# PCFG-Based Framework for Cracking Korean Users' Passwords

Yong Heon Lim, Jong Chan Seo, and Hae Young Lee

Cheongju University, Cheongju 28503, Republic of Korea
whichmeans@gmail.com

**Abstract.** This poster presents a framework for cracking passwords from Korean-speaking users. The framework uses a probabilistic context free grammar (PCFG) to systematically analyze the passwords and to effectively generate possible password guesses based on the analysis results. By analyzing exposed and publicly available password dataset from Korean users, we have designed our PCFG to consider Korean strings and year strings since they often appeared in the dataset. We have implemented a proof-of-concept of the framework, in which multiple workers can involve in distributed password cracking.

**Keywords:** Password Guessing, Password Cracking, Textual Password.

## 1 Introduction

The continuous dominance of textual passwords in user authentication applications means that it remains important to further our understanding of user-generated passwords to improve password security [1]. However, most research efforts in understanding and guessing user-generated passwords have been focused on English- or Chinese-speaking users [2-4]. Users speaking different languages would prefer different patterns in creating their passwords [4], so that knowledge on these passwords may not help to guess Korean-speaking users' textual passwords. Recently, Chae *et al*. [5] conducted the first study on analyzing Korean-speaking users' textual passwords from "Collection #1" and discovered some semantic patterns appeared in the passwords. However, a method for guessing the passwords was not presented in their study.

In this poster, we present a framework for cracking passwords generated by Korean-speaking users, based on probabilistic context free grammars (PCFG) [2]. To understand Korean users' patterns in creating their passwords, we have analyzed exposed and publicly available passwords from Korean users. Based on our observations, keyboard walks [3], alpha (English) strings, Hangul (Korean) strings, year strings, digit strings, and special strings. The framework consists of a trainer module, a controller module, and a pool of workers. For a given password dataset, the trainer module has generated a PCFG. For a given password hash, the controller module distributes base structures of the PCFG, based on their probabilities, to workers in the pool. Distributed password cracking is then performed by the workers.

## 2 PCFG for Cracking Korean Users' Passwords

In our framework, 6 types of non-terminals – keyboard walks (*K*), alpha strings (*A*), Hangul strings (*H*), year strings (*Y*), digit strings (*D*), and special strings (*S*) – are considered. Table 1 shows examples of non-terminals and production rules in our PCFG.

**Table 1.** Examples of non-terminals and production rules in our PCFG

| Non-terminals | Description | Production rules |
|---|---|---|
| $K_n$ | Keyboard walks | $K_4 \rightarrow$ qwer \| 1qaz, $K_8 \rightarrow$ 1q2w3e4r |
| $A_n$ | Alpha (English) strings | $A_8 \rightarrow$ password \| iloveyou |
| $H_n$ | Hangul (Korean) strings | $H_6 \rightarrow$ dkssud \| dydgjs, $H_8 \rightarrow$ yongheon |
| $Y_n$ | Year strings | $Y_4 \rightarrow$ 2025, $Y_6 \rightarrow$ 202508 |
| $D_n$ | Digit (number) strings | $D_2 \rightarrow$ 26, $D_8 \rightarrow$ 20250820 |
| $S_n$ | Special strings | $S_1 \rightarrow$ @, $S_2 \rightarrow$ !! |

## 3 Framework for Distributed Password Cracking

Our framework shown in Fig. 1 consists of 3 modules: (a) trainer, (b) controller, and (c) worker pool. For a given password dataset, (a) trainer generates a PCFG using dictionaries for Korean and English languages, resulting in the production of base structures and terminals, together with their probabilities. For a given password hash, (b) controller distributes each of the base structures to a worker in (c) worker pool, based on the probability. Upon receiving a base structure, the worker generates possible password guesses using the structures and terminals then performs password cracking.

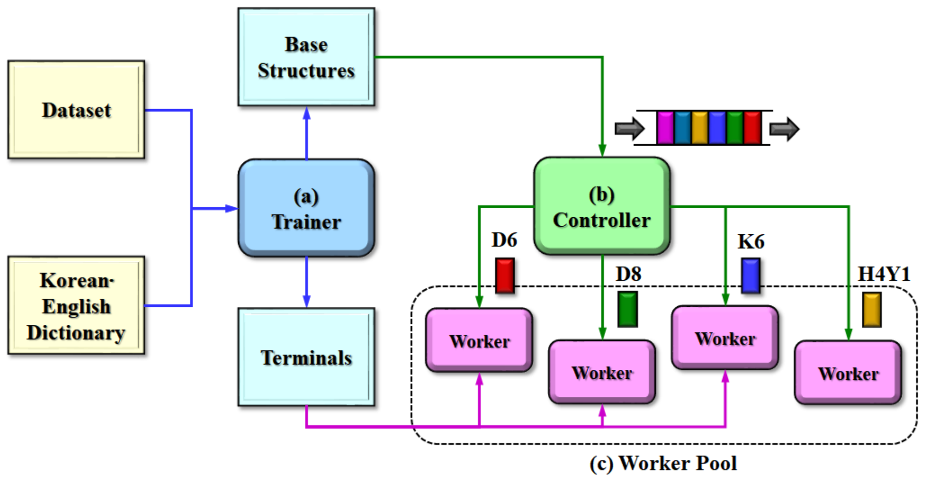

**Fig. 1.** Overview of our framework

## 3.1 Training Phase

In the training phase, the trainer module analyzes the given password dataset and generates PCFG as shown in Fig. 2. For each substring of a password in the dataset, the module first checks if the substring fits a keyboard walk. It then performs 'leet' transformations [6] to check if the transformed one exists in the English dictionary (i.e., an alpha string). If so, it replaces the substring with the transformed one and stores these two ones as terminals. For example, through 'leet' transformation, it replaces substring 'c4v3' with 'cave' and stores both 'c4v3' and 'cave' as terminals. Next, it checks if the substring is an alpha string or Korean strin. To distinguish between alpha and Korean strings is non-trivial since Korean-speaking users often choose both as their passwords. For example, a password from a Korean user may include 'dndkacave,' which is a concatenation of Korean string 'dndka' and alpha string 'cave.' Also, passwords from Korean-speaking users often include Korean names in English (e.g., 'yongheon'). To distinguish between alpha and Korean strings, it uses the dictionaries and fastText [7] as well. The next step is to check whether the substring is a 'noise' string, which consists of alphabets and but passed the previous steps (e.g., 'zv'). Noise strings are considered alpha strings. Next, it checks if the substring is a year string, such as '2025' or '202508,' since such a string often appeared in our observations. Finally, it checks if it is a digit string and then a special string.

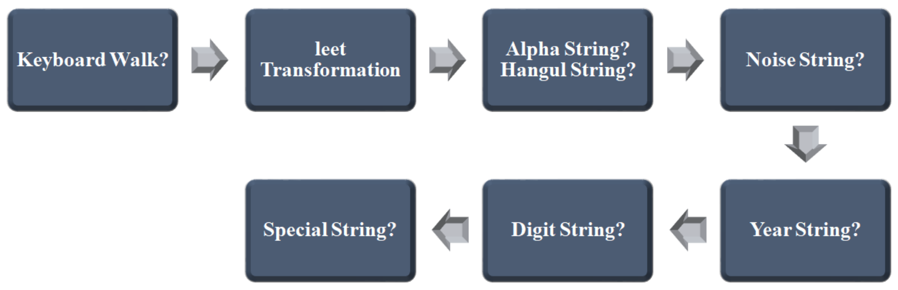

**Fig. 2.** Substring classification process in the trainer module

## 3.2 Cracking Phase

When a password hash is given, the controller module delivers base structures stored in the database to workers in the pool. Base structures with higher probabilities are delivered first. Based on the base structure received from the controller module, each of the worker uses terminals collected through the training phase to generates possible password guesses. It then performs password cracking with John the Ripper [8] or hashcat [9].

## 4 Conclusions and Future Work

In this poster, we presented a PCFG-based framework for cracking passwords generated by Korean-speaking users. By analyzing exposed and publicly available passwords from Korean users, we have designed our framework to consider keyboard walks, alpha strings, Hangul strings, year strings, digit strings, and special strings. When a password dataset is given, a PCFG is generated by the trainer modules, which also leads to the production of base structures and terminals. For a given password hash, the controller module delivers base structures to workers in the pool, each of which perform password cracking by generating possible password guesses based on the base structures and terminal collected through the training phase. A proof-of-concept version[1] of the framework has been implemented. We will evaluate the performance of the framework by comparing with existing ones. We will also combine the framework with a brute-force approach to generating all possible Korean terminals.

## References

1. Wang, Y. *et al*.: SE#PCFG: Semantically Enhanced PCFG for Password Analysis and Cracking. IEEE Transactions on Dependable and Secure Computing 22(4), 4428-4441 (2025)
2. Weir, M. *et al*.: Password Cracking Using Probabilistic Context-Free Grammars. Proc. of the 30th IEEE Symposium on Security and Privacy (2009)
3. Houshmand, S. et al.: Next Gen PCFG Password Cracking. IEEE Transactions on Information Forensics and Security 10(8), 1776-1791 (2015)
4. Li, Z., Han, W.: A Large-Scale Empirical Analysis of Chinese Web Passwords. Proc. of the 23rd USENIX Security Symposium (2014)
5. Chae, M. *et al*.: The Threat of Password Guessing Attacks Exploiting Linguistic Characteristics: A Case Study on the Korean Domains. Proc. of the 2024 Silicon Valley Cybersecurity Conference (2024)
6. Li, W. et al.: Leet Usage and Its Effect on Password Security. IEEE Transactions on Information Forensics and Security 16, 2130-2143 (2021)
7. fastText, https://fasttext.cc/, last accessed 2025/07/25
8. John the Ripper password cracker, https://www.openwall.com/john/, last accessed 2025/07/25
9. hashcat – advanced password recovery, https://hashcat.net/hashcat/, last accessed 2025/07/25

[1] https://github.com/Creeper0809/PCFGCracking