Password Cracking by Dictionary Attack

2071008 SeoyeonKim

Overview

This Python script is designed to crack hashed passwords using a dictionary attack. There are mainly two outputs: what a salt is and
what is original password corresponding to each hashed password. It reads hashed passwords from a file, along with a dictionary of potential plain-text passwords. It then attempts to crack each hashed password by comparing it against the hash of each password in the dictionary. finally, we find 4-character salt and original password corresponding to any specific hashed passwords in given hashed files.

This can be implemented by 2 steps.

- Step 1 : **we firstly find unknown salt.** we just need certain plain password and hashed passwords. we can get all possible salts using `generate_salt()`. Then, we generates hashes for the given plain password using each possible salts and checks if any new hash match the given hashed passwords. If matches, we can conclude that the original salt is the salt that makes that new hash. we can use Python Dicionary to save new hashes and reduce time for search.

- Step 2 : with the salt Known, we can simply **crack original password for each hashed password**. The Purpose is to find original password X where satisfy H(X) == H(X'). we just iterates through each hashed password H(X) and compares it against each new hash H(X') of plain-text password and salt. If hashed password matches H(X) with certain new hash H(X'), the original password X for the hash is one that makes that new hash, X'.

Dependencies

The script uses the following Python libraries:

- `hashlib`: Used to generate MD5 hashes of passwords.
- `itertools.product`: Used to generate all possible combinations of characters to create salt.

You don't need to install any dependencies since every libraries used in this script is default installed with python3.

Files

- `1MillionPassword_hashed.txt`: Contains a list of hashed passwords.
- `1MillionPassword_wordlist.txt`: Contains a dictionary of all potential passwords.

Functions

`read_files(file_path)`

- **Input**: `file_path` - Path to the file to be read.
- **Output**: List of lines read from the file.
- **Functionality**: Opens the specified file and reads its contents into a list, removing newline characters.

`generate_md5_hash(plain_password, salt)`

- **Input**: `plain_password` - The plain-text password to be hashed. `salt` - The salt to be appended to the password before hashing.
- **Output**: MD5 hash of the salted password.
- **Functionality**: Appends the salt to the plain password, hashes it using MD5, and returns the hashed value.

`generate_salts()`

- **Output**: List of potential salts.
- **Functionality**: Generating all combinations of 4-character strings consisting of lowercase letters or numbers

`find_salt(plain_password, salt_list, hashes)`

- **Input**: `plain_password` - An any plain-text password in dictionary used to generate hash. `salt_list` - A list of potential salts. `hashes` - A list of hashed passwords.
- **Output**: The salt used to hash a given plain password.
- **Functionality**: Generates hashes for the given plain password using each salt in the list and checks if any new hash match the hashed passwords in the provided.

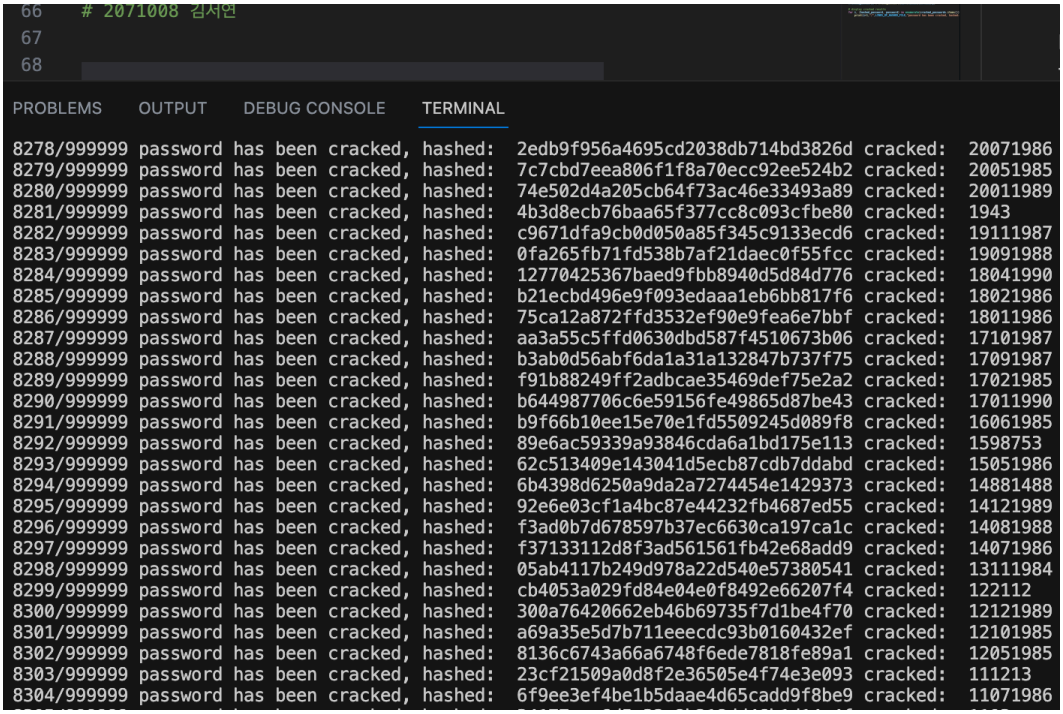
`crack_password(salt, hashes, dictionary)`

- **Input**: `salt` - The salt used for hashing passwords. `hashes` - A list of hashed passwords. `dictionary` - A list of potential plain-text passwords.
- **Output**: A dictionary mapping hashed passwords to their cracked plain-text equivalents.
- **Functionality**: It iterates through each hashed password and attempts to match it with a password from the dictionary, by generating new hash. If a match is found, the original password is stored.

Execution

1. Reads the hashed passwords and dictionary from their respective files.
2. Generates a list of potential salts.
3. Finds the salt used to hash the first password in the dictionary.
4. Attempts to crack each hashed password using the dictionary attack approach.
5. Displays the cracked passwords along with their corresponding hashed values.
6.

Result


66 # 2071008 김서연
67
68
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
8278/999999 password has been cracked, hashed: 2edb9f956a4695cd2038db714bd3826d cracked: 20071986
8279/999999 password has been cracked, hashed: 7c7cbd7eea806f1f8a70ecc92ee524b2 cracked: 20051985
8280/999999 password has been cracked, hashed: 74e502d4a205cb64f73ac46e33493a89 cracked: 20011989
8281/999999 password has been cracked, hashed: 4b3d8ecb76baa65f377cc8c093cfbe80 cracked: 1943
8282/999999 password has been cracked, hashed: c9671dfa9cb0d050a85f345c9133ecd6 cracked: 19111987
8283/999999 password has been cracked, hashed: 0fa265fb71fd538b7af21daec0f55fcc cracked: 19091988
8284/999999 password has been cracked, hashed: 12770425367baed9fbb8940d5d84d776 cracked: 18041990
8285/999999 password has been cracked, hashed: b21ecbd496e9f093edaaa1eb6bb817f6 cracked: 18021986
8286/999999 password has been cracked, hashed: 75ca12a872ffd3532ef90e9fea6e7bbf cracked: 18011986
8287/999999 password has been cracked, hashed: aa3a55c5ffd0630dbd587f4510673b06 cracked: 17101987
8288/999999 password has been cracked, hashed: b3ab0d56abf6da1a31a132847b737f75 cracked: 17091987
8289/999999 password has been cracked, hashed: f91b88249ff2adbcae35469def75e2a2 cracked: 17021985
8290/999999 password has been cracked, hashed: b644987706c6e59156fe49865d87be43 cracked: 17011990
8291/999999 password has been cracked, hashed: b9f66b10ee15e70e1fd5509245d089f8 cracked: 16061985
8292/999999 password has been cracked, hashed: 89e6ac59339a93846cda6a1bd175e113 cracked: 1598753
8293/999999 password has been cracked, hashed: 62c513409e143041d5ecb87cdb7ddabd cracked: 15051986
8294/999999 password has been cracked, hashed: 6b4398d6250a9da2a7274454e1429373 cracked: 14881488
8295/999999 password has been cracked, hashed: 92e6e03cf1a4bc87e44232fb4687ed55 cracked: 14121989
8296/999999 password has been cracked, hashed: f3ad0b7d678597b37ec6630ca197ca1c cracked: 14081988
8297/999999 password has been cracked, hashed: f37133112d8f3ad561561fb42e68add9 cracked: 14071986
8298/999999 password has been cracked, hashed: 05ab4117b249d978a22d540e57380541 cracked: 13111984
8299/999999 password has been cracked, hashed: cb4053a029fd84e04e0f8492e66207f4 cracked: 122112
8300/999999 password has been cracked, hashed: 300a76420662eb46b69735f7d1be4f70 cracked: 12121989
8301/999999 password has been cracked, hashed: a69a35e5d7b711eeecdc93b0160432ef cracked: 12101985
8302/999999 password has been cracked, hashed: 8136c6743a66a6748f6ede7818fe89a1 cracked: 12051985
8303/999999 password has been cracked, hashed: 23cf21509a0d8f2e36505e4f74e3e093 cracked: 111213
8304/999999 password has been cracked, hashed: 6f9ee3ef4be1b5daae4d65cadd9f8be9 cracked: 11071986

Caption