Leveraging data from leaked passwords to inform the creation of more secure, yet memorable passwords

Aadeesh Jain and Joe Isaacs

Received December 28th, 2022
Accepted April 19th, 2023
Electronic access June 30th, 2023

Protecting yourself in the online world has become crucial, as cyber-criminals continue to create a nuisance with a plethora of cyber-attacks. Password creation is the most common way to protect data on the internet, however, the unpredictability of a password also known as password entropy plays a major role in revealing the true strength of a password. In this paper, we have attempted to demonstrate the strength of the most commonly used passwords (400 million hashed passwords hashed in SHA-1), by trying to crack them using three methods coded in python programming. Our ultimate aim is to show that the common practices of password creation put the data of a user at risk, hence informing the user on how to create strong yet memorable passwords.

Introduction

Computers have become faster as technology has advanced, making it easier to gain access to insecure systems that use password-based authentication[1]. Passwords created by the modern user generally has low entropy, with most passwords containing common phrases, words, and publicly available information such as names, birthdates, addresses, and so on. Therefore, creating passwords with high entropy has become a must for all in order to protect your online persona from the plethora of attacks aimed at stealing sensitive online information such as credit card numbers, bank details, etc. Online hacks have costed the global economy billions of dollars[2]. It has now become crucial to bring about a change in the password creation habits of the general public, by awaring them, through studies like this.

Earlier studies like Brute-force and dictionary attack on hashed real-world passwords by L. Bošnjak, J. Sreš and B. Brumen, shows that students at the University of Maribor in Slovenia created passwords that included common phrases, names, and other personal information that were easy to guess, reinforcing our key belief that passwords created by the larger populous are also easy to crack with modern technology. Research paper, Password Security: A Case History. 1979 by Robert Morris and Ken Thompson showed that passwords are stored in hashed forms like SHA-1, which can be deciphered easily, also strengthens the motive of our research, to aware the reader that we cannot rely on the safety measure taken by websites to store our data. Hence, these two papers have motivated us to investigate this further.

This paper has tried to address the issue of password security by checking the strength of a leaked password database hashed in SHA-1, which we will be calling: Shapwned, obtained from a popular website: haveibeenpwned.com[3]. We have performed three primary attacks on this database, namely the brute-force attack, the conventional dictionary attack, and a list of previously used passwords. The results of these attacks have shown that most of the cracked passwords were made up of names, birthdates, and other commonly used phrases and publicly known information. Thus, this calls to the attention that most passwords created by the present-day user, are easily decipherable[4], putting a risk to the user's online persona.

Background

Previous research by L. Bonjak et al.1 demonstrates that passwords created by students at the University of Maribor, Slovenia included common phrases, names, and other personal information which is easy to guess. Research by Robert Morris et al. in their paper[5], shows that passwords are still being saved in hashed forms like SHA-1, which is easy to crack. In our study we build upon the works of these two research papers, to attempt to crack a list of leaked passwords Shapwned. Fig. 1(a) shows an algorithm used by all attacks. The attacks consist of iterating through the list of potential passwords that are to be checked. Each element is hashed using the SHA-1 algorithm and checked to see if it is present in the Shapwned database.

The list of passwords checked in a brute-force attack is generated by every possible combination of the characters given in a character set, which is a set of possible characters that could include the password which is to be cracked. Fig.1(b) shows that with each addition of a new character and/or the increase

```
for p in [[passwords to be checked]]:
  h = hash(p)
  check if h is in the list Shapwned
```

(a)

```
N = number of characters
L = Length of combination
```

(b) $\#combinations = N^L$

Fig. 1

Table 1 Performed attacks' algorithm results

ATTACK	Number of Hashes Cracked	Number of Hashes Calculated	Time Taken(s)
Total	28,449,360	4,897,038,148	16,606
PREVIOUSLY USED PASSWORDS METHOD	8,698,400	57,377,568	640
Previously Used Passwords(Rock You List, 14 Million passwords)	8,698,400	57,377,568	640
BRUTE FORCE ATTACK	19,496,409	3,505,604,448	10,396
Brute Force Attack(Lengths 3 to 6, lower case alphabets)	4,212,866	797,644,800	2,589
Brute Force Attack(Lengths 3 to 6, Upper case alphabets)	462,602	797,644,800	1,471
Brute Force Attack(Lengths 3 to 5, lower and Upper case alphabets)	1,181,743	1,040,891,904	2,741
Brute Force Attack(Lengths 3 to 6, only numbers)	1,106,494	4,444,000	9
Brute Force Attack(Lengths 3 to 8, only numbers)	13,836,571	444,444,000	2,819
Brute Force Attack(Lengths 3 to 5, numbers and lower case alphabets)	1,880,853	187,244,256	424
Brute Force Attack(Lengths 3 to 5, numbers and Upper case alphabets)	366,745	187,244,256	273
Brute Force Attack(Lengths 3 to 4, numbers, Lower and Upper case alphabets)	514,950	46,046,432	70
CONVENTIONAL DICTIONARY ATTACK	254,551	1,334,056,132	5,570
Dictionary Attack(200k words Dictionary)	139,496	777,732	75
Dictionary Attack(Combining first 10k words of dictionary with itself)	33,870	400,000,000	691
Dictionary Attack(Updated Dictionary with Birth year(1900-2100) behind each word)	2,866	155,546,400	385
Dictionary Attack(Updated Dictionary with 1 to 3 digit numbers behind each word)	80,317	777,732,000	4,419

in the length of the passwords to be checked, the number of possible combinations increases exponentially, thus the time consumed also increases exponentially.

The list of passwords checked in a Conventional dictionary attack is generated by the words found in a dictionary, however, the scope of this attack is increased when the words found in the dictionary are combined with themselves or appended with numbers like birthdates, years, etc.

The previously used passwords method is already in the form of a list, in our case the Rock You list[6], which contains passwords previously used in the real world.

Methods

The Shapwned database is extensive, approximately 32GB, thus due to the limitation of computing power, we had to use only the first 16GB part of the list, further broken into 4 parts of 4GB each. With the help of Python programming language, we ran three major attacks, and some modified sub attacks, and recorded the results in the form of a table as shown in Table. 1.

Brute Force

Due to computing power limitations, we had to run the brute-force attack multiple times for different lengths and character sets. We used lengths from 3 to 8 and characters: lower and uppercase alphabet and numbers. Since this is a rudimentary attack, we expect the number of passwords cracked to be lower and the time taken to be higher as all the possible combinations are being checked.

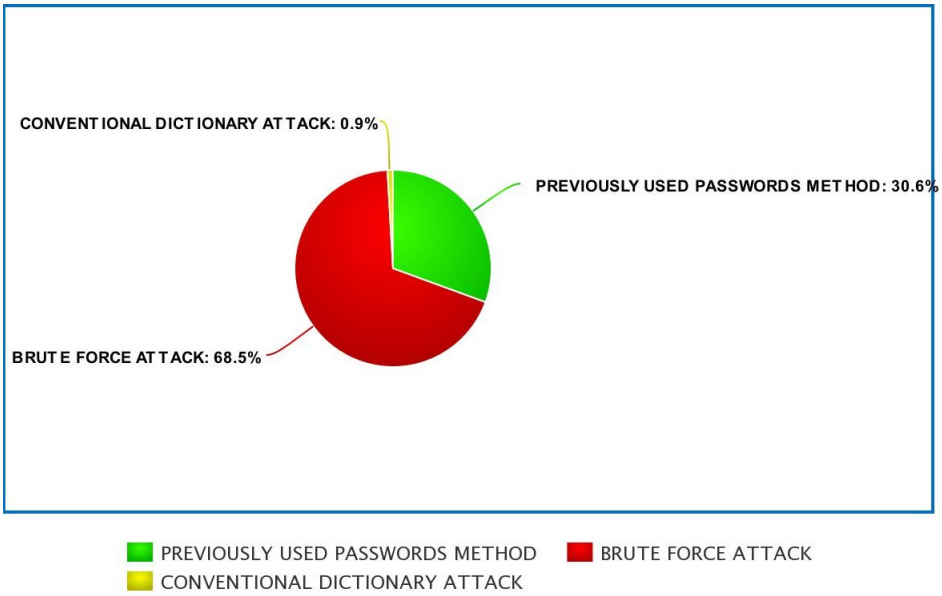

Fig. 2 Percentage Passwords Cracked

Conventional Dictionary Attack

We have used a list of approximately two hundred thousand English dictionary words for our dictionary attack. We first ran the conventional dictionary attack without modifications to crack the passwords. Then we modified the attack three times to combine each word with every other word in the dictionary or include numbers after each word. Since we have used an English dictionary and also modified it, the expected number of passwords to be cracked should be higher, while the time taken would depend on the number of hashes that need to be checked.

Previously used passwords

For the previously used method, we have used the RockYouList, which contains a list of thirteen million previously used passwords. As this list already contains the possible passwords that are used in the real world, the number of passwords cracked is the highest in this attack.

Results

The results obtained from the attacks are shown in Table 1. The table reflects the attacks and the sub-parts of attacks, if any, with corresponding values of the number of hashes cracked, the number of hashes calculated, and the time taken to run the program. A supporting figure, Fig. 2 shows the percentage of the number of hashes cracked by each of the three attacks.

The previously used passwords method cracked around 8.5 million passwords, which was a little lower than what we had expected, as Rock You List contained 13 Million most probable passwords.

Unexpectedly, the Brute-force attack cracked approximately 19.5 Million unique passwords, thus the most successful attack. The table shows, that approximately 70% of passwords are cracked by the brute-force attack consisting of numbers ranging from lengths three to eight. Approximately 28% of the passwords cracked by the brute-force attacks only consisted of alphabets, both upper and lower case, ranging from lengths three to six. The Rest 2% of the passwords cracked by the brute force attacks were made up of alphanumeric characters.

The Dictionary attack cracked approximately 250 thousand unique passwords, of which 32% of the passwords had 1-3 digits behind each word, and 1.2% of the passwords cracked by the dictionary attack contained possible birth years behind them.

The results show the majority of the passwords cracked were either made up of only numbers or only alphabets and were in the range of three to eight. Alphanumeric passwords were cracked the least and thus have the highest password entropy.

The dictionary attack didn't perform at par with the expectations, as probably most passwords didn't use dictionary words, but rather alphanumeric strings not found in the dictionary.

Discussion

The results are shown to reinforce the key objective of the paper, that a significant number of passwords have low entropy in the database, as they were cracked in a short period of time. Such passwords provide a false sense of cyber security but are susceptible to being cracked by cybercriminals. Thus, creating passwords with more rare and longer combination of characters, having high entropy is necessary to protect online data[7].

This paper had limited results due to the availability of low computing power as the machine used for this paper was very limited in its RAM, thus number of passwords cracked per second played a major issue. Further studies need to be conducted to improve these attacks by checking larger string lengths and character set and further inform the reader about the repercussions of weak passwords[8].

Conclusion

The creation of strong passwords is therefore important, as evidenced by the number of passwords cracked. The passwords generated by password generators are the safest option as they are lengthy random alphanumeric strings with special characters, which you do not need to memorize as they are saved by a password manager[9].

Acknowledgments

We would like to express our gratitude towards Lumiere Education, who gave us the opportunity to work on this research paper.

References

1 L. Bošnjak, J. Sreš and B. Brumen, *Brute-force and dictionary attack on hashed real-world passwords*.
2 L. Graham, *CNBC Website*, `http://www.cnbc.com/2017/02/07/cybercrime-costs-the-global-economy-450-billion-ceo.html`, accessed 1 March 2023, ¡.
3 `https://haveibeenpwned.com`, accessed 6 August 2022,.
4 D. Florencio and C. Herley, *A Large-Scale Study of Web Password Habits*.
5 R. Morris and K. Thompson, *Password Security: A Case History*.
6 W. J. Burns, *Kaggle Website*, `https://www.kaggle.com/datasets/wjburns/common-password-list-rockyoutxt`, accessed 15 August 2022,.
7 MYildirim and I. Mackie, *Encouraging users to improve password security and memorability*.
8 K. Chanda, *Password Security: An Analysis of Password Strengths and Vulnerabilities*.
9 C. Luevanos, J. Elizarraras, K. Hirschi and J.-h. Yeh, *Analysis on the Security and Use of Password Managers*.