online attacks need to be very fast – within at most tens of guesses the password should be cracked out.

4.2 A revisit of PCFG

Personal-PCFG method is based on the idea of PCFG method. Before we introduce Personal-PCFG. We revisit the basic principle of PCFG method. PCFG method preprocesses passwords by their structures. From the pre-process base structures such as "$L_5D_3S_1$" are generated for each of the passwords. Then PCFG method substitute the "D" and "S" segments using segments of same length learned from the training set. These substitute segments are ranked by probability of occurring in training set. Therefore higher probability segments will be tried first. For example, "123" is a high probability candidate for D_3 while "691" may not be a high probability candidate. As a result, one base structure may have many substitutions, for example, "$L_5D_3S_1$" can have "$L_5$123!" and "$L_5$691!", etc as its substitutions. These new representations are called pre-terminal structures. No ”L” segment is substituted because the space of alpha strings is too large to learn from the training set. These pre-terminals are ranked from high probability to low probability. And finally ”L” segments are substituted using a dictionary. Besides the basic idea, PCFG method also carries an efficient algorithm to enumerate passwords from high probability to low probability on the fly, which will also be used in Personal-PCFG. Therefore PCFG method generates password guesses in descending probability order. These guesses are hashed to compare with the values in password databases. Since PCFG method output statistically high probability password first. It reduces the guessing number of traditional dictionary attacks significantly.

4.3 Personal-PCFG

Personal PCFG leverages the basic idea of PCFG method. Beside ”L”, ”D”, and ”S” symbols, it adds more semantic symbols to PCFG method. These additional symbols include ”B” for birthday, ”N” for name, ”E” for email address, ”A” for account name, and ”I” for id number. More semantics makes Personal-PCFG more accurate in predicting passwords. To make Personal-PCFG work, additional matching phase and adapt-substitution phase are added to the original PCFG method. Personal-PCFG works as following.

4.3.1 Matching

Given a password string, we first try to match it or a substring of it to personal information. The algorithm used is the same as that introduced in Section 2.2.2. However, this time we also record the length of the matching segment. We replace the matched segments in password with corresponding symbols and length. Unmatched segments will remain unchanged. Again, We take Alice, who was born in Aug. 16 1988, as an example. This time we assume her password is ”helloalice816!”. Matching phase will replace ”alice” with ”N_5” and ”816” with ”B_3. The leftover is kept unchanged. Thereby the outcome of this step is ”$helloN_5B_3$!”

4.3.2 Processing

In this step we simply run the processing routine of original PCFG method on the output of last step. The segments that are matched to personal information are not processed in this step. The output in this step is a base structure for Personal-PCFG. Now the password is fully described by symbols of Personal-PCFG method. Our example password ”$helloN_5B_3$!” will update to ”$L_5N_5B_3S_1$”.

4.3.3 Guess Generation

Like original PCFG method, we replace ”D” and ”S” symbols with actual strings learned from the training set in descending probability order. ”L” symbols are also replaced with words from a dictionary. We use the ”NEXT” function in [] to output the results on the fly so we do not need to wait all possibilities for guesses are calculated and sorted. The results will be kept outputting for next step. Note that we do not replace any symbols for personal information so the guesses are still not in terminal status. We do not handle personal information in this step because personal information for each user is likely to be different. Personal information symbols can only be substituted when the target is specific one user. In this step our base structures generate pre-terminals, which are partial guesses that contain part of actual guesses and part of Personal-PCFG symbols. The example ”$L_5N_5B_3S_1$” is now ”$helloN_5B_3$!”. if ”hello” is the first 5 symbols long string in the input dictionary and ”!” has highest probability of occurring for 1 symbol long special character strings in the training set. Note that for ”L” segments, every words of same length has 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.

4.3.4 Adapt-substitution

The result of last step can be applied to any target user. However, this will no longer be true in this step because the guesses will be further substituted with personal information, which are specific to only one target user. Personal information symbols are replaced by corresponding personal information of same length. If there are multiple candidates of same length, every combination is output for trial. In our example guess $helloN_5B_3$!, N_5 will be directly replaced by ”alice”. However, B_3 has many candidate segments. Any length 3 substring of ”19880816” can be a candidate. Therefore every substring will be output. The guesses are ”helloalice198!”, ”helloalice988!”, . . ., ”helloalice816!”. We know the final try is exactly the password of Alice. Note that on the opposite of having multiple candidate, not all personal information segments can be replaced because same length segments may not always be available. For example, a $helloN_6B_3$! is not suitable for Alice for her name is at most 5 symbols long. In this case no guess from this pre-terminal structures will be output for Alice.

4.4 Cracking Result

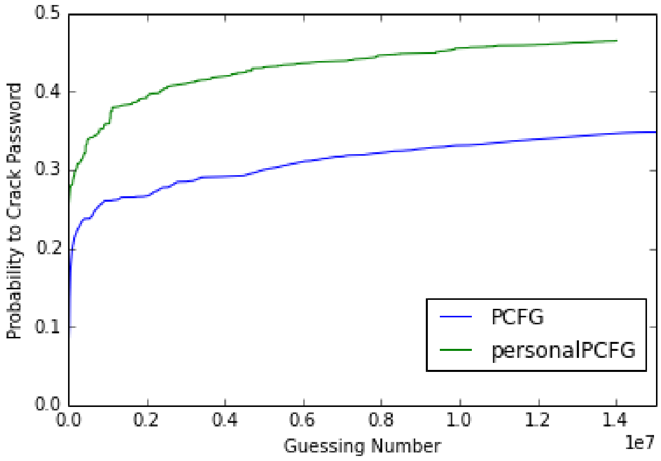
We use 12306 dataset in both Personal-PCFG and original PCFG method. 12306 dataset has 131,389 users. We use half of them as training set, and the other half as testing set. Note that we realize There is a major difference between the target of Personal-PCFG and PCFG – Personal PCFG generate personalized guesses, which is only for specific users while original PCFG generate guesses for every user and we aim to show the efficiency of the 2 methods when attacking only one person. To eliminate unnecessary errors, we do not arbitrarily choose one or several persons for experiment. Instead we use Personal-PCFG to attack the whole testing set to show the grand picture. Namely we use each user in the testing set as an individual experiment and show the average case.

Besides, both of the methods involve selection of dictionaries for the ”L” segments. Dictionaries are a vital part for password cracking. [] proved that vocabulary in password datasets are more closed to each other than to an ordinary dictionary therefore using dictionaries collected from other dataset is more efficient. To eliminate the effect of unfair dictionary selection, we use ”perfect” dictionaries in both of the methods. ”Perfect” dictionaries are dictionaries we collected directly from the testing set, in which any string is useful and no string is uncovered. Therefore a perfect dictionary is guaranteed to find correct alpha strings efficiently.

The result is shown in Figure 3, in which we compute the probability of a password that can be cracked given the guessing number. We found that Personal-PCFG works substantially better than original PCFG method. Some interesting observations are listed in following.

1. Both PCFG and Personal-PCFG can crack passwords quickly when they just start. However, the line become

Figure 3: Coverage distribution



flat when the guessing number is growing large. This is normal because both methods try high probability guesses first.

2. Personal-PCFG works much better than PCFG when only one person is the target. With 10 million guesses Personal-PCFG achieves the probability that is achieved with over 300 million guesses by PCFG method. That is to say, Personal-PCFG can crack password much much faster than PCFG method.

3. Personal-PCFG covers more passwords than PCFG method. This is because personal information provides rich personalized strings which may not be covered in the training set.

5. PASSWORD PROTECTION

After showing that user passwords are highly guessable, we wonder how could users protect their passwords. It is long said that there is trade-off between password memorability and password security. A secure password

6. RELATED WORKD

Left blank

7. ACKNOWLEDGMENTS

8. REFERENCES

APPENDIX

A. FULL LIST OF BIRTHDAY AND NAME IN MATCHING ALGORITHM