CS 4351: Computer Security **Instructor: Dr. Deepak Tosh**
Fall 2022 **Due Date:** 11/16/2022, Midnight

ASSIGNMENT# 1

INTRODUCTION:

The purpose of this assignment is to let the students to be familiar with password management, message digests, the concept of hashing a password, salting technique, and cracking passwords with a brute force or dictionary approach.

In this assignment, you will write programs (recommended to execute the programs in **Linux/Unix based Operating Systems**) that will crack as many of the passwords as you can. You will also write a report. This assignment includes a sample password files for each type of password file, and dictionary file.

- **You are not allowed to use any existing password cracker programs instead build your own little password cracker.**
- **It is not recommended to use Azure cloud for this assignment. Instead use your own computer/laptop as this assignment requires more compute time.**

Problem# 1: [30%]

The password file (**shadowfile.txt**) was generated with creation of 200 users (crack01—crack200) with password taken from random English dictionary words and some commonly used passwords. We downloaded a dictionary file from the Internet, named **commonPasswdFile.txt**, containing about 100,000 entries. You should be able to find similar dictionary files by searching in Internet. The password file contains for each user: username, salt, hashed password, and some more details.

The goals in this problem are to do the following activities:

a. List all the attributes that are stored for an individual user in a shadow file.
b. Where do the passwords in Windows system get stored?
c. A Python program (must be well documented) that uses the **commonPasswdFile.txt** and compute the necessary hash of the passwords to compare them with the passwords stored in shadow file.
   - The program should output all the **username:password** combinations after successful crack. Please list all the cracked combinations in the report.

Note: To make your testing easier, I am providing the password for 1 user here.
Testing Sample: *Username*: crack01 and *password*: 123456

Things to submit for problem# 1:

I. Write answers to (a) and (b).
II. Report showing evidence of password cracking. The report must include snapshots of the program's output (b).

III. Well documented/commented Python program
IV. Instructions to test the python program

Problem# 2a: [30%]

In this part, imagine you have 500 password hashes from a password database of a web service (extracted to the file "*UnsaltedPassTable.txt*"). Assume that the users are not really familiar with password security and create their passwords in the following ways:

1. An English word as the password (e.g. "secret")
2. A string of up to 8-digits as password (e.g. "87654321")
3. An English word followed by some digits, but together no more than 10 characters (e.g. password89)
4. Concatenate two English words together (e.g. "secretpassword")

When choosing English words, users only use the words from the provided dictionary of 10,000 most common English words ("*words.txt*").

**Hint: Users are not really inclined to use very short words, try passwords of length>5.

Testing Sample: *Username*: user1 and *password*: revenge234

Problem# 2b: [40%]

Imagine, you got access to another password table from the same web server, but this contains usernames and password of 100 VIP users (extracted to the file "*SaltedPassTable.txt*"). The website has SALTED the password hashes to provide better security, however the users still created their passwords based on same way as discussed above. There are only 100 password hashes in this table, and the salt values are provided alongside the hashes.

The goals in the above two problems is to

a. Find all the cracked passwords.
b. Report the time taken to crack for all the users and provide your remarks on why they are different.

Testing Sample: *Username*: user1 and *password*: 16083058

Things to submit for problem# 2a, 2b:

I. Report showing evidence of password cracking. The report must include snapshots of the program's output.
II. Well documented/commented Python program
III. Instructions to test the python program

Instructions for Homework Files Download

You can download the necessary files from the following Google Drive link:
https://drive.google.com/drive/folders/1Xk0AoUf_V_mdt22YvcIMJ9YiZqWPUmpx?usp=sharing

Inside the drive, there are 3 directories for each problem. Under Problem-1 directory, there are two files named **shadowfile.txt** and **commonPasswdFile.txt** which you will be using to solve the Problem# 1. Under Problem-2, I have provided 3 files. (1) "*words.txt*" which contains 10,000 English words that people use to create their passwords. (2) *"UnsaltedPassTable.txt"* contains the password hashes you need to crack for Part 2a, with "{username}:{hash}" on each line. For example, if the username is "john" and password hash is 7e23..2b41, then that line will be:

john:7e238a22c982f0d9de093fc7bca92b41

(3) "*SaltedPassTable.txt*" contains the password hashes for Part 2b. It has username, salt, and hashed password on each line, separated by ":". Using the same username and password as above, with a salt value 67ef98c0, the corresponding line in the password table is:

john: 67ef98c0:55457bde7a1a4816ba636d09017ff30a

Grading:

The assignment will be graded on following items:

1. Execution evidences
2. Programs and their correctness
3. # of passwords cracked
4. Programs' readability
5. Clarity of Report with all the required responses

Submission:

Please submit your reports in PDF format (lastname_firstname_assign1.pdf) and codes in single ZIP file on **BLACKBOARD only**. Make sure to answer the questions in order and CLEARLY STATE YOUR ASSUMPTIONS, if you are unsure about something.

Note: You are open to download your own dictionary file for password cracking.