Advanced PDF Password Cracking Tool

LOHITH KUMAR REDDY BALIREDDIGARI, SRI SURYA GANESH YANAMADALA, POOJITHA NIHARIKA KOLLOJU.

COMPUTER SCEINCE

Abstract— **The Advanced Password Cracking Tool is developed in Python and supports Brute force, Dictionary, and Random attacks to crack files and hashes like PDFs. Python's ease of use and large library of modules make it well-suited for password cracking. The Agile methodology was used for development to meet evolving requirements. The tool is valuable to security experts and system administrators for testing password strength and identifying vulnerabilities. Its use of multiple cracking techniques enables it to crack a wide range of passwords.**

***Keywords— Password Cracking Tool Brute force, testing** (key words)*

I. Introduction

Python is a high-level programming language that is widely used in the field of cybersecurity due to its simplicity, versatility, and large library of modules that can be used for password cracking. The proposed tool will be capable of cracking files and hashes like PDFs, which are commonly used to store sensitive information.PDF password security is a commonly used method for securing sensitive information in digital documents. Passwords are used to protect the confidentiality and integrity of the data stored in PDF documents by preventing unauthorized access, modification, or distribution. However, if the password used to protect the PDF document is easy to guess or crack, it can become a significant vulnerability, undermining the intended security of the document. Attackers can use automated tools like password cracking software, which can systematically try out different combinations of characters, words, and phrases until the correct password is found. Easy-to-guess passwords like common dictionary words, common phrases, or personal information such as dates of birth, anniversaries, or names can be easily cracked by password cracking tools. Attackers can also use brute force techniques that try out all possible combinations of characters within a certain length and character set, eventually leading to a successful password cracking. The consequences of weak password security in PDFs can be severe, especially when sensitive information is at stake. Attackers can gain unauthorized access to sensitive information and use it for illegal activities such as identity theft, financial fraud, and corporate espionage. This can result in significant financial losses, legal liabilities, and reputational damage for individuals and organizations. Therefore, it is essential to use strong and complex passwords when setting PDF password security. Strong passwords that are long, include a mix of uppercase and lowercase letters, numbers, and symbols, and avoid common words and phrases can significantly reduce the risk of password cracking. Additionally, users can use password management tools that help generate, store, and manage strong passwords to further enhance the security of their digital documents. The tool will employ various password cracking techniques, including Brute force Attacks, Dictionary Attacks. Brute force Attacks are an exhaustive search technique that tries every possible combination of characters to find the correct password. Dictionary Attacks use a list of common words and phrases to attempt to guess the password. By combining these techniques, the tool will be able to crack a wide range of passwords. The proposed tool will be developed using Agile methodology, which is an iterative and collaborative approach to software development. Agile methodology will enable the development team to work closely with the stakeholders and respond to evolving requirements. The tool will be tested at each iteration to ensure quality and will be delivered on time and within budget. In conclusion, the proposed Advanced Password Cracking Tool will provide security experts and system administrators with a powerful tool to test the strength of passwords and identify vulnerabilities. By using Python and employing various password cracking techniques, the tool will be capable of cracking a wide range of passwords, including those used to protect sensitive information stored in PDFs. The use of Agile methodology will ensure that the tool meets the requirements and is delivered on time and within budget.

II. Literature Review

PASSWORD STORING MECHANISMS :

This paper proposes an architecture to improve the way passwords are stored on machines in order to enhance the security of passwords. It is based on the concept of steganography, which is the technique of concealing messages in text or pictures. Password cracking is a process of guessing or recovering a password from stored locations or from a data transmission system. Multiple solutions, such as using unique, complex, long and alpha-numeric passwords, have been discussed, but it is difficult to remember long and complex passwords. Steganography is different from cryptography in that it hides messages inside a cover medium, while cryptography encrypts the message before sending to the destination.The paper provides an overview of methods of storing passwords and authenticating users on Linux and Windows systems, as well as an analysis of password cracking techniques. It also proposes an architecture which uses steganography to enhance protection, and concludes with further scope of the proposed solution.

ANALYSIS OF BRUTE FORCE , DICTONARY ATTACKS:

The Secure Hash Algorithm 1 (SHA-1) is an example of a familiar hash function released by NIST. Attacks can be performed on hashed data using brute-force, dictionary, maskattack, and word-list. GPGPU (general-purpose computing on graphics processing unit) is used to process intensive numeric computation like a CPU. This research was done to understand what kind of real-world password pattern is used by common users and measure the computational power needed to crack those passwords. Based on the test conducted, brute-force attack using alphanumeric character set showed the best result, with 770,884 passwords cracked. For passwords consisting of 8 or more characters, the test showed that brute-force isn't very effective. This is due to the computational need to perform long password length cracking is considerably higher than the shorter one. The most important details in this text are that dictionary attack tests are showing better results than brute-force on long passwords, and that combination of brute-force and dictionary attack can crack 663,992 passwords and 169,080 passwords with 9 or more characters. Additionally, all tests were conducted using several attack methods and results on unsecure password patterns, which were taken from "Splash Data 2012 Worst Passwords".

PASSWORD SECURITY STANDARDS:

This research investigates the possibility of breaking password security standards using offline attacks and public user attributes. Ten million compromised passwords were used to establish the Dictionary Character Set, while 1.4 billion compromised email accounts were used to establish the User-Attribute Character Set. Results showed that there is a high chance of breaking 8–10-character length passwords in a reasonable time. A Machine Sensitive Key Based Password Strength Metric is proposed based on the ratio of risk and safe probabilities of the password. Entropy models the password strength by estimating the average minimum number of bits needed to encode and transmit the Password on a communication channel based on the frequency of the symbols. The Entropy H of Password X with possible characters x1, x2,...

This research investigates if current password standards are still applicable despite the availability of compromised passwords and public data. Cheswick characterized a strong password to be at least seven characters in length and contain a combination of uppercase and lowercase letters, numbers, and standard symbols based on the Shannon Entropy. However, with the availability of compromised and public data, the possibility of breaking the current password standards could be realized. Two cracking schemes were explored: the first is based on the Shannon Entropy, while the second is based on a combination of the two.

CRACKING MORE PASSWORD HASHES WITH PATTERNS:

This paper proposes a novel method for improving dictionary attacks. It exploits password patterns that are commonly preferred by users when trying to choose a complex and strong password. To analyze and show success rates, cracking tests were performed on real-life leaked password hashes using both a traditional dictionary and our pattern-based dictionary. Results showed that our pattern-based method is superior for cracking password hashes. Authentication is an important requirement for information security, and password-based systems are the most commonly used method for authentication. However, passwords are often targeted during cyber-attacks, and attackers use SQL injection vulnerabilities to access database tables. Developers must never store passwords in plaintext within databases, but it is also a critical security weakness if the hash value of a password is calculated and stored without appending a per-user unique salt value to the password before hashing. In a classical scenario, a user chooses a password by a registration process, and the hash value is calculated on the backend-server and stored in the database. This implementation is very insecure, as attackers can perform brute-force, dictionary or rainbow-table attacks to reveal input values from the given output values.

This paper proposes a new method for increasing the success rates of dictionary attacks by analyzing leaked real-life user passwords and identifying several patterns. The method uses brute-force, dictionary, and rainbow-table attacks, which all have pros and cons. Brute-force attacks are time consuming, dictionary attacks are fast, but the success rate is not sufficient, and rainbow-table attacks require having a large disk storage capacity. This paper proposes a new method for increasing the success rates of dictionary attacks by analyzing leaked real-life user passwords and identifying several patterns.

III. Methodologies

BRUTE FORCE ATTACK METHODLOGY:

Brute Force Attack Methodology:

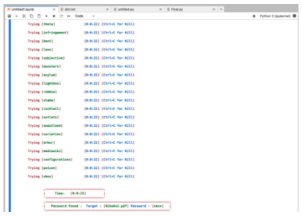
Brute force attacks are time-consuming and computationally intensive. They involve generating every possible combination of characters until the correct password is found. Firstly Identify the password hash: The first step in a Brute Force attack is to identify the password hash. Password hashes are generated by applying a mathematical function to the password, making it difficult to reverse engineer the original password. Password hashes can be found in various locations, such as in operating system files, databases, or application configurations. The next step is to choose a character set. The character set includes all the possible characters that can be used in the password, such as upper- and lower-case letters, numbers, and symbols. The choice of character set can significantly affect the time required to crack the password. A larger character set will increase the number of possible combinations, making the attack more time-consuming. The password length is a critical factor in a Brute Force attack. The longer the password, the more time it will take to crack. Therefore, it's essential to determine the maximum password length that can be used in the attack. If the password length is unknown, a range of possible lengths can be tested. The Brute Force attack generates all possible combinations of characters based on the chosen character set and password length. The order in which the characters are tested can vary, such as starting with all single character passwords, then all two-character passwords, and so on, or using a random order. The generated password combinations are tested against the password hash until the correct password is found. If the password matches the hash, the attack is successful, and the password is revealed. If the password does not match, the next combination is generated and tested. This process continues until the correct password is found or all possible combinations have been tested. Brute Force attacks can be optimized by using parallel processing or distributed computing. Parallel processing involves breaking up the password cracking task into smaller chunks and processing them simultaneously on multiple cores or computers. Distributed computing involves using multiple computers connected over a network to work on different parts of the password cracking task. These techniques can significantly reduce the time required to crack the password. Brute Force attacks can be illegal and unethical if used without authorization. Therefore, it's essential to ensure that the attack is legal and ethical before proceeding. Brute Force attacks should only be performed with the owner's explicit permission or as part of a legal investigation. So the Brute Force attacks are a powerful password cracking technique that involves generating all possible password combinations until the correct password is found. The attack's success depends on the password length, character set, and available computing resources. Optimization techniques such as parallel processing and distributed computing can significantly reduce the time required to crack the password. However, Brute Force attacks can be illegal and unethical if used without authorization, so it's essential to ensure that the attack is legal and ethical before proceeding.

DICTIONARY ATTACK METHODLOGY:

Dictionary attacks are faster and more efficient than Brute Force attacks. They use precompiled word lists to guess the password. The following is a methodology for a Dictionary attack which is The first step in a Dictionary attack is to collect wordlists. A wordlist is a text file containing a list of words that are commonly used as passwords. Wordlists can be found online or created by analyzing previous data breaches or password dumps. A good wordlist should contain a diverse range of words, including common passwords, names, dates, and phrases. Once the wordlists have been collected, they need to be cleaned. Cleaning involves removing duplicates, special characters, and any irrelevant words. The cleaned wordlist should contain only valid words that are likely to be used as passwords. The next step is to generate permutations of the words in the cleaned wordlist. Permutations are created by manipulating the words in the wordlist, such as adding numbers, symbols, or capital letters to the beginning or end of the word. This process creates additional password variations based on the words in the wordlist. The generated permutations are tested against the password hash until the correct password is found. If the password matches the hash, the attack is successful, and the password is revealed. If the password does not match, the next permutation is generated and tested. This process continues until the correct password is found or all permutations have been tested. Dictionary attacks can be optimized by using techniques such as rule-based attacks or hybrid attacks. Rule-based attacks involve applying a set of rules to the wordlist to generate permutations. For example, a rule could be to add a number to the end of every word. Hybrid attacks involve combining Dictionary attacks with other password cracking techniques, such as Brute Force attacks. These techniques can significantly reduce the time required to crack the password. Consider legal and ethical issues: Dictionary attacks can be illegal and unethical if used without authorization. Therefore, it's essential to ensure that the attack is legal and ethical before proceeding. Dictionary attacks should only be performed with the owner's explicit permission or as part of a legal investigation. So the Dictionary attacks are a password cracking technique that involves using a list of commonly used passwords to crack a password. The success of the attack depends on the quality of the wordlist and the available computing resources. Optimization techniques such as rule-based attacks and hybrid attacks can significantly reduce the time required to crack the password. However, Dictionary attacks can be illegal and unethical if used without authorization, so it's essential to ensure that the attack is legal and ethical before proceeding.

IV. RESULTS

Brute force method is effective, but it can be time-consuming and resource-intensive, especially for longer passwords or passwords that contain complex characters. Brute force attacks are often used when the password is not known or is very complex.

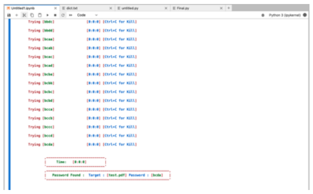


```
Trying [theta]            [0:0:22] [Ctrl+C for Kill]
Trying [infringement]     [0:0:22] [Ctrl+C for Kill]
Trying [bent]             [0:0:22] [Ctrl+C for Kill]
Trying [laos]             [0:0:22] [Ctrl+C for Kill]
Trying [subjective]       [0:0:22] [Ctrl+C for Kill]
Trying [monsters]         [0:0:22] [Ctrl+C for Kill]
Trying [asylum]           [0:0:22] [Ctrl+C for Kill]
Trying [lightbox]         [0:0:22] [Ctrl+C for Kill]
Trying [robbie]           [0:0:22] [Ctrl+C for Kill]
Trying [stake]            [0:0:22] [Ctrl+C for Kill]
Trying [cocktail]         [0:0:22] [Ctrl+C for Kill]
Trying [outlets]          [0:0:22] [Ctrl+C for Kill]
Trying [swaziland]        [0:0:22] [Ctrl+C for Kill]
Trying [varieties]        [0:0:22] [Ctrl+C for Kill]
Trying [arbor]            [0:0:22] [Ctrl+C for Kill]
Trying [mediawiki]        [0:0:22] [Ctrl+C for Kill]
Trying [configurations]   [0:0:22] [Ctrl+C for Kill]
Trying [poison]           [0:0:22] [Ctrl+C for Kill]
Trying [xbox]             [0:0:22] [Ctrl+C for Kill]

        Time:   [0:0:22]

        Password Found :  Target : [Nihaks2.pdf] Password : [xbox]
```

Fig 01: Dictionary Attack



```
Trying [bbdc]             [0:0:0] [Ctrl+C for Kill]
Trying [bbdd]             [0:0:0] [Ctrl+C for Kill]
Trying [bcaa]             [0:0:0] [Ctrl+C for Kill]
Trying [bcab]             [0:0:0] [Ctrl+C for Kill]
Trying [bcac]             [0:0:0] [Ctrl+C for Kill]
Trying [bcad]             [0:0:0] [Ctrl+C for Kill]
Trying [bcba]             [0:0:0] [Ctrl+C for Kill]
Trying [bcbb]             [0:0:0] [Ctrl+C for Kill]
Trying [bcbc]             [0:0:0] [Ctrl+C for Kill]
Trying [bcbd]             [0:0:0] [Ctrl+C for Kill]
Trying [bcca]             [0:0:0] [Ctrl+C for Kill]
Trying [bccb]             [0:0:0] [Ctrl+C for Kill]
Trying [bccc]             [0:0:0] [Ctrl+C for Kill]
Trying [bccd]             [0:0:0] [Ctrl+C for Kill]
Trying [bcda]             [0:0:0] [Ctrl+C for Kill]

        Time:   [0:0:0]

        Password Found :  Target : [test.pdf] Password : [bcda]
```

Fig 02: Brute Force Attack

On the other hand, Dictionary attack method is much faster than a brute force attack because it does not need to try every possible combination of characters. However, it is less effective against complex passwords that do not appear in the dictionary list. Dictionary attacks are often used when the attacker has some knowledge of the target's password, such as the target's interests, job title, or other personal information.

In summary, both brute force and dictionary attacks have their strengths and weaknesses, and their effectiveness depends on the specific password being cracked. A password cracking tool that uses both methods in combination can increase the chances of success in cracking a password.

V. Conclusion

As part of our project, we have researched and analyzed password cracking tools and their benefits in today's digital world. Our findings suggest that these tools have numerous advantages, including the ability to recover lost or forgotten passwords, saving time and effort, preventing data loss, and improving security.

We have also identified the potential for these tools to evolve and expand their capabilities in the future. Specifically, there is an opportunity for developers to create more sophisticated tools that can crack a wider range of file types, such as zip files and Excel files, in addition to their current capabilities of cracking passwords for various operating systems and applications.

However, Another potential disadvantage of password cracking tools is the risk of causing damage to the system or data being targeted. In some cases, attempting to crack a password can cause data loss or corruption, or even lead to system crashes or other types of damage. Furthermore, using these tools can be time-consuming and may require significant technical expertise, which can be a barrier to entry for some users.

Finally, password cracking tools can sometimes be limited by the strength of the encryption used to protect the password. If the password is too strong or uses advanced encryption methods, it may be impossible for even the most advanced password cracking tools to break it. This can be frustrating for users who are trying to recover important data but are unable to do so because of the strength of the password.

Overall, we believe that password cracking tools are valuable resources for professionals who need to access locked-out systems or data. As technology advances and encryption methods become more complex, the need for these tools will only increase. We look forward to seeing how these tools continue to evolve and improve in the future.

References

[1] "Hackers Leak Data Allegedly Stolen from Chinese Chamber of Commerce Website," http://news.softpedia.com/news/Hackers- Leak-Data-Allegedly-Stolen-from-Chinese-Chamber-of-Commerce-Website-396936.shtml, 2013.

[2] V. Goyal, V. Kumar, M. Singh, A. Abraham, and S. Sanyal, "Compchall: Addressing password guessing attacks," in *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)*. IEEE Computer Society, 2005, pp. 739–744.

[3] L. Staneková and M. Stanek, "Analysis of dictionary methods for pin selection," *Comput. Secur.*, vol. 39, pp. 289–298, Nov. 2013.

[4] B. J. Fogg, "Persuasive technology: Using computers to change what we think and do," *Ubiquity*, vol. 2002, no. December, Dec. 2002.

[5] C. Yiannis, "Modern Password Cracking: A hands-on approach to creating an optimised and versatile attack.", Info. Security Grp., Univ. of London, Surrey, Tech. Rep. RHUL–MA–2013– 7, pp. 5-6 , May 2013.

[6] G.Smitha,E. Baburaj,"A survey on image steganography based on Least Significant bit Matched Revisited (LSBMR) algorithm" in 2016 International Conference on Emerging Technological Trends (ICETT), 2016, pp. 1-6.

[7] J.A. Chester, "Analysis of Password Cracking Methods & Applications",Honors Res. Project, Univ. of Akron, OH, pp 9-14, 2015.

[8] M. Agarwal, "Text Steganographic Approaches: A Comparison", International Journal of Network Security & Its Applications (IJNSA), Vol.5, No.1, Jan. 2013, pp 91-106.

[9] T. Gautam,A. Jain,"Analysis of Brute Force Attack using TG – Dataset" in SAI Intelligent Systems Conference, London, UK, Nov. 2015, pp. 984-988.

[10] J. Kävrestad, "Cracking," in *Fundamentals of Digital Forensics: Theory, Methods, and Real-Life Applications*, Cham, Springer International Publishing, 2018, pp. 93-103.

[11] M. Marks, J. Jantura, E. Niewiadomska-Szynkiewicz, P. Strzelczyk and K. Góźdź, "Heterogeneous GPU&CPU Cluster for High Performance Computing in Cryptography," *Computer Science,* vol. Vol. 13 (2), pp. 63-79, 2012.

[12] W. Qiu, Z. Gong, Y. Guo, B. Liu, X. Tang and Y. Yuan, "GPU-Based High Performance Password Recovery Technique for Hash Functions," 2016.

[13] I. Marvridis, E. Androulakis, B. Halkias, P. Mylonas, "Real-life paradigms of wireless network security attacks," in Proceedings 2011 Panhellenic Conference on Informatics, pp. 112-116.

[14] M. Weir, "Password cracking using probabilistic context-free grammars," In Security and Privacy, 2009 30th IEEE Symposium on, pp 391-405.

[15] C. Karlof, D. Wagner, "Hidden Markov model cryptanalysis," Lecture Notes in Computer Science, 2003, pp. 17-34.

[16] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, "Optimization by simulated annealing," Science, 1983, 220(11):650-671.