Assignment 1 – CSCI 397

Instructions and Overview

- **Due Date:** Friday, October 15, 11:59 pm EST
- This is an individual assignment**.** You are allowed to discuss your approach with a partner; however, you must submit your own work on Canvas. You may discuss general ideas and approaches but **NOT** exact solutions. If you work with someone, make sure to mention their name along with your submission.
- If you plan to use any late days for the assignment, make sure to submit the details on the Late Days Google Form provided on Canvas.
- You are required to write your solution in Python programming language, version 3.5 or later. Make sure that your code should be properly documented and readable.
- There are a total of 4 tasks in this assignment, which cover the contents from lectures 1 through 7. Make sure to read the complete the complete assignment before attempting.
- Text highlighted in **`blue`** are names of files provided, or names of modules. Text highlighted in **`red`** are the names of files you need to submit.

Task 1: Security Review **[30 Points]**

Developing a ***security mindset*** is one of the integral goals of this course. This involves being able to observe the technologies around us and view them through the lens of security and privacy. Cybersecurity professionals routinely perform security reviews for systems and technologies used by various organizations. In this task, you are required to perform a security review of a relatively new technology of your choice. Your security review should contain the following details:

- **A summary of the technology** that you are evaluating. You may to choose a single product (Google Nest) of a general category of products (wearable fitness devices). This summary should be approximately 100 words.
- Explain **two security goals** that are essential for the correct operation of this technology. It may be helpful to recall the security properties we talked about in class (confidentiality, integrity etc..) and trying to envision how these properties are implemented in context of the technology to achieve certain security goals. Describe each goal in 2-3 sentences.
- Discuss **two potential threats** that surround the use of the technology. A threat is defined as an action by an adversary that can compromise the technology. Threats can be due to a specific design, or a given potential risk or weakness. Remember that different threat adversaries have different capabilities. Describe each threat in 2-3 sentences.
- State **two defenses** against that the technology could use or might already be using to address the potential threats in the bullet you described above. Describe each defense in 2-3 sentences.

- Finally, provide a **conclusion** that includes a thoughtful reflection on your responses above. Also, discuss the relevant "big picture" issues, ethics, and the future evolution of that technology. Your conclusion should be around 50 words.

You may write the review in an editor of your choice. To submit, export your review as a PDF and save it as a file named **`review.txt`**.

Task 2: Breaking Substitution Ciphers [35 Points]

Substitution ciphers are a class of ciphers in cryptography (the science of writing, analyzing, and deciphering codes) which convert standard language or plaintext into coded language or ciphertext, by replacing units of plaintext in accordance with a fixed set of rules, knows as the key.

For this task, you have been provided with a file **`encrypted.txt`** that contains ciphertext generated from English plaintext. Your task is to write a Python program named **`analysis.py`** that takes this ciphertext as an input and performs a frequency analysis attack to discover the key and corresponding plaintext.

You will be required to use an English language frequency table for this task which you can easily obtain online. Note that not all pieces of texts have the same exact frequency distribution of alphabets. For instance, while Wikipedia states that the percentage of letter O (7.5%) is greater than N (6.7%), it's possible that a given piece of text may have more occurrences of the letter "N" that the letter "O". Due to this, you may have to perform some manual substitutions in the key. If you perform any manual substitutions in the key, make sure you hardcode them in **`analysis.py`**

Along with your Python program, write a paragraph that describes your approach to this problem, and include links to any online resources that you used. You should not any cryptography specific module for this task.

Task 3: Dictionary Attacks on Passwords [35 Points]

A dictionary attack is a method to break authentication systems based on the assumption that users are likely to use common words (present in a dictionary) as passwords for accounts. One way to perform such an attack is actively trying to login with multiple password attempts until one guesses the correct password. This, however, is infeasible as only a few incorrect attempts will trigger a firewall or an intrusion detection system and block the malicious IP address from making further login attempts. Alternatively, if the attacker somehow gets access to hashes of the users' passwords, they can perform an offline dictionary attack by computing hashes for a large range of dictionary words and then compare them to the actual password hashes. In this case, if two hashes match, the attacker will know the password and can directly authenticate in a single attempt.

Your job in this task is to perform an offline dictionary attack to crack passwords of user accounts obtained from a Linux server at running in the W&L datacenter. Linux systems store the details of user accounts and their password hashes in the file: **`/etc/shadow`**. A simplified version of the **`shadow`** has been provided. While many online tools already exist, you are required to write your own password cracker. To assist, you've also have also provided **`dictionary.txt`** which you can use as a starting point.

The shadow file contains passwords of varying levels of difficulty, and you will be required to diversify your cracking approach. Here are a few details that will help you devise your program:

- Different passwords have been hashed using different hashing algorithms: md5, sha1, etc.
- One or more passwords may have has been hashed with a SALT.

You can assume that all passwords are alphanumeric ranging from 5-12 characters in length and can be found on **`dictionary.txt`** You may assume a SALT to be of only numeric value, in the range (0-9) and 5 digits in length. The SALT is appended at the end of a password before it is hashed e.g.: **`hash(passwordSALT)`**

For this task, write your program in a file named **`cracker.py`**. You are allowed to use standard python libraries, including the **`hashlib`** library, which you will require. In addition to your source code, submit a file **`passwords.txt`** that contains the plaintext passwords for each user on the server.

Submission Instruction
Include all files highighted **`red`** inside a folder named **`assignment_1`**. Create a zip archive of this folder and upload it on Canvas.