**Attempting to crack the 20 most common passwords**

**Nick Lund**

**Abstract**

Passwords are becoming more important each year due to an increased amount of information secured behind passwords. Computing power is also increasing each year making password attacks even more threatening. Therefore, people need to understand that having an easy to remember password is insecure and can result in a significant amount of private information being leaked. In this project I have used John the ripper with both incremental and wordlist mode to attempt to crack the 20 most used passwords. I Hashed the passwords with both md5 and bcrypt to see the difference between a good and bad hashing algorithm. I found that bcrypt is much more secure than md5 due to its slow hashing algorithm. Additionally, wordlist mode was much more effective than incremental mode. The worrying thing about these results is that even when using a good hashing algorithm like bcrypt, a wordlist attack can instantly crack most of the passwords. This project hopes to display how insecure easy to remember passwords are.

**Introduction**

The problem with password security is that many people don’t want to make a good password because they are too hard to remember. So, people make easy to remember passwords that are at the same time just as easy to crack. Passwords are used to protect banking information, emails, your cell phone, and many more accounts/ computer systems that most people use daily. A compromised password can be very destructive to a company or an individual. For example, if an individual’s password gets crack for their banking information, the hacker can now do whatever they want with that users’ funds. A real example of how devastating password leaks/cracks can be seen in “Loose Password Security in Chinese Cyber World Left the Front Door Wide Open to Hackers—An Analytic View” where they say “In 2011, a huge cache of personal data from China’s most popular websites leaked onto the Internet. Between December 21 and 25, hackers released more than 100 million users’ account information, including usernames, passwords, and emails” [1]. Even with how widespread the use of passwords is and how devastating a password crack can be, many people still have easy to crack passwords. This may be due to a lack of care or more probable a lack of knowledge/ misconceptions about password security. One startling statistic is that 61% of people admit to using the same password for multiple accounts [2]. This is a bad idea because if a hacker gets your password for one account, they now have it for multiple accounts without doing any extra work.

Nowadays having a good password is more important than ever before due to the increasing size of private information being stored on the internet. Passwords are the primary way to authenticate users on the internet. The start of password systems dates to the 1970’s with the development of the UNIX operating system. The UNIX developers decided that storing passwords in plain text would be a bad idea because if someone hacked the file storing the passwords, they could read all the passwords. So, the UNIX developers decided to use hashing to store the passwords in a more secure way [3]. “Cryptographic Hash Function is a function that takes an input of arbitrary length and produces a small output of fixed length” [4]. Hashing a password essentially mean to put the original plain test password through a hashing algorithm that then outputs a string of seemingly random and useless characters. That way if a hacker was able to get the file containing the passwords, they would just see a bunch of useless hashes. One way to calculate password strength is called entropy. This calculation considers password size and types of characters used. An example entropy calculation for password “1234” is log2(104). This is because the password “1234” only uses digits so it only has a possible 10 different characters, and it is only 4 character long. So, the total possible passwords for that character set and length are 104. That is the entered into log base two to find the entropy, in this case 13.29. This is called proactive password checking. Although this is a good way to check password strength, in “An Online Password Guessing Method Based on Big Data” they used a large dataset of previously leaked passwords and were able to guess passwords that had good entropy [5].

**Methodology**

The list of most common passwords will be from <https://nordpass.com/most-common-passwords-list/>. All 20 of the passwords will then have their entropy calculated to show their expected strength. All the passwords will then be converted into both bcryrpt and md5 hash output. The passwords hashed in bcrypt will use <https://bcrypt-generator.com/>. The passwords hashed in md5 will use <https://md5calc.com/hash/md5/11234>. I will download John the ripper from <https://www.openwall.com/john/>. John the ripper will be used to attempt to crack all the hashed passwords. “John the Ripper (called John for short) is an open-source software focusing on recovering Unix passwords” [6]. The passwords will first be attacked with John’s incremental mode which acts like a smart brute force. This will run for a max of two hours. After the two hours are done, I will record which passwords were cracked and which ones were not. Next the passwords will be attacked by a wordlist mode. This test will also run for a max of two hours per password to see if they can be cracked. I will use the rockyou word list from <https://www.kaggle.com/wjburns/common-password-list-rockyoutxt>. My GitHub can be found here: <https://github.com/NickLund13/Senior-Research>.

**Results**

I found the top 20 most used passwords and calculated the entropy for each one, as seen in **Table1**. Then I hashed all password with both md5 and bcrypt, as seen in **Table1**. When attempting to crack all md5 and Bcrypt passwords with John’s incremental attack method I found that bcrypt had many more passwords that I could not crack in two hours compared to md5, as seen in **Table2**. When attempting to crack all hashed passwords with the wordlist mode I found that bcrypt performed better than md5 but still did not perform well, as seen in **Table2**. The effectiveness of each hashing algorithm and each attack method can be seen in **Table3** and **Table4**. **Table3** shows the percentage of the passwords for each hashing algorithm that were cracked within two hours using either incremental or wordlist. **Table4** shows the percentage of the passwords for each hash algorithm that were cracked instantly (less than one second) using either incremental or wordlist. Both passwords “picture1” and ”000000” took 4 seconds on md5 incremental but over two hours on bcrypt incremental. When comparing the attack speed between the two algorithms, md5 is a significantly faster hash. During an incremental attack against md5, my computer got up to a speed of 77, 644,000 passwords tested per second, as seen in **Figure1**. On the other hand, during an incremental attack against bcrypt, my computer got up to a speed of 194 passwords tested per second, as seen in **Figure2**. Both md5 and bcrypt had over two-hour passwords on incremental that became instant with wordlist. One issue I ran into was that for four passwords during the bcrypt wordlist attacks I had to prematurely stop my attack because I had noticed my CPU was reaching uncomfortable temperatures of 87 Celsius. Although I was not able to finish the attack on the four passwords it should be noted that they would have all been eventually cracked (possibly taking longer than 2 hours) because all the passwords were in the rockyou wordlist. Looking at this data it can be concluded that common passwords can be easily cracked especially when hashed with an outdated hashing algorithm or using a wordlist.

**Table1.** This table shows each of the top 20 common passwords, their entropy, their md5 hash, and their bcrypt hash.

|  |  |  |  |
| --- | --- | --- | --- |
| Passwords | Entropy | Md5 hash | Bcrypt hash |
| 123456 | 19.93 | e10adc3949ba59abbe56e057f20f883e | $2a$12$1auql9AO6f8GHaKfxjTzWOFETJTHwSrgoSR2M1a.0XqRe5xZsSuB. |
| 123456789 | 29.90 | 25f9e794323b453885f5181f1b624d0b | $2a$12$B286ncJaWKIiXKKUa8MOo.tm6.c4lVpQsYGJuD4X66mSA2cjF70VW |
| picture1 | 41.36 | 2712e0b4e97c4a17a90a6417ccf757ba | $2a$12$aa/KkQLgX2F2uzjyuKGezuP/jrdLIyDcw8lexkN20Bblsww0M44Ze |
| password | 37.60 | 5f4dcc3b5aa765d61d8327deb882cf99 | $2a$12$.m4oLuYG0EE4XqwtphGJY.Xa8bSaz.OMIqPrn34lzYAyktoQxljqa |
| 12345678 | 26.58 | 25d55ad283aa400af464c76d713c07ad | $2a$12$oFpea9e3yrT1W3x2EN4hhutaUEVL2DMMaOo0nnBAwLJUAwUZqnWiS |
| 111111 | 19.93 | 96e79218965eb72c92a549dd5a330112 | $2a$12$au1/Ixy37mzlVP0LUTSV4uj2OEiziqO.MmhcfbXJ8IfUSYxr7egL6 |
| 123123 | 19.93 | 4297f44b13955235245b2497399d7a93 | $2a$12$WZpNEVcqgWw.I9vlEiZtieO0HsJjz6RmExvYZ426LTfWXIhkGRD5i |
| 12345 | 16.61 | 827ccb0eea8a706c4c34a16891f84e7b | $2a$12$ogrwCnbiyaYtTWu6HTaGNO5kPT.0koOouK5pJG0WTmd85G81QBLlq |
| 1234567890 | 33.22 | e807f1fcf82d132f9bb018ca6738a19f | $2a$12$KL/nFGqSEAeGMjGPWZXPvem/SytHy2/43Rlks8vz.4Fc9xkC.fCce |
| senha | 23.50 | e8d95a51f3af4a3b134bf6bb680a213a | $2a$12$21Ew7ryRwUn8o/ul0tWJI.vWpyIkBbHBxfeTpb6kQ9J9x/SPfShza |
| 1234567 | 23.25 | fcea920f7412b5da7be0cf42b8c93759 | $2a$12$XjLk.hWLlQiF4KhDTE6E0eeyub9iBjS8A9vzOGZuxxN0VEiNrcRVm |
| qwerty | 28.20 | d8578edf8458ce06fbc5bb76a58c5ca4 | $2a$12$s9pMswJZhq9662sx9oJYLugdlog.avpG7gg76XE5tTofzMQ2vjYKC |
| abc123 | 31.02 | e99a18c428cb38d5f260853678922e03 | $2a$12$Yfk6zDT2zDIVL29EyArxXeQHwQlPwlCGhBiRYHFhCK8CqB9Vuq0L6 |
| Million2 | 47.63 | f67336a9f6f8bd9502f48d655ab852c8 | $2a$12$E1qLUqQCu7od/EZGP1dAC.nU6/iAbN2J9TD4si2Rm2YFRjg/R89ky |
| 000000 | 19.93 | 670b14728ad9902aecba32e22fa4f6bd | $2a$12$KaRRLEVZBwbkDbiC3.tsHORB1Z43zQrjxUIjXNOltjLI.MHNOlOve |
| 1234 | 13.29 | 81dc9bdb52d04dc20036dbd8313ed055 | $2a$12$fKxirr9E/2oCDxvebM5.6eRvOI0yHgoikxsbuHoHO.AdKl/4QxU5W |
| iloveyou | 37.60 | f25a2fc72690b780b2a14e140ef6a9e0 | $2a$12$XAt.xXQEWqwdcZqW41OD2u5HdAYh737DPctc0kTsqq0VtHuMmrIh2 |
| aaron431 | 41.36 | 96d594cc3042eb642dafa3a41b641703 | $2a$12$eh5PxaT6rioG5ngqT1qz8eHqSP1ypINb1i1W3IkGjWrJUMtD/myw |
| password1 | 46.53 | 7c6a180b36896a0a8c02787eeafb0e4c | $2a$12$bmAlQPhwMmJRRYl0j33f8ee0Dtu8cc5evl4WZn6EEzyX8WJxkF2Rm |
| qqww1122 | 41.36 | 748f5db2ee3eb8c5d27aba054ac99048 | $2a$12$8qNSG51NogrtvxeU.vWa3uEZkAOylnwcY0JHJog8EY7AwqvViF0/y |

**Table2.** This table shows each password, both hashes for each password, and how long it took to crack each hashed password using incremental and wordlist.

|  |  |  |  |
| --- | --- | --- | --- |
| Password | Hash | Incremental | Wordlist |
| 123456 | Md5 | 0 seconds | 0 seconds |
| 123456 | Bcrypt | 1 second | 0 seconds |
| 123456789 | Md5 | 21 seconds | 0 seconds |
| 123456789 | Bcrypt | 2 hours+ | 0 seconds |
| picture1 | Md5 | 4 seconds | 0 seconds |
| picture1 | Bcrypt | 2 hours+ | 1 minute 42 seconds |
| password | Md5 | 3 seconds | 0 seconds |
| password | Bcrypt | 35 minutes 50 seconds | 0 seconds |
| 12345678 | Md5 | 6 seconds | 0 seconds |
| 12345678 | Bcrypt | 2 hours+ | 0 seconds |
| 111111 | Md5 | 0 seconds | 0 seconds |
| 111111 | Bcrypt | 1 second | 0 seconds |
| 123123 | Md5 | 0 seconds | 0 seconds |
| 123123 | Bcrypt | 1 second | 0 seconds |
| 12345 | Md5 | 0 seconds | 0 seconds |
| 12345 | Bcrypt | 1 second | 0 seconds |
| 1234567890 | Md5 | 2 hours+ | 0 seconds |
| 1234567890 | Bcrypt | 2 hours+ | 0 seconds |
| senha | Md5 | 7 seconds | 0 seconds |
| senha | Bcrypt | 2 hours+ | 13 minutes+ |
| 1234567 | Md5 | 11 seconds | 0 seconds |
| 1234567 | Bcrypt | 2 hours+ | 0 seconds |
| qwerty | Md5 | 1 minute 22 seconds | 0 seconds |
| qwerty | Bcrypt | 2 hours+ | 0 seconds |
| abc123 | Md5 | 0 seconds | 0 seconds |
| abc123 | Bcrypt | 1 minute 8 seconds | 0 seconds |
| Million2 | Md5 | 2 hours+ | 0 seconds |
| Million2 | Bcrypt | 2 hours+ | 13 minutes + |
| 000000 | Md5 | 4 seconds | 0 seconds |
| 000000 | Bcrypt | 2 hours+ | 0 seconds |
| 1234 | Md5 | 0 seconds | 0 seconds |
| 1234 | Bcrypt | 1 second | 0 seconds |
| iloveyou | Md5 | 1 minute 27 seconds | 0 seconds |
| iloveyou | Bcrypt | 2 hours+ | 0 seconds |
| aaron431 | Md5 | 11 minutes 3seconds | 0 seconds |
| aaron431 | Bcrypt | 2 hours+ | 13 minutes+ |
| password1 | Md5 | 30 seconds | 0 seconds |
| password1 | Bcrypt | 2 hours+ | 0 seconds |
| qqww1122 | Md5 | 2 hours+ | 0 seconds |
| qqww1122 | Bcrypt | 2 hours+ | 13 minutes+ |

**Table3.** This table shows the percentage of the passwords for each hashing algorithm that were cracked within two hours using either incremental or wordlist.

|  |  |  |
| --- | --- | --- |
| %Cracked in 2 hours | Incremental | Wordlist |
| Md5 | 85% | 100% |
| Bcrypt | 35% | 80% |

**Table4.** This table shows the percentage of the passwords for each hash algorithm that were cracked instantly (less than one second) using either incremental or wordlist.

|  |  |  |
| --- | --- | --- |
| %Cracked instantly | Incremental | Wordlist |
| Md5 | 30% | 100% |
| Bcrypt | 0% | 75% |

Graphical user interface, text, chat or text message

Description automatically generated

**Figure1.** This figure shows the attack speed in passwords tested per second for incremental md5.

Text

Description automatically generated

**Figure2.** This figure shows the attack speed in passwords tested per second for incremental bcrypt.

**Discussion**

These results are not only important for computer science related professions, but for everyone. These results display and stress how insecure a common or simple password is. The significance of the results is that any information locked behind a weak password can be quite easy to access. This should worry anyone that may have a weak password for something as important as their banking account. One limitation of the results is that I only worked with 20 passwords and only ran the attacks for 2 hours. So, for future work someone could expand on the number of passwords used or the duration of the attacks. Another limitation is the computer I ran the attacks on. My computer has an i9 10850k CPU overclocked to 5.2GHz on the core and 4.7GHz on the ring. Although my computer is quite powerful for a personal desktop, it still does not have the computing power of large servers. This limitation also opens future work to use more powerful machines to crack the passwords. Another limitation for this project is that for four of the passwords on bcrypt wordlist I had to prematurely stop the attack after just 13 minutes because I realized my CPU was reaching an uncomfortable temperature of 87 Celsius. Recently the new largest list of passwords has been posted on a hacking forum. The new list dubbed “Rock You2021” has 8.4 billion passwords, surpassing the previous largest password list of 3 billion [7]. These large lists of passwords are compiled from previous data breaches. So, they contain passwords people really use. Therefore, it would be a great idea to reset your password every 90 days [8]. One problem with password security is many people “severely overestimated the benefit of adding digits to passwords and underestimated the predictability of keyboard patterns and common phrases (e.g., “iloveyou”). In essence, participants did not realize how common these behaviors are” [9]. Another problem is that it has been found that even in articles published in respectable databases such as ACM there are misunderstandings about password security. In “Addressing Misconceptions About Password Security Effectively” they discovered that in 20 relevant publications there was 23 misconceptions about password security [10] Some of these misconceptions include “The inclusion of numbers makes passwords automatically more secure”, “A word from another language than the user’s mother tongue is a secure password”, and “Reusing passwords is OK for secure passwords, but should be avoided for weak passwords” [10].

In these results bcrypt can be seen as a much more secure way to hash a password compared to md5 due to the difference in time required to complete the hash. Additionally, a word list attack is significantly more successful at cracking a hashed passwords compared to an incremental or brute force method.

**Bibliography**

[1]

C. Yang, J.-L. Hung, and Z.-X. Lin, “Loose password security in Chinese cyber world left the front door wide open to hackers: an analytic view,” p. 6.

[2]

R. Bonnette, “What Makes Passwords Secure?,” *The Identity and Access Management Blog*, Mar. 07, 2017. <https://www.avatier.com/blog/what-makes-passwords-secure/> (accessed Oct. 01, 2021).

[3]

E. Rescorla, “A look at password security, Part I: history and background.” <https://blog.mozilla.org/en/mozilla/password-security-part-i/> (accessed Oct. 01, 2021).

[4]

A. Shakya and N. Karna, “Enhancing MD5 hash algorithm using symmetric key encryption,” in *Proceedings of the 3rd International Conference on Cryptography, Security and Privacy  - ICCSP ’19*, Kuala Lumpur, Malaysia, 2019, pp. 18–22. doi: [10.1145/3309074.3309087](https://doi.org/10.1145/3309074.3309087).

[5]

Z. Li, T. Li, and F. Zhu, “An Online Password Guessing Method Based on Big Data,” in *Proceedings of the 2019 3rd International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence*, Male Maldives, Mar. 2019, pp. 59–62. doi: [10.1145/3325773.3325779](https://doi.org/10.1145/3325773.3325779).

[6]

C. Gong and B. Behar, “UNDERSTANDING PASSWORD SECURITY THROUGH PASSWORD CRACKING,” p. 7, 2018.

[7]

G. Guthrie, “New 8.4 billion password hack breaks records,” Jun. 08, 2021. <https://www.consumeraffairs.com/news/new-84-billion-password-hack-breaks-records-060821.html> (accessed Oct. 01, 2021).

[8]

M. Pena, “8 Best Practices for Password Security,” *Liquid Web*, Oct. 26, 2020. <https://www.liquidweb.com/blog/password-security/> (accessed Oct. 01, 2021).

[9]

B. Ur, J. Bees, S. M. Segreti, L. Bauer, N. Christin, and L. F. Cranor, “Do Users’ Perceptions of Password Security Match Reality?,” in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, San Jose California USA, May 2016, pp. 3748–3760. doi: [10.1145/2858036.2858546](https://doi.org/10.1145/2858036.2858546).

[10]

P. Mayer and M. Volkamer, “Addressing misconceptions about password security effectively,” in *Proceedings of the 7th Workshop on Socio-Technical Aspects in Security and Trust - STAST ’17*, Orlando, Florida, 2018, pp. 16–27. doi: [10.1145/3167996.3167998](https://doi.org/10.1145/3167996.3167998).