Password Hardening using ML based algorithms

Nividh Singh
[1]*Institute for Computing in Research, Portland, Oregon, United States of America*

Abstract. Some of the biggest issues in cybersecurity are password hacking and social engineering. This paper focuses on password hacking. This involves a brute force attack to try to find a user's password. However, these brute force attacks can be changed to find passwords faster. For example, a malicious hacker may try to use the english dictionary with special characters in between words. Another hacker may go in alphabetical order, and another might go in order of most probable in leaked passwords. This paper tries to simulate these various hacks to see if passwords are secure or not. This is significant because if an ethical hacker can go through passwords and find which ones are weak, then the user can be alerted to change their passwords. This makes softwares and programs more secure, and also protects user information. The program tested passwords against three different types of attacks. The first one is a traditional brute force attack which goes in alphabetical order. The hacker could also put numbers before letters, put upper cases before lower case, or choose any rules for "alphabetical", so the program also tested these combinations. Since the program is given the password, it is able to calculate after how many tests would that password be found, and this makes it run a lot faster (instead of actually going through all the tests and comparing them). The next method that hackers use is creating probabilistic guesses for passwords using leaked passwords. In the last decades, billions of passwords have been leaked. Hackers can use these leaks to find patterns in passwords. For example, this program found that after 8 letters there's an increased probability of a number after. This would make it highly probable that this order of brute force would find passwords quicker. Finally, the last method checked involved using english words and filling in the remaining spots with letters, numbers or symbols. Many passwords use English words because they are easier to remember. However, by doing so, it also makes it more susceptible to a hack that involves using these words. The results showed that these algorithms worked and were able to identify passwords that weren't secure. The programs created didn't perform as well as some of the other programs out there. However, the program was successful in that it found patterns in the passwords and used them to find weaknesses in passwords that might have otherwise been missed In conclusion, these programs are really important because they find patterns that normal human intuition or basic computer checkers would miss. The algorithm was able to find patterns that aren't obvious to humans. For example, even if somewhat randomly generating a password, humans might by nature alternate between vowels and consonants, or atleast try not to have a long string of consonants next to each other. Humans might not notice this, but small nuances like this make the process of hacking more efficient

1 Introduction

In today's society, two of the biggest concerns regarding cybersecurity are password hacking and social engineering. Historically, encryption and decryption have been the primary focus in the field of cybersecurity. However, with the rise of algorithms like AES and RSA, the transfer of files is more secure than ever before because of the lack of computing power needed to factorize large numbers. However, password hacking and social engineering remain some of the biggest problems in keeping accounts and information secure. Password hacking usually involves a brute force attack in order to try to "guess" the password. Most softwares that involves passwords have some basic requirements that make passwords more secure. For example, a website might require passwords to be 12 characters long, have special characters, have numbers, and have upper and lowercase letters. A majority of the time, this approach provides protection against passwords. However, newer attack softwares are getting smarter, and instead of pure brute force attacks to find passwords, they use words and patterns found in leaked passwords to more efficiently be able to hack into accounts The second problem is social engineering. Social engineering is the term used for a broad range of malicious actions to trick users into disclosing private information. One of the most common types of social engineering actions are phishing attacks. These involve emails or messages with a link that takes users to websites that appear to be from a credible source. For example, the link might take the user to a website that looks and acts like Facebook.com, but in reality is Facebooks.com. Then, the user signs into their account on the fake website, and the attacker gains their login information. Many platforms have programs that identify messages that might be social engineering attacks. For example, most mail services have spam filtering that mark these attacks as spam. However, this isn't true for all services, and many times these messages make it past filters. One prime ex-

**Correspondence to*: Nividh Singh (nividh.singh@gmail.com)

ample of this is on Instagram, where many people have experienced being hacked due to these social engineering attacks. They might send a message from one account offering a chance to win money if they sign into Instagram from a link. Then, when some people fall for the trick, they use their accounts to target that individual's friends. This cycle continues until many users have lost their accounts. This research paper focuses on the first issue, password hacking. There is a lot more research on the second problem, and many algorithms have been written to find attempts at social engineering. Many of these algorithms use machine learning to identify and filter out malicious links and messages. However, there isn't as much research in the first problem. The biggest program focused on the first issue is John the Ripper. This research paper falls into the category of ethical hacking. Though hacking is typically associated with malicious intent, there is another part of hacking called ethical hacking. This involves trying to hack into a system. However, the intent is not to do damage, but rather is to make systems more robust against posibile malicious hacks.

2 Motivation

The goal of this research is to be able to identify weak passwords so users can change them instead of being hacked. In cybersecurity, there is a constant battle between how advanced malicious hackers are and ethical hackers are. This research is meant to go through passwords and find weak passwords that are susceptible to being hacked. That way services can require the user to change the password before the account is hacked.

3 Method

For this research, multiple ways of hacking were simulated to test if a password was secure. A description of the methods is below. Each method has a description and examples of passwords it labels as good or bad

3.1 Method 1: Alphabetical Order

One common and simple method for hacking is to go in alphabetical order. This approach is simple to program and doesn't require a high level of expertise to implement. Thus, a password should be robust against this basic hacking attempt. Because of its simplicity to implement, if a password isn't robust against this there is a high likelihood of being hacked. The other keynote is that the hacker could make alphabetical any way they want, so numbers could be before the alphabet, and uppercase could be before or after lowercase. Thus, it is important to test all of those cases. However, some people may overcompensate for this and try to make their passwords more secure by making them something like 'zyxtz'. However, hackers also go reverse alphabetical order, which makes these passwords really weak.

Good Passwords	Bad Passwords
gamscakht7ty9vc5	ABCDEFG
Orangeceltics	AAAAA12342
2b73004faf006983703a2ab33	DaRes!1
adrianazarel9602211	78Apollo
3991seivadwb1123	asta4882

3.2 Method 2: Probabilistically Order

This method of hacking is harder to implement but is also harder to create robust passwords against. Throughout the history of computers, there have been many data leaks resulting in billions of passwords being released. This method of hacking uses these passwords to find trends and patterns, and then uses those to try to hack into programs. The program orders all the characters from most probable to least probable. Then from there, given the first character, it orders the second character from most to least probable and continues to do that. This way, the program has a much higher chance of finding the password faster.

3.2.1 Pure Probability

One approach to this type of hacking is to sequence the leaked passwords and find the probability of the next letter based on previous letters. Doing this, we find that s and d are the most common first letters. From there, we look at all the passwords with s as the first letter, and find the most common second letter. We keep doing this, and that way we are able to go through the set of possible passwords much more efficiently.

3.2.2 Using Machine Learning

The other option is to create an ML program that takes the previous letters as input and the output is the most probable next letter. One way of implementing this is by training a model to use Softmax output and each output node is associated with a character. This approach is better than the pure probability approach because it can identify passwords. For example, if a password is 'andth_-', the 'th' might seem to be the start of the word 'the'. A pure probability approach might say that the highest probable is f.

3.2.3 Pure Probability vs. Machine Learning

Machine Learning is generally better than pure probability. Pure probability gives all of the previous characters the same weightage. So if we're trying to find the 12th character, all 11th characters matter. However, the better way, which is implemented with machine learning is to give the closer characters more importance. This allows the program to find patterns that it wouldn't have with pure probability

Good Passwords	Bad Passwords
1784mamaq	ABCDEFG
cp57ji4bwnavy2	BLUEABA2
73Ekmrfcfh73	khongkimthanh
gamscakht7ty9vc5	EJNIH
3991seivadwb1123	asta4882

3.3 Method 3: Taking Out Words

The next method tested was putting characters in between words. Many passwords, like "5139210maddog" or "emmaprofileimage" which are found in the weakpass list, include english words (like "mad", "dog", "profile" and "image"). This method tries to put characters before, between, and after English words. The algorithm can find which English words are more prevalent, and start with them. This is faster, because then instead of going through $94^(12)$ combinations for a 12 character password, the program can decrease it down to $94^4 \times 61569^2$. Though this doesn't seem too much faster, it can find passwords 1 million times faster.

Good Passwords	Bad Passwords
gamscakht7ty9vc5	BLUEABA2
54429863asd	Orangeceltics
2b73004faf006983703a2ab33	funseeker4347
eb3-R1a7n	78Apollo
3991seivadwb1123	emmaprofileimage

4 Results

The results showed that these algorithms worked and were able to identify passwords that weren't secure. The programs created didn't perform as well as some of the other programs out there. However, the program was successful in that it found patterns in the passwords and used them to find weaknesses in passwords that might have otherwise been missed

5 Conclusion

In conclusion, these programs are really important because they find patterns that normal human intuition or basic computer checkers would miss. The algorithm was able to find patterns that aren't obvious to humans. For example, even if somewhat randomly generating a password, humans might by nature alternate between vowels and consonants, or atleast try not to have a long string of consonants next to each other. Humans might not notice this, but small nuances like this make the process of hacking more efficient

6 Future work

The program right now tests passwords to be robust against a few types of hacking algorithms. In the future, more tests on passwords could be implemented. Here are a list of things to add to the checking methods:

- Check just numbers (this normally runs quicker)
- Check just lowercase/just uppercase letters Check numbers first, letters in the middle, and numbers in the end (or without the numbers in the beginning or the end) Add other languages so the program can use words from other languages Add words with letters flipped with symbols to english dictionary
  - 3 instead of e (Jun3 / June)
  - @ instead of a (@pple / apple)
  - ! instead of i (!sland / island)

Another thing that could increase the performance is using a custom training loop with TensorFlow. Finally, the last thing to improve this project would be to create a GUI so that users could more easily run the program. In the future, the goal is to implement a website as the frontend so that users could test passwords without having to run a python program.

7 Acknowledgements

Thank you to Dr. Shih-Lien Lu from Werner Pacific University for his guidance and help for this project. Also thank you to the Institute for Computing in Research for the structure and support for this project

8 References

- https://www.imperva.com/learn/application-security/social-engineering-attack/: :text=Social%20engineering%20is%20the%20term,or%20giving%20away%20sensitive%20information.
- https://github.com/openwall/john
- https://github.com/ohmybahgosh/RockYou2021.txt
- https://www.softwaretestinghelp.com/password-cracker-tools/
- https://www.techopedia.com/definition/13667/password-hardening#: :text=Explains%20Password%20Hardening-,What%20Does%20Password%20Hardening%20Mean%3F,othe