To guess passwords I used the hashcat utility and passwords dictionary Common credentials list top 10-million. In hashcat I used attack modes: Dictionary attack, Mask attack и Rule-based attack.

1. Set-3 (MD5)

Dictionary attack	
Estimated time	6 min, 15 secs
Speed	~700 MH/s
Recovered	88067 / 175911 (50.06 %)
Notes	
A larger dictionary (or the one that was used for generation) would give more decrypted passwords (90%) according to the description of password generation.	

Rule-based attack	
Estimated time	7 min, 33 secs
Speed	~10400 kH/s
Recovered	132961 / 175911 (75.58 %)
Notes	
Instead of widespread rule sets, a set of rules was used based on the description of the generation algorithm.	

Mask attack	
Estimated time	4 days, 16 hours (8 symbols)
Speed	~850 MH/s
Recovered	> 75.58 % (Probably)
Notes	
With more video memory, it would be possible to iterate passwords up to ~10 characters long in a reasonable amount of time.	

Without salt, the MD5 algorithm iterates too quickly. This process can be accelerated using Rainbow tables. The table for loweralpha-numeric is 1-10 characters long and weighs only 316 GB.

2. Set-1 (SHA1 + salt)

Dictionary attack	
Estimated time	1 hour, 42 mins
Speed	~12900 kH/s
Recovered	102121 / 190000 (53.75 %)
Notes	
A larger dictionary (or the one that was used for generation) would give more decrypted passwords(90%) according to the description of password generation. The presence of salt increased the number of hashes to be searched.	

Rule-based attack	
Estimated time	29 days
Speed	~160 MH/s
Recovered	~ 75 %
Notes	
Instead of common rule sets, a set of rules based on the description algorithm. The expected amount of time is too high, as fewer salts will need to be tried as passwords are found. This way it will be possible to restore were the same as for MD5.	

The presence of the salt discards the use of Rainbow tables. Brute-force and Rule-based attacks are still possible if you have the required amount of the video cards due to the speed of SHA1 calculations, but you will have to brute-force passwords for each salt, but they are not repeated.

3. Set-1 (bcrypt)

Dictionary attack	
Estimated time	2 years, 21 days
Speed	~2900 H/s
Recovered	~50 %
Notes	
For a dictionary with 10 million values, searching on a regular PC takes an unreasonable amount of time. The search for the top 100 passwords would take 2 days.	

This algorithm is resistant to Brute-force attacks as it requires a lot of computational resources. It also makes it impossible for Dictionary and Rule-based attacks on non-specialized hardware. Prohibiting users from using common passwords and changing passwords from time to time will make the Dictionary almost impossible, and the Rule-based attack greatly complicated.

4. Summary

Algorithms that require a lot of resources for calculations or that have a cost parameter should be used for hashing. It is also imperative to use a random and sufficiently long (at least 16 characters long) salt. This will deprive attackers of the opportunity for Brute-force attacks. It will also greatly slow down Dictionary and Rule-based attacks, which can be dealt with in other ways. Examples of such algorithms are: argon2, bcrypt, scrypt.