### Hash Cracker / Password Cracker Documentation

This Python script demonstrates a basic password cracker that attempts to crack hashed passwords using a list of possible passwords (wordlist) and various hashing algorithms (MD5, SHA-1, SHA-256). The script defines a HashCracker class and provides functionality to read hashed passwords, read a wordlist of potential passwords, and attempt to crack the passwords. Below is a detailed explanation of how the code works:

#### Code Structure and Functionality:

The code is organized into a class, HashCracker, which encapsulates the password cracking logic. The class includes the following methods:

1. HashCracker **Class**:
   * The class represents a password cracking operation.
   * Constructor (\_\_init\_\_): Initializes the paths to the hash file, password file (wordlist), and the file to save the cracked password.
2. hash\_file **Method**:
   * Reads the hash file, which contains hashed passwords to be cracked.
   * Splits the file into a list of hashed passwords.
   * Handles exceptions and prints a message if the file is not found.
3. wordlist\_file **Method**:
   * Reads the password file (wordlist), which contains potential passwords to be tested.
   * Splits the file into a list of potential passwords.
   * Handles exceptions and prints a message if the file is not found.
4. hash\_crack **Method**:
   * Iterates through each potential password and hashing algorithm (MD5, SHA-1, SHA-256).
   * Computes the hash of the potential password using the selected algorithm.
   * Compares the computed hash with the list of hashed passwords.
   * If a match is found, the cracked password is saved to a file, and the process stops.
   * If no matches are found, a message is printed indicating that the password cracking attempt is completed.
5. **Main Section (**if \_\_name\_\_ == "\_\_main\_\_":**)**:
   * Defines the paths to the hash file, password file (wordlist), and the file to save the cracked password.
   * Creates a HashCracker object with the specified paths.
   * Calls the hash\_file method to read the hash file.
   * Calls the wordlist\_file method to read the wordlist.
   * Calls the hash\_crack method to attempt password cracking.

#### How the Code Works:

1. The script defines the HashCracker class and its methods.
2. In the main section:
   * Paths to the hash file (pwd\_hash.txt), password file (wordlist), and the file to save the cracked password (cracked\_pass.txt) are specified.
3. A HashCracker object is created with the specified paths.
4. The script calls the hash\_file method to read the hash file, and the wordlist\_file method to read the wordlist.
5. The hash\_crack method is called to attempt password cracking:
   * For each potential password in the wordlist and each hashing algorithm, the script computes the hash.
   * If a match is found with one of the hashed passwords, the cracked password is saved, and the process stops.
   * If no matches are found, a message is printed indicating that the password cracking attempt is completed.
6. The script handles exceptions and prints error messages when necessary.

This script provides a basic implementation of a password cracker. It can be further enhanced with additional features and algorithms for more robust password cracking capabilities. Please note that password cracking should only be performed for legitimate and authorized purposes, such as security testing or recovery of forgotten passwords. Unauthorized use of password cracking tools is illegal and unethical.