# The Effects of Password Length and Complexity on Password Resiliency

Preston Navor
University Hawai'i – West O'ahu

## Abstract

Passwords serve as a primary method for users to access systems and accounts. Even malicious actors with minimal cyber security knowledge can break into a user's account if the password is weak enough and easy to guess. This study examines at the password length and complexity as variables for password strength. A total of 9 passwords of varying lengths and complexity were selected. As passwords are often stored as hashes, the hashes of these passwords will be computed and hashcat will be used to try to crack the passwords.

## Introduction & Research Question

Background

Alphanumeric passwords are one of the most common forms of authentication[1] and these passwords are usually user-generated.[2] To help guide users to create stronger passwords, developers create policies and guidelines to enforce stronger passwords. However, these policies and guidelines can be too strict and lead users to write down their passwords to help remember them.[3] The issue with the user-generated passwords is that users often do not know best practices for creating passwords.[4]

Research Question

How does password length and complexity affect password strength and its resiliency against brute force attacks?

## Research Design & Data Collection

This research was conducted within a Kali Linux virtual machine hosted on a Windows 10 system. The hashcat tool was used to crack the hashes of the passwords. Each password was run through hashcat in 3 different attack modes. The first attack mode is a dictionary attack. This utilizes a list of words or possible passwords and run their hashes against the passwords hash to see if it can quickly match the hashes. The second attack is the default brute force attack. This attack runs through a set mask (list of character sets) to see if it can crack the password. The last attack uses all possible characters and combinations to try to crack the password. The time it takes to crack the passwords will be compared between the different passwords as a measurement for password strength.

## Results

**Dictionary Attack:** Utilizing the rockyou text file, the dictionary attacks were able to immediately crack the passwords with low complexity regardless of length. The passwords with medium and high complexity could not be cracked by a pure dictionary attack.

**Default Mask:** The default mask brute force attack was able crack low complexity passwords for all 3 password lengths. The time to crack increased as password length increased. The low complexity 6-character password was cracked in 5 seconds, the low complexity 8-character password was cracked in 14 minutes and 55 seconds, and the low complexity 10-character password was cracked in 12 hours, 15 min, and 24 seconds. The attacked failed to crack any password with medium or high complexity. Although hashcat never fully attempted the medium and high complexity 10-character passwords, the estimated time to attempt all possible passwords were over 1 year.

**Full Brute Force:** The full brute force attack sets the hashcat program to try all possible characters and combinations for every password length to a specified point. The attack was able to successfully crack every 6-character password. The low complexity was cracked in 3 min 47 seconds, the medium complexity was cracked in 18 min 34 seconds, and the highest complexity was cracked in 29 minutes 24 seconds. 8-character passwords of any complexity estimated over 1 year to attempt all possibilities. Although the experiment was unable to determine the estimated time for any of the 10-character passwords, the difference between the 6-character passwords (less than 1 hour for each) and the 8-character passwords (estimating over 1 year), it is safe to assume a 10-character password will take significantly more than 1 year to attempt all possible passwords.

| Dictionary Attack | 6 Characters | 8 Characters | 10 Characters |
|---|---|---|---|
| Low Complexity | 2 seconds | 2 Seconds | 2 seconds |
| Medium Complexity | Failed to crack | Failed to crack | Failed to crack |
| High Complexity | Failed to crack | Failed to crack | Failed to crack |

| Default Attack | 6 Characters | 8 Characters | 10 Characters |
|---|---|---|---|
| Low Complexity | 5 seconds | 14 minutes<br>5 seconds | 12 hours<br>15 minutes<br>24 seconds |
| Medium Complexity | Failed to crack | Failed to crack | Estimated over 1 year |
| High Complexity | Failed to crack | Failed to crack | Estimated over 1 year |

| Full Brute Force Attack | 6 Characters | 8 Characters | 10 Characters |
|---|---|---|---|
| Low Complexity | 3 minutes<br>47 seconds | Estimated over 1 year | Estimated over 1 year |
| Medium Complexity | 18 minutes<br>34 seconds | Estimated over 1 year | Estimated over 1 year |
| High Complexity | 29 minutes<br>24 seconds | Estimated over 1 year | Estimated over 1 year |

## Discussion & Analysis

The experiment demonstrated that increased password length and complexity resulted in exponentially longer time to crack each password. Password length was more effective in increasing the time required to recover the password. Complexity helped protect against dictionary attacks which were able to quickly crack passwords on the wordlists. Dictionary attacks were able to quickly crack the low complexity passwords but were unsuccessful with medium and high complexity passwords. Something to note that is not seen in the study is how fast hashcat processed all possibilities of passwords that were 5 or less characters. When conducting the default mask and full brute force tests, the program provides an update and information after eliminating all possibilities for each password length. During every test, the program would eliminate all possibilities that were 5 or less characters long in about 1 minute. The full brute force attack took the longest but had the most successful attempts. This is because it tries every possible combination so it can and will crack any password given enough resources and time. The purpose of password length is to increase the amount of possibilities making this attack method impractical.

## Conclusions

Although the industry standard of an 8 character password containing 1 upper case, 1 lower case, 1 number, and 1 special character makes for a strong password, an attacker can crack it within a reasonable time. This is dangerous if the password is protecting sensitive information or if the account is integrated into a system with sensitive information. This does not include other security measures that should be considered when designing password security. Proper security implementations can include techniques such as salting to help strengthen passwords and prevent dictionary attacks even if the user develops a weak password. As a user, it is important to keep in mind while you cannot develop the software itself, you can still create a strong password to help yourself or your company defend from attackers should they gain access to the database of passwords. As a system developer or security manager it is important to train users and inform them of the threats against weak passwords and how to defend against them. Setting up a strong policy that requires a minimum password length and complexity also ensures that passwords cannot be easily brute forced.

## References

1. Herley, C., Van Oorschot, P. C., & Patrick, A. S. (2009, February). Passwords: If we're so smart, why are we still using them?. In *International Conference on Financial Cryptography and Data Security* (pp. 230-237). Springer, Berlin, Heidelberg.
2. Adams, A., & Sasse, M. A. (1999). Users are not the enemy. *Communications of the ACM*, *42*(12), 40-46.
3. Brumen, B. (2020). System-Assigned Passwords: The Disadvantages of the Strict Password Management Policies. *Informatica*, *31*(3), 459-479.
4. Yıldırım, M., & Mackie, I. (2019). Encouraging users to improve password security and memorability. *International Journal of Information Security*, *18*(6), 741-759.