ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-7605-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICASSP39728.2021.9414886

IMPROVED PROBABILISTIC CONTEXT-FREE GRAMMARS FOR PASSWORDS USING WORD EXTRACTION

Haibo Cheng *Wenting Li* *Ping Wang*[*] *Kaitai Liang*

Peking University
{hbcheng,wentingli,pwang}@pku.edu.cn

Delft University of Technology
Kaitai.Liang@tudelft.nl

ABSTRACT

Probabilistic context-free grammars (PCFGs) have been proposed to capture password distributions, and further been used in password guessing attacks and password strength meters. However, current PCFGs suffer from the limitation of inaccurate segmentation of password, which leads to misestimation of password probability and thus seriously affects their performance. In this paper, we propose a word extraction approach for passwords, and further present an improved PCFG model, called *WordPCFG*. The WordPCFG using word extraction method can precisely extract semantic segments (called word) from passwords based on cohesion and freedom of words. We evaluate our WordPCFG on six large-scale datasets, showing that WordPCFG cracks 83.04%–95.47% passwords and obtains 12.96%–71.84% improvement over the state-of-the-art PCFGs.

***Index Terms*—** Word extraction, password, probabilistic context-free grammar.

1. INTRODUCTION

Passwords have been widely used by Internet users and service providers as the primary method for authentication [1]. However, due to poor generation habits, passwords can be easily predicted and cracked by attackers [2, 3]. To precisely describe password habits (a.k.a. password distributions), two types of probability models have been introduced: one is *template-based model* (also called PCFG model) [2, 4] and the other is *char-based model* [3, 5]. These models are further used for a lot of password applications, e.g., password strength meters [6, 7], password enhance advisors [8], and honey password vaults [4, 9].

Template-based model is based on an intuitive idea that *users habitually choose several different meaning segments and group them together as a password*. Following the idea, the model uses a probabilistic context-free grammar (PCFG) to parse the (semantic) segments in passwords and assigns a probability to each segment as well as the template; the password probability is calculated as the product of the probabilities of these segments and the template. But the inaccurate segmentation leads to misestimation of password probability. For example, "jordan23" is the password consisting of Michael Jordan's name and his jersey number. Current models (e.g., [2, 4]) divide the password to two independent segments and fail to capture the relationship between them.

In contrast, a char-based model directly captures the password generation char by char and assigns a conditional probability for each char given previous chars; the password probability is calculated as the product of the probabilities of the chars. But this type of model (e.g., [3, 5]) cannot reveal the semantics in passwords and further may be not easily improved.

1.1. Our Contributions

We propose a method to precisely extract semantic segments (called words) in passwords, based on our defined cohesion and freedom of words. This word extraction does not require any extra dictionaries, but can extract keyboard patterns (e.g., "1q2w3e"), language words or phrases (e.g., English words "password", Spanish phrases "teamo"), hybrid words (e.g., "jordan23", "welcome2") and etc. from passwords. Due to its good performance, we reveal some interesting password habits. Further, with this word extraction, we can precisely parse the generation of passwords and propose an improved PCFG model, *WordPCFG*. WordPCFG significantly improves the accuracy of capturing password distributions. With six real-world datasets within the total amount of 77.7 million passwords, we evaluate the WordPCFG on the efficiency for password guessing attacks. The experimental results show that WordPCFG can crack 12.96%–71.84% more passwords than the state-of-the-art PCFGs.

2. BACKGROUNDS

2.1. Probabilistic Context-free Grammar

A probabilistic context-free grammar is a five tuple $G = (S, N, \Sigma, R, p)$ where N is a set of non-terminals, $S \in N$

[*]Corresponding author. Ping Wang is also with Key Laboratory of High Confidence Software Technologies (PKU), Ministry of Education, China.

This research is supported by National Key R&D Program of China (2017YFB1200700) and by European Union's Horizon 2020 research and innovation programme under grant agreement No. 952697 (ASSURED).

is the start symbol, Σ is a set of terminals, R is a set of production rules $N \rightarrow (N \cup \Sigma)^*$, and p is a function mapping each rule to a probability. It requires that $\sum_{\alpha \in (N \cup \Sigma)^*} p(X \rightarrow \alpha) = 1$ for any non-terminal $X \in N$.

Weir et al. [2] propose the first PCFG model for password. We denote it as PCFG$_W$ in this paper. PCFG$_W$ divides a password into three types of segments, including letter segment L, digit segment D, and special-symbol segment S with subscript denotes the segment length. The construction of these segment types is called template for the password. For example, "password123" is constructed by a L_8 segment "password" and a D_3 segment "123", and its template is L_8D_3. Then the generation of password can be modeled by a rule of template ($S \rightarrow L_8D_3$) and the rules of segments ($L_8 \rightarrow$ password, $D_3 \rightarrow 123$). The probability of the password can now be defined by the PCFG, i.e., the product of probabilities of rules.

PCFG$_W$ leverages an external dictionary for L segments. Ma et al. [3] slightly improve PCFG$_W$ by training L segments with password datasets. The improved model is denoted as PCFG$_M$ in this paper. Lately, Chatterjee et al. [4] improve the segmentation in PCFG with external dictionaries of English words, keyboard patterns, names and etc. In this way, their PCFG (denoted as PCFG$_C$) models the password generation more precisely along with richer rules.

2.2. Password Guessing Attack

Password guessing attack is the foundational research topic in the context of password security and attracts a lot of attentions, e.g., [2, 3, 10, 11]. In the classic scenario of offline password guessing attacks, an attacker steals a file containing the hash values of passwords and then tries to recover the plaintext passwords by: 1) generating a dictionary of password guesses and 2) calculating the hash value for each guess. If a guess-hash-value matches the stolen one, then the attack is successful.

The efficiency of the offline attack depends on the order of guesses in the password dictionary. In the descending order of (real) password probability, the attack achieves the best efficiency, i.e., recovering the maximum number of passwords under a given amount of calculations/guesses. In the literature (e.g., [2, 3]), password models are usually used in the offline guessing attack by generating password guesses in the descending order of model-estimated probability. The more precise probability a password model can provide, the more efficient the attack (built on the model) can be.

3. OUR PROPOSED METHOD

We propose a method to extract semantic segments from passwords, and further design an improved PCFG, *WordPCFG*, based on our extraction.

3.1. Word Extraction for Password

In natural language processing, word extraction aims to extract (unknown) words in some languages without a word delimiter, e.g., Chinese and Japanese. The extracted words can be used for word segmentation [12, 13, 14, 15]. Inspired by a Chinese word extraction approach [16], we design a method that can be used in the context of password to extract semantic segments (also called word for consistency). Our extraction tells whether a string is a word depends on its *cohesion* and *freedom*.

Cohesion is the evaluation of a string's internal association. We define cohesion via point-wise mutual information (PMI). PMI is a good criterion to indicate the correlation of two (sub-)strings s_1 and s_2, which is defined as

$$\mathrm{PMI}(s_1; s_2) = \log \frac{p(s_1 || s_2)}{p(s_1) \cdot p(s_2)}, \qquad (1)$$

where $s_1 || s_2$ represents the concatenating of s_1 and s_2. Subsequently, the cohesion of a string s is defined as:

$$\mathrm{Coh}(s) = \min_{s_1 || s_2 = s} \mathrm{PMI}(s_1; s_2). \qquad (2)$$

Here, s_1, s_2 can be an arbitrary segmentation for s.

Freedom is the evaluation of a string's independence from its context, which also shows the degree of free using of the string. Intuitively, a string tends to be a word, if it has as many different adjacent characters as possible. This can be well captured by Shannon Entropy. We define the freedom of string s as follows:

$$\mathrm{Fdm}_l(s) = -\sum_{c \in \Sigma} \Pr(c || s) \cdot \log \Pr(c || s), \qquad (3)$$

$$\mathrm{Fdm}_r(s) = -\sum_{c \in \Sigma} \Pr(s || c) \cdot \log \Pr(s || c), \qquad (4)$$

$$\mathrm{Fdm}(s) = \min_{x \in \{r, l\}} \mathrm{Fdm}_x(s). \qquad (5)$$

Here, the (left and right) adjacent character c can be an arbitrary character in the alphabet Σ. For passwords, Σ (usually) is the set of all printable ASCII characters.

With the above definitions of a string, bigger cohesion value means tighter internal association in the string, more specifically, it should be treated as a word rather than a combination of two words; bigger freedom value means freer using of it, i.e., it can be used in conjunction with many other words. Therefore, we can use cohesion and freedom to precisely extract words from passwords. We set two thresholds T_c and T_f for cohesion and freedom, respectively. If string s satisfies that $\mathrm{Coh}(s) \geq T_c$ and $\mathrm{Fdm}(s) \geq T_f$, it will be extracted as a word.

3.2. WordPCFG

With the word extraction, we introduce a new non-terminal type, word denoted as W, to the original PCFG, and propose

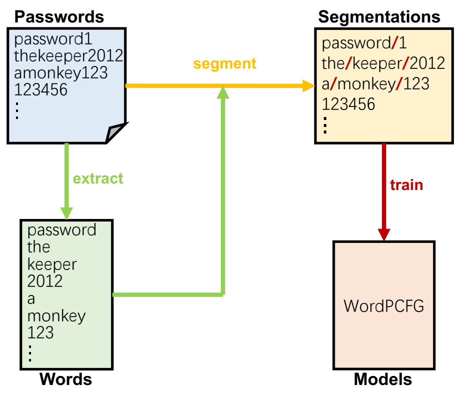


Fig. 1: The training process for WordPCFG

an improved version called WordPCFG. More specifically, the non-terminal set N of WordPCFG includes the start symbol S, the W segment variables W_i, the L segment variables L_i, the D segment variables D_i, and the S segment variables S_i ($l_{\min} \leq i \leq l_{\max}$, where $l_{\min}$ and $l_{\max}$ are the minimum and maximum lengths of passwords, respectively).

As shown in Fig. 1, to train WordPCFG, we first extract words from passwords, then use the dictionary of words to segment passwords (i.e., parsing in PCFG), and finally estimate the probabilities of rules (i.e., segments and templates) in PCFG from the segmented passwords.

1. In word extraction, we empirically set $T_c = 0.01$ and $T_f = 1.0$ according to their good performance (for password guessing).
2. In word segmentation, we recognize words in a password by longest matching, classify the words according to their length (e.g., "password" is a W_8 segment), and parse the rest parts as PCFG$_W$.
3. In probability estimation, we leverage maximum likelihood estimation (MLE) as PCFG$_W$.

4. EXPERIMENTS AND RESULTS

4.1. Password Datasets

We here use six real-world password datasets which are widely used in password research, e.g., [2, 3, 10, 11]. We argue that using them in the experiment only is ethical, as it will bring no harm to users. The experimental results will just benefit the research on password protection.

The statistics of the datasets are given in Table 1. The first three datasets are from English-speaking countries, and the rest are from China. The Rockyou dataset is the largest clear-text password dataset used extensively in password research. It includes over 32 million passwords disclosed from social website Rockyou. Clixsense, which pays people to take online surveys, experienced a password breach in which about 2.2 million clear-text passwords were exposed by the hackers. Website hosting service 000Webhost suffered from a data breach by a malicious exploit of system bugs. There are over 15 million clear-text passwords in the disclosed data dump. The CSDN dataset contains about 6 million user accounts from the Chinese Software Developer Network which is a popular technology portal and community of IT from China. The Dodonew dataset includes about 16 million password entries from a paid online gaming website from China. The Duowan dataset includes about nearly 5 million passwords from Duowan, a popular gaming portal from China.

The six datasets with various services and users provide approximately 77.7 million passwords. Our experiment thus can show a scalable and comprehensive view of password habits in the real-world applications.

Table 1: Password dataset information

Dataset	Unique	Total	Service
Rockyou	14,326,970	32,581,870	Social Network
000Webhost	10,583,709	15,251,073	Web Hosting
Clixsense	1,628,471	2,222,046	Online Surveys
CSDN	4,037,605	6,428,277	IT Community
Dodonew	10,135,260	16,258,891	Online Gaming
Duowan	3,119,060	4,982,730	Gaming Portal

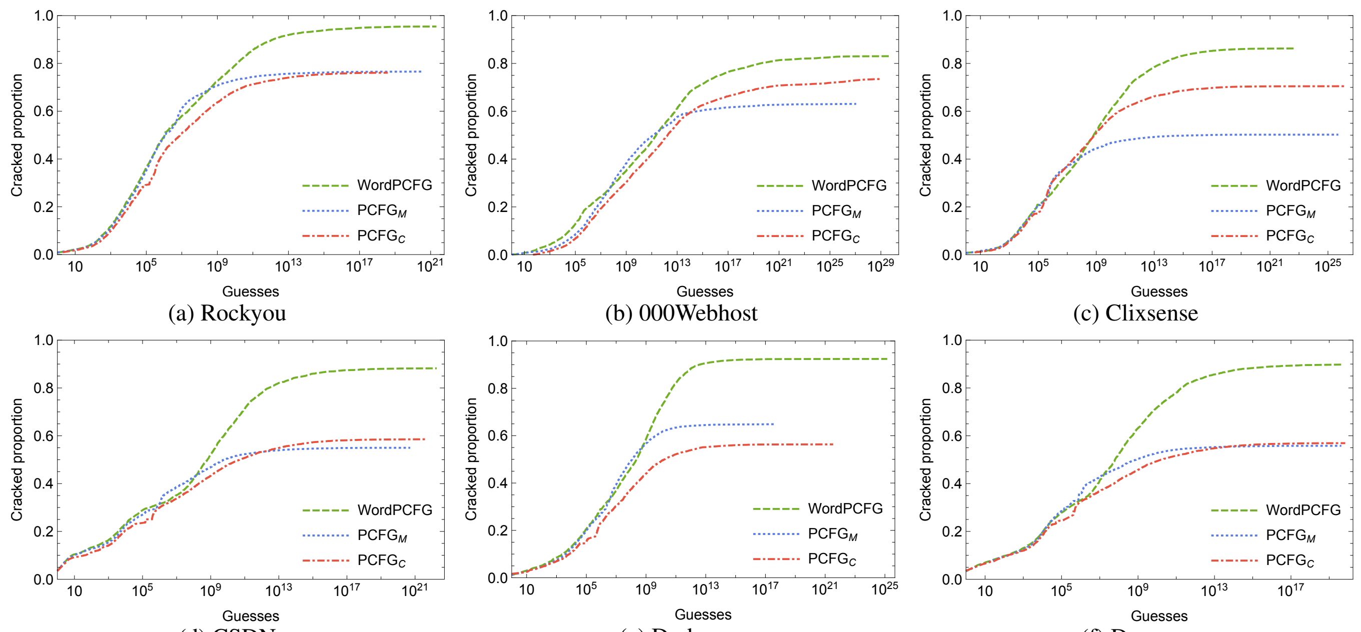
Table 2: Extracted words from passwords via our method

Type	Examples
Keyboard pattern	qwerasdf 1q2w3e zxcvbn 1qaz 123456
English word	superstar skateboard lucky dragoon
Chinese pinyin	woaini woshi mima baobei haha
Name	steven wangming
Phrase	iloveu teamo byebye mylife howareyou
Hybrid	kobe24 jordan23 welcome2 4ever

4.2. Word Extraction

We perform word extraction on the password datasets. Table 2 presents some typical words, including keyboard patterns, English words, Chinese pinyin, names, phrases, and hybrid words. From these words, we explore some interesting password habits. For instance, some users prefer to simplify their passwords via leet, such as "iloveu" replacing "you" with "u", "welcome2" changing "to" with 2 and "4ever" using 4 for "for". Someones choose a word consisting of a letter string and a digital string with a special meaning, e.g., "kobe24" and "jordan23".

Note that the external dictionaries used by current PCFG models are usually collected from natural language texts which may have a very different distribution to password. This gap leads to the inaccuracy of password segmentations, and further causes the inaccuracy in capturing password distribution. This will seriously affect the effectiveness and efficiency of the password attacks using the models. In contrast, our method can extract these words directly from the



(a) Rockyou (b) 000Webhost (c) Clixsense

(d) CSDN (e) Dodonew (f) Duowan

Fig. 2: The performance of WordPCFG for password guessing attack

Table 3: Cracked proportions under the given guess numbers

Dataset	Model	10^3	10^9	10^{12}	10^{15}	Final
Rockyou	WordPCFG	11.79%	72.61%	89.52%	93.82%	95.47%
	PCFG$_M$	11.03%	70.69%	75.15%	76.18%	76.58%
	PCFG$_C$	9.92%	63.76%	72.82%	75.47%	76.12%
000Webhost	WordPCFG	4.22%	35.12%	54.67%	71.49%	83.05%
	PCFG$_M$	2.43%	38.19%	53.47%	60.37%	63.09%
	PCFG$_C$	1.48%	30.28%	48.53%	62.56%	73.52%
Clixsense	WordPCFG	6.22%	51.75%	74.47%	83.30%	86.30%
	PCFG$_M$	5.98%	44.47%	48.70%	49.79%	50.22%
	PCFG$_C$	5.42%	50.98%	64.06%	68.64%	70.48%
CSDN	WordPCFG	16.42%	52.25%	77.96%	85.97%	88.23%
	PCFG$_M$	15.37%	46.89%	53.27%	54.65%	54.98%
	PCFG$_C$	14.18%	43.33%	53.26%	57.30%	58.55%
Dodonew	WordPCFG	8.52%	58.63%	88.39%	91.99%	92.42%
	PCFG$_M$	7.48%	56.73%	64.14%	64.76%	64.88%
	PCFG$_C$	6.96%	44.01%	53.89%	56.02%	56.35%
Duowan	WordPCFG	13.03%	63.22%	83.17%	88.33%	89.81%
	PCFG$_M$	12.37%	50.38%	54.86%	55.64%	55.81%
	PCFG$_C$	11.98%	45.76%	53.50%	56.25%	56.94%

password datasets. Based on this extraction, our WordPCFG will not have the issue and significantly improve accuracy.

4.3. Performance on Password Guessing

To evaluate the model accuracy, we use WordPCFG to carry out a password guessing attack as in [2, 3] with the Monte Carlo method [17]. We choose the state-of-the-art PCFG$_M$ [3] and PCFG$_C$ [4] for comparison. Specifically, for each of the datasets, we use half of it to train the PCFG models (including the word extraction for WordPCFG), and take the rest for testing (i.e., launching password guessing attacks against the passwords in the rest part).

We leverage the curve of cracked proportion vs. guess number to indicate the performance of the PCFG models as in [2, 3]. As Fig. 2 and Table 3 shown, when the guessing number is decreasing (to less than 10^8), the performance of the three models are close; but if the number climbs to 10^{10} (10^{12} for the 000Webhost dataset), our WordPCFG achieves a significant improvement as compared to others. Our experiments show that PCFG$_M$ and PCFG$_C$ cracks 50.22%–76.58% and 56.35%–76.12% passwords, respectively; meanwhile WordPCFG can crack 83.04%–95.47%, which achieves a 12.96%–71.84% improvement.

Besides, PCFG$_C$ performs worse than PCFG$_M$ on the Dodonew dataset. This may be because the external dictionaries used by PCFG$_C$ cannot apply to this dataset. Our WordPCFG with word extraction is free from this hindrance and further has better scalability in various datasets.

5. CONCLUSIONS

We propose a word extraction method to extract semantic parts from password. This extraction is based on the well-defined notions of cohesion and freedom, and does not require any external dictionaries. We further introduce an improved PCFG model - WordPCFG. The model, leveraging the dictionary of the extracted words, can provide precise segmentations for passwords. We also perform the experiments on the six real-world datasets and show that WordPCFG achieves a significant improvement on password guessing and outperforms others w.r.t. attack accuracy and scalability.

6. REFERENCES

[1] J. Bonneau, C. Herley, P. C. Oorschot, and F. Stajano, "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes," in *Proc. IEEE S&P 2012*, pp. 553–567.

[2] M. Weir, S. Aggarwal, B. D. Medeiros, and B. Glodek, "Password cracking using probabilistic context-free grammars," in *Proc. IEEE S&P 2009*, pp. 391–405.

[3] J. Ma, W. Yang, M. Luo, and N. Li, "A study of probabilistic password models," in *Proc. IEEE S&P 2014*, pp. 689–704.

[4] R. Chatterjee, J. Bonneau, A. Juels, and T. Ristenpart, "Cracking-resistant password vaults using natural language encoders," in *Proc. IEEE S&P 2015*, pp. 481–498.

[5] L. Bauer, W. Melicher, B. Ur, S. M. Segreti, and L. F. Cranor, "Fast, lean, and accurate: Modeling password guessability using neural networks," in *Proc. USENIX Security 2016*, pp. 175–191.

[6] M. L. Mazurek, S. Komanduri, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, P. G. Kelley, R. Shay, and B. Ur, "Measuring password guessability for an entire university," in *Proc. ACM CCS 2013*, pp. 173–186.

[7] M. Golla and M. Dürmuth, "On the accuracy of password strength meters," in *Proc. ACM CCS 2018*, pp. 1567–1582.

[8] B. Ur, F. Alfieri, M. Aung, L. Bauer, N. Christin, J. Colnago, L. F. Cranor, H. Dixon, P. Emami Naeini, H. Habib, *et al.*, "Design and evaluation of a data-driven password meter," in *Proc. CHI 2017*, pp. 3775–3786.

[9] H. Cheng, Z. Zheng, W. Li, P. Wang, and C.-H. Chu, "Probability model transforming encoders against encoding attacks," in *Proc. USENIX Security 2019*, pp. 1573–1590.

[10] B. Pal, T. Daniel, R. Chatterjee, and T. Ristenpart, "Beyond credential stuffing: Password similarity models using neural networks," in *Proc. IEEE S&P 2019*, pp. 814–831.

[11] D. Pasquini, A. Gangwal, G. Ateniese, M. Bernaschi, and M. Conti, "Improving password guessing via representation learning," in *IEEE S&P 2021*. https://arxiv.org/pdf/1910.04232.pdf.

[12] S. Chen, Y. Xu, and H. Chang, "A simple and effective unsupervised word segmentation approach," in *Proc. AAAI 2011*, pp. 866–871.

[13] J. Heymann, O. Walter, R. Haeb-Umbach, and B. Raj, "Iterative bayesian word segmentation for unsupervised vocabulary discovery from phoneme lattices," in *Proc. IEEE ICASSP 2014*, pp. 4057–4061.

[14] X. Wang, D. Cai, L. Li, G. Xu, H. Zhao, and L. Si, "Unsupervised learning helps supervised neural word segmentation," in *Proc. AAAI 2019*, vol. 33, pp. 7200–7207.

[15] Q. Zhang, X. Liu, and J. Fu, "Neural networks incorporating dictionaries for chinese word segmentation.," in *Proc. AAAI 2018*, pp. 5682–5689.

[16] S. He and J. Zhu, "Bootstrap method for chinese new words extraction," in *Proc. IEEE ICASSP 2001*, vol. 1, pp. 581–584.

[17] M. Dell'Amico and M. Filippone, "Monte carlo strength evaluation: Fast and reliable password checking," in *Proc. ACM CCS 2015*, pp. 158–169.