МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра информационных систем и технологий

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №11 НА ТЕМУ:**

**Исследование криптографических хеш-функций**

Выполнил студент 3 курса 6 группы

       Подобед Владислав

Минск 2024

**Цель:** изучение алгоритмов хеширования и приобретение практических навыков их реализации и использования в криптографии.

**Задачи:**

1. Закрепить теоретические знания по алгебраическому описанию и алгоритмам реализации операций вычисления однонаправленных хеш-функций.

2. Освоить методику оценки криптостойкости хеш-преобразований на основе «парадокса дня рождения».

3. Разработать приложение для реализации заданного алгоритма хеширования (из семейств MD и SHA).

4. Оценить скорость вычисления кодов хеш-функций.

5. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

1. **Описание приложения**

Приложение написано на языке программирования Python и позволяет:

* хешировать входное сообщение;
* оценивать скорость выполнения алгоритма.

1. **Методика выполнения поставленных задач**

SHA256 – хеш-функция из семейства алгоритмов SHA-2, которая предназначена для создания дайджестов для сообщений произвольной длины. Длина дайджеста – 256 бит. Исходное сообщение дополняется до нужной длины, а затем разбивается на блоки. Каждый блок – на 16 слов. Каждый блок сообщения пропускается через 64 итерации. На каждой итерации 2 слова преобразуются, функцию преобразования задают остальные слова. Результаты обработки каждого блока складываются, сума – значение хеш-функции. Т.к. инициализация внутреннего состояния производится результатом обработки предыдущего блока, то нет возможности обрабатывать блоки параллельно.

Результат обработки строки с помощью алгоритма SHA256 представлен на рисунке 1.

послужат промежуточным значением хэш-кода:

a, b, c, d, e, f, g, h

Основой алгоритма является модуль, состоящий из 64 циклических обработок каждого блока M(i):

T_{1} = h + \Sigma _{1}^{(256)}(e) + Ch(e, f, g) + K_{t}^{(256)} + W_{t}
\\
T_{2} = \Sigma _{0}^{(256)}(a) + Maj(a, b, c)
\\
h = g
\\
g = f
\\
f = e
\\
e = d + T_{1}
\\
d = c
\\
c = b
\\
b = a
\\
a = T_{1} + T_{2}

где Ki{256} - шестьдесят четыре 32-битных константы, каждая из которых является первыми 32-мя битами дробной части кубических корней первых 64 простых чисел.

Wt вычисляются из очередного блока сообщения по следующим правилам:

W_{t} = M_{t}^{(i)} , 0 \le  t \le  15
\\
W_{t} = \sigma _{1}^{(256)}(W_{t-2}) + W_{t-7} + \sigma _{0}^{(256)}(W_{t-15}) + W_{t-16} , 
\\
16 \le  t \le  63

i-ое промежуточное значение хэш-кода H(t) вычисляется следующим образом:

H0(i) = a + H0(i-1)

H1(i) = b + H1(i-1)

H2(i) = c + H2(i-1)

H3(i) = d + H3(i-1)

H4(i) = e + H4(i-1)

H5(i) = f + H5(i-1)

H6(i) = g + H6(i-1)

H7(i) = h + H7(i-1)

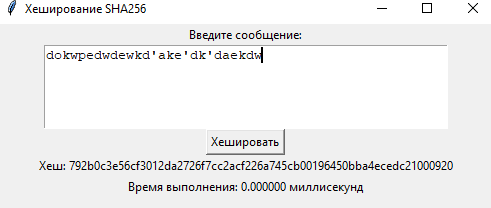


Рисунок 1 – Результат хеширования SHA256

Функция для хеширования строки с помощью алгоритма SHA256 представлена в листинге 1.

|  |
| --- |
| def \_\_init\_\_(self, master):          self.master = master          master.title("Хеширование SHA256")          master.geometry("500x200")          self.label = tk.Label(master, text="Введите сообщение:")          self.label.pack()          self.textbox = tk.Text(master, height=5, width=50)          self.textbox.pack()          self.button = tk.Button(master, text="Хешировать", command=self.hash\_message)          self.button.pack()          self.result = tk.Label(master, text="")          self.result.pack()          self.time\_label = tk.Label(master, text="")          self.time\_label.pack()      def hash\_message(self):          message = self.textbox.get("1.0", "end-1c")          start\_time = datetime.datetime.now()          hashed = sha256(message.encode()).hexdigest()          self.result.config(text="Хеш: " + hashed)          end\_time = datetime.datetime.now()          self.time\_label.config(              text="Время выполнения: {:.6f} миллисекунд".format((end\_time - start\_time).total\_seconds() \* 1000)) |

Листинг 1 – Функция для хеширования строки

В алгоритмах хеширования часто используется такое понятие как соль. Соль – строка данных, которая передается хеш-функции вместе с входными данными для вычисления хеша. Используется для усложнения определения прообраза хеш-функции методом перебора по словарю возможных значений, включая радужные атаки. Позволяет скрыть факт использования одинаковых прообразов при использовании разной соли.

Таким образом, были реализованы все поставленные задачи. Были исследованы асимметричные шифры.

**Вывод**

В ходе лабораторной работы были приобретены навыки хеширования используя алгоритм SHA256. Были изучены основные принципы работы хеширования.

Также было разработано приложение, на языке программирования Python, для реализации задач, связанных с хешированием данных.