**Санкт-Петербургский государственный университет**

**Факультет прикладной математики и процессов управления**

**Лабораторная работа №1**

**Исследование хеш-функций**

**6 вариант**

|  |  |  |
| --- | --- | --- |
| Студентка гр. Б15-ПУ |  | Гладкая М.В. |
| Преподаватель |  | Дик А.Г. |

Санкт-Петербург

2023 г.

**Оглавление**

1. **Цель работы ………………………………………………………………3**
2. **Задача………………………………………………………………………3**
3. **Описание программы ……………………………………………………3**
   1. **Список используемых библиотек………………………………...…3**
   2. **Описание функций……………………………………………………3**
   3. **Общий ход программы………………………………………………4**
4. **Ход работы…………………………………………………………………5**
5. **Анализ полученных данных…...…………………………..……………5**
6. **Рекомендации программиста……………………………………………6**
7. **Вывод……………………………………………………………………….6**
8. **Цель работы**

Исследовать влияние размера соли на данных на время расшифровки различных хеш-функций.

1. **Задача**

Расшифровать исходный дата-сет, выяснить размер соли. Засолить дата-сет солями различной величины, захешировать несколькими различными функциями и проанализировать как размер соли влияет на время расшифровки дата-сета.

1. **Описание программы**

Программный код написан на языке Python.С кодом можно ознакомиться по ссылке <https://github.com/9Neechan/differentDataSaltHashes> .

* 1. **Список используемых библиотек**

Список используемых библиотек представлен в Таблице 1.

*Таблица 1. Список используемых библиотек*

|  |  |
| --- | --- |
| **Название** | **Описание** |
| hashcat | Используется для хеширования дата-сета функциями MD5, SHA1, SHA256 |

* 1. **Описание функций**

В таблице 2 представлено писание функций файла anonymity.py.

*Таблица 2. Функции main.py*

|  |  |  |
| --- | --- | --- |
| **Имя функции** | **Входные**  **данные** | **Описание функции** |
| addDict() | key – ключ  dict - словарь | Проверяет, есть ли ключ в словаре, если нет, то добавляет его со значением 1, если да – прибавляет 1 к значению по ключу |
| writeToFile() | file – файл  array – массив | Построчно записывает в файл значения из массива |
| hashMD5() | data - данные | Переводит данные в строку и хеширует их функцией MD5 |
| hashSHA1() | data - данные | Переводит данные в строку и хеширует их функцией SHA1 |
| hashSHA256() | data - данные | Переводит данные в строку и хеширует их функцией SHA256 |

* 1. **Общий ход программы**

На рисунке 1 представлена блок-схема файла main.py.

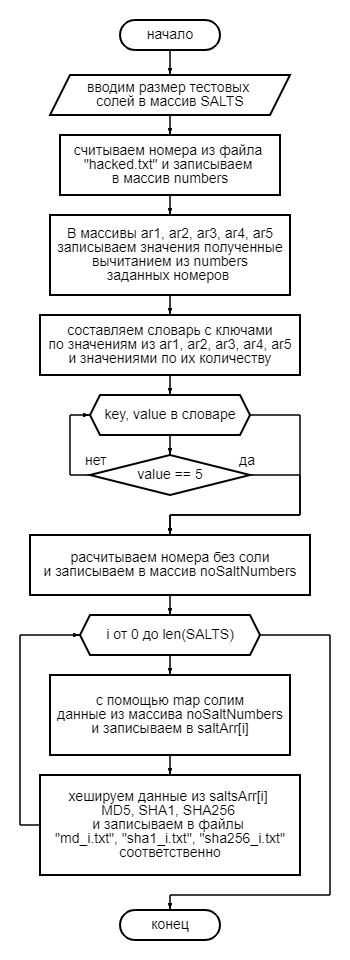
****

Рисунок 1. Блок-схема main.py

1. **Ход работы**

Были расшифрованы данные из файла “scoring\_data\_v.1.6.xlsx” с помощью hashcat. Строки из “scoring\_data\_v.1.6.xlsx” были записаны в “hahaha.txt ”. В командной строке в папке, где лежит hashcat была введена команда “hashcat -m 0 -a 3 hahaha.txt -o hacked.txt ?d?d?d?d?d?d?d?d?d?d?d”. В ее результате взломанные хеши были записаны в “hacked.txt”.

После чего была найдена соль, ее размер равен 3400319, вычтена из засоленных номеров. Полностью расшифрованные номера были записаны в файл “noSaltNumbers.txt”.

Далее к расшифрованным номерам были прибавлены тестовые соли различных размеров, засоленные номера были зашифрованы хеш-функциями MD5, SHA1, SHA256.

Потом все засоленные и захешированные дата-сеты были расшифрованы.

В ходе чего была составлена таблица 3, где отображено время расшифровки для каждой хеш-функции в зависимости от размера соли на данных.

*Таблица 3. Время расшифровки хеш-функций в зависимости от размера соли*

|  |  |  |  |
| --- | --- | --- | --- |
| **Размер соли** | **MD5, мин/сек** | **SHA1, мин/сек** | **SHA256, мин/сек** |
| 11111111 | 4.30 | 5.02 | 6.36 |
| 22222222 | 4.56 | 5.04 | 6.53 |
| 33333333 | 4.57 | 5.22 | 6.50 |
| 44444444 | 4.42 | 5.06 | 7.07 |
| 55555555 | 4.49 | 5.08 | 6.45 |
| 66666666 | 4.51 | 5.17 | 7.35 |
| 77777777 | 4.45 | 5.15 | 7.39 |
| 88888888 | 4.41 | 5.15 | 7.35 |
| 99999999 | 5.02 | 5.21 | 7.30 |

1. **Анализ полученных данных**

При анализе таблицы 3 замечено, что время расшифровки зависит от величины цифр, составляющих соль – в среднем, чем больше цифры, тем дольше идет расшифровка.

Время расшифровки с солью «11111111» заметно меньше, чем с «99999999» - в MD5 на 32 секунды, в SHA1 – на 19, в SHA256 – на 54.

1. **Рекомендации программиста**

Для запуска программы необходима 64-битная операционная система Windows и python не ниже 3.9. Для работы с кодом необходима IDE PyCharm или другая любая среда разработки для python.

Импортируйте библиотеки из пункта 3.1.

1. **Вывод**

В ходе проделанной работы было изучено влияние размера соли на хешированных данных на время их расшифровки.