Assignment-3

This **assignment-3** is a team/group assignment to foster student's ability to work in a team by following a common set of rules to achieve common goals and learn secure programming.

Goals:

The goals of this group assignment are as follows:

- To learn a ***System Login and Credential Management (SLCM)*** module of an enterprise software system. A robust SLCM is at the heart of strong authentication and authorization. This assignment is about setting up a programming environment and develops a small version of SLCM.

  Other goals are:

- To foster discussion and learning on the topics of the course discussed during an entire week.
- To encourage using a collaboratory tool such as a github.com repository that can be shared and edited remotely by multiple members.
- To encourage using a programming collaboratory tool such as Slack, Zoom, and email to communicate and effectively discuss with each other.
- To bolster skills in documenting a software project

  Inside your Pycharm project you created for ***assignment-****2 (in previous weeks)*, download and copy a skeletal python module named ***SecureAuth.py*** from Instructor's repository **301-sp** in github.com.

  - A complete repository can be downloaded as a zip file through https://github.com/convexnaresh/301-sp/archive/main.zip

  The module has two major functions/methods with the stated inputs, logic, and outputs. Your job is to implement the following functions as described:

1. A function named ***secure_hashed_passwd(username, passwd)***

   i) Receives two parameters—***username*** and plain text password (***passwd***), as string parameters.
   ii) Use ***salt*** *and* ***pepper*** to hash ***passwd*** using SHA-3-224 algorithm, and return hex version of the ***salt, pepper*** *and* ***hash value***

2. A function named ***verify_hashed_passwd(username, hpasswd)***

   i) Receives two parameters—***username*** and plain text password ***(passwd)*** as string arguments.
   ii) Search and retrieve the row in ***hlogins.dat*** that matches with ***username*** received as an argument of this function.

a. If no row matches the username, then display a message of "Authentication unsuccessful!" and return False
b. If a row matches the username, then compute a temporary hash value ***(tempo_hash)*** of ***passwd*** using retrieved salt and pepper using sha-3-224 algorithm.

iii) Check if ***tempo_hash*** matches with ***hpasswd,***

a. Displays message of "Authentication successful" if the two values match, and return True; otherwise, display a message "Authentication unsuccessful!" and return **False**.

Example:

- Assume that the function receives "shyamal@gmail.com" and 'abcTextPasword" as two parameters.
- You need to search in the file **hlogins.dat** for the row matching its first field to "shyamal@gmail.com"
- If a row is found, then you have to get the row's second field value and set it to a '**salt'** variable; get row's third field value and set it to a "**pepper**" variable; and get row's fourth value and set it to a "*stored_hpasswd*". See below the row structure of **hlogins.dat**
- Compute "tempo_hash" value of "passwd"
- Compare "tempo_hash" value with "stored_hpasswd" variable. If they matches then return True; otherwise, return False.

The main function **main() (already written for you, do not modify this function)**

With lists of username and password, the two functions are tested to produce **hlogins.dat**

i) Invokes function ***secure_hashed_passwd*** **()**
ii) Invokes function ***verify_hashed_passwd*** **()**

After you implement the **function 1**, your expected contents of the file **hlogins.dat** must be something like this; each row of this form.

Fields values: *first, second, third, fourth, $*

Fields values: *username, salt, pepper, sha 224-hashed-password,$*

shyamal@gmail.com,30c4f3f965764f4eb475d6e7814d0655,9eeb09b05dcdec2163fe391551235902,601ca18e595500b65990d9baea4024bbca1d32af3dae2f56c579e3a3,$
lifegivesalot@protonmail.com,37019cf0049c430d9c8c515966e4a071,581f7a437ed9a09164210f3ece18a1e9,77fddd95c17d4c5bc303d7404a31ad7179a1422446441ffe6f0bf99a,$
ghana@makai.com,9674f509819043899c84065ee15f3034,dfcd5492060e41b8d6f9305eada13517,cd265c71cd4312b5c49e8a964a2b8df1f0cb5cbcab47d8e5a050e038,$
david@inst.edu,9603c5b26d9f42bf8c3e97b633f8247d,7bb7dd5674c08572b2727f021aa4c1e3,0e543019ac80bd0addc849d3bd30b038dd51a34139a8313500e0e1f0,$
buttlerbusiness@sru.edu,7b75478a82b24725a6c8d7f6e8080003,73031c7ae9dc62a39ab99a30926e5044,b6e04cb9c2e4a0fc6bbb21d7edc5928452ed6aefa76e7f5ecef61885,$

After you implement the function 2, your expected output is:

- Login succesful for user shyamal@gmail.com
- <!> Login failed for user brutforce@yahoo.com
- Login succesful for user lifegivesalot@protonmail.com
- <!> Login failed for user rainbow@sru.edu
- Login succesful for user ghana@makai.com
- Login succesful for user david@inst.edu
- Login succesful for user buttlerbusiness@sru.edu
- <!> Login failed for user myChurch45@state.edu

Deliverables for this group assignment

Each group should create its group's repository on github.com and commit final versions of the following two files into the same repository:

1. The program file ***SecureAuth*.py** with the above functions and implementations in python.
2. The ***hlogins.dat*** created by the above function to store all valid usernames and passwords, as listed above.

To have your submissions graded, please add your instructor ***(convex.naresh@gmail.com)*** as one of the project's collaborators. Your instructor can download and execute your programs to grade them. No modification to files in the repository is permitted after the due date of this assignment. On your D2L dropbox for this assignment, please submit the ***HTTPS URL*** of your repository.

In case you have a problem, any single member of your group should submit the above two files in your D2L dropbox for this assignment.

Grading Criteria:

1. Creating and sharing a project repository for this assignment with instructor's email address ***convex.naresh@gmail.com*** **+40 pts**. The repository must contain two files:
   a. ***SecureAuth.py***
   b. ***hlogins.dat***

   Note:
   - It does not apply to those who have already shared a GitHub repository with the instructor while doing assignment-1. However, the same repository must contain the python program files.
   - To secure a full credit, the repository must be updated with the final program version by the due date of this assignment.
2. File names as specified format, program heading, proper spacings, comments, and documentation of the project, **+50 pts.**
3. Function ***secure_hashed_passwd()***
   a. Correct function names, parameters and correct returns (if any) **(10pts)**
   b. Correct logic and implementation **(40pts)**
4. Function ***verify_hashed_passwd*** **()**

a. Correct function names, parameters and correct returns (if any) **(10pts)**
b. Correct logic and implementation **(40pts)**

Total: 190 pts to be scaled to 100.

For details, refer the **rubrics** attached with the assignment-3.

R*emember to agree on, check or verify what your group member submits if anything wrong will affect all the members equally.*

Good Luck