TITLE: HASHING PROJECT

PREPARED BY: WENDELL AKUBIA NYARKO

Table of Contents

Objective

To try and reverse a particular hash given.

Executive Summary

To provide the processes and methodologies involved that were used to find the hash that was given.

Introduction

The aim of this project was: Can you reverse a particular SHA3-512 hash? If not, just write the reason).

Even though hashes are mathematically irreversible, I had to find a way to a way to reverse the hash. The hash given was:
a2099f4c2c2de141afb474dfe4b765ce83448100e77f4359314d94807b00862d53316c03963fc60cbdb
d7bc6915778f1830f0f4fd9364a4bc71a09c5e83a0a67.

The only way hashes can be reversed is when we know the particular text, word, password or even the software that was used to generate the hash, the hash generated will then be compared to the original hash of the particular product. If the hashes are both the same, it means the product given was the valid product.

Methodology

There is a file called rock you text file. This is a plain text file that contains a list of commonly used password. This file contains over 14,341,564 passwords that were previously leaked in data

breaches. This file is commonly used for brute force or dictionary attacks on web applications. Anyone who wants to hack an account can use a rockyou file.

So using the rockyou file I developed a python program:

- To first find the SHA 3 512 hash of the passwords and then compare those hashes with the hash that was given.
- If the program finds that any of the hashes are the same, It will output that particular word that has that hash.

```python
import hashlib

given_hash = "a2099f4c2c2de141afb474dfe4b765ce83448100e77f4359314d94807b00862d53316c03963fc60cbdbd7bc6915778f1830f0f4fd9364a4bc71a09c5e83a0

with open("rockyou.txt") as file: #reads the lines of files in the rockyou text

    for line in file:

        word = line.strip() #to read through the respective lines in the file

        hashed_word = hashlib.sha3_512(word.encode()).hexdigest() #to hash the words to SHA3 512

        # code:
        if hashed_word == given_hash:

            print(word)
```

- After running the program, the word that was printed as the output was lovely.

Vulnerability Assessment

If a particular user uses this type of commonly used passwords, it makes your credentials very vulnerable and your device is susceptible to an attack.

Recommendations

- Prevent the use of commonly and easily guessable passwords, use strong passwords
- Use strong passwords that have a minimum of eight characters, a combination of uppercase letters, numbers and symbols.

Conclusion

I was successfully able to reverse the hash because it was a commonly used password that was found in a particular data breached file.

References

Python.

Google.