To: Management at XXX company

From: Student

Date: 29-03-2021

Subject: Crack leaked password database findings

Madam/Sir

Below is a report of my findings after decrypting the password file. I had used Hashcat, an open-source password recovery utility, to crack passwords in the hash.txt file with various attacks such as brute-force, combinator, and dictionary.

1. What type of hashing algorithm was used to protect passwords?

The **MD5 hash function**, which is designed as a cryptographic function has been used to encrypt the passwords

1. What level of protection does the mechanism offer for passwords?

**MD5** is a cryptographic algorithm, often used to store passwords in a database. MD5 hash function maps a set of data to a bit string of a fixed size, the hash value, which is 128 bits. It verifies that a file has been unaltered by producing a digit representing the sum of the correct digits in the password before and after, and then compares those digits. However due to brute force attacks are faster than ever, dictionary tables are big and two distinct messages can hash to the same value making it quite vulnerable.

1. What controls could be implemented to make cracking much harder for the hacker in the event of a password database leaking again?
2. **Anti-debug methods**. Checks the presence of tools used by crackers before the program is run. If the tools are detected the program is suspended. Checks for debuggers and breakpoints.
3. **Extra protected file**. Encrypt and compress the executable file of an application so that when there is any attempted modification the file will not run. After protection, a special code is added to the structure.
4. **Key stretching algorithms**. Significantly reduces the rate at which passwords can be tested by converting a password to a longer and more random key. Can be done by applying a cryptographic hash function or block cypher repeatedly in a loop, and using cryptographic hash functions that have large memory requirements.
5. **Salting**. A salt is randomly generated for each password and concatenated and processed with a cryptographic hash function. Salt must not be reused or too short. Makes it more challenging for a hacker to use lookup, reverse lookup, and rainbow tables.
6. **Using a more secure encryption hash**. Ideally, a hash function should return no collisions but MD5 does have many collisions. Algorithms like SHA-2, SHA-3 are considered more secure than MD5 and SHA-1. One can also use deliberately **slow hash constructions**, such as scrypt, bcrypt and PBKDF2.
7. What can you tell about the organization’s password policy (e.g. password length, key space, etc.)?

The organisations current password policy does not require special characters, capital letters, or numbers, and the minimum length seems to be 6 characters. Repeated numbers are allowed, and the policy does not discourage common passwords such as “password”, “qwerty”, and “qazxsw”.

1. What would you change in the password policy to make breaking the passwords harder?
   1. A minimum of 10 characters
   2. At least 1 special character, number and capital letter
   3. No repeated numbers
   4. Common passwords that include “password” or “qwerty” or those which include the name of the current application/site not allowed
   5. Warning to the user when years or a location are e.g. “Mumbai” or “2020”
   6. Encourage changing the passwords every 6 months

I hope this report will prove helpful in meeting your needs.