Homework 1
Due date: 11:59 pm 2/10/2024

Name:
ID:

1. Overview

The attacker can apply the bruteforce attack and intelligent search attack to guess your password. The one-way function (cryptographic hash function) takes the password and converts it to a fixed-size string (hash value). However, it is impossible to convert it back. Then the one-way function property can guarantee the security of your password in some applications. Each distinct password produces a distinct hash value. It means that even the similar data produces very different hashes. This allows you to be sure that the starting data are the same if two hashes are identical. The objective of this lab is for students to gain practical insights into the different attacks, security of one-way functions and learn how to use them.

There are many well-known cryptographic hash functions (e.g., MD5, SHA1, SHA2-256, SHA2-512) in cybersecurity and existing programming libraries to be used. This lab covers the following topics:

- Implementation of hashing your message with existing libraries
- Implementation of bruteforce attack

Lab environment. There is no specific requirement of the programming language. You can use any programming language (e.g., Java, C/C++, and Python). The example will be illustrated by Python.

Resources:

https://docs.python.org/3/library/hashlib.html

https://www.tutorialspoint.com/java_cryptography/java_cryptography_message_digest.htm

2. Example

There are many hash functions defined in the "hashlib" library in python. This example deals with explanation and working of MD5 hash. This hash function accepts sequence of bytes (input can be any size) and returns 128 bit hash value (output is always 128 bits). Functions associated :

- **encode()**: Converts the string into bytes to be acceptable by hash function.
- **hexdigest():** Returns the encoded data in hexadecimal format.

```
1  import hashlib
2
3  # initializing string
4  str2hash = "Security"
5
6  # encoding original string using encode()
7  # then sending to md5()
8  result = hashlib.md5(str2hash.encode())
9
10 # printing the equivalent hexadecimal value.
11 print("The hexadecimal equivalent of hash is : ", end ="")
12 print(result.hexdigest())
```

3. Tasks

You will get familiar with the attacks for guessing passwords. Alice sets a password for her account. Now you should crack her password with different settings. Please write the programs to achieve the **brute force attack** and get Alice's password. You need to call the **Check(string) function** to check if your find the correct answer.

Task 1: The hash value of her password by MD5 is 94d9e03c11395841301a7aee967864ec. You know the password length is 14. (The password can be combined with A-Z, a-z, or 0-9)

Task2: The hash value of her password by MD5 is f593def02f37f3a6d57bcbc9480a3316. You know the specific password format: <4-UPPER CASE LETTER><3-DIGIT-NUMBER><4-SMALLER CASE LETTER> (e.g., XYEX234ascf).

Task3: The hash value of her password by MD5 is bfb2c12706757b8324368de6a365338b. You know the specific password format: the length is 11 and it includes 7 upper case letters and the number "1234". But you don't know the position of "1234".

Task4: Please compare the attacking time of above three tasks and discuss the difficulty level to attack them. (which one is easiest and which one is hardest and reason)

```python
import hashlib

def Bruteforce():
    #write your code here and in


def Check(string): # Change the hash value for different tasks
    gussedpwd = hashlib.md5(string.encode())
    if gussedpwd.hexdigest() == "94d9e03c11395841301a7aee967864ec":
        print("You succeed and password is", string)
    else:
        print("Not correct")

Bruteforce()
```

4. Submission

Please submit two files in Canvas separately. One is the ".pdf" report of your homework. In the report, you should have very clear answer for each task (e.g., screen shots of your bruteforce attack code, Alice's password). Another one is your source code.