Лабораторная работа № 3 по курсу: криптография

Выполнил студент группы М8О-308Б-17 МАИ *Милько Павел.*

Задача

1. Строку в которой записано своё ФИО подать на вход в хеш-функцию ГОСТ Р 34.11-2012 (Стрибог). Младшие 4 бита выхода интерпретировать как число, которое в дальнейшем будет номером варианта. Процесс выбора варианта требуется отразить в отчёте.

2. Программно реализовать один из алгоритмов функции хеширования в соответствии с номером варианта. Алгоритм содержит в себе несколько раундов.

3. Модифицировать оригинальный алгоритм таким образом, чтобы количество раундов было настраиваемым параметром программы. в этом случае новый алгоритм не будет являться стандартом, но будет интересен для исследования.

4. Применить подходы дифференциального криптоанализа к полученным алгоритмам с разным числом раундов.

5. Построить график зависимости количества раундов и возможности различения отдельных бит при количестве раундов 1,2,3,4,5,... .

6. Сделать выводы.

Определение номера варианта

```
❯ ipython
Python 3.8.2 (default, Apr  8 2020, 14:31:25)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.13.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: from pygost import gost34112012256

In [2]: gost34112012256.new("Милько Павел Алексеевич".encode('utf-8')).digest().hex()[-1]
Out[2]: '7'
```

Вариант 7 - keccak (sha3)

Реализация

В качестве реализации я использовал как основу данный репозиторий. Для внесения изменений скопировал файл keccak.go в локальный проект.

Чтобы удостовериться что в коде действительно алгоритм SHA-3 я решил сверить-ся со стандартом. Там даже есть ссылка на примеры (25 страница), но увы пользоваться ими очень неудобно потому что приведены вычисления для сообщений размером 0, 5, 30, 1600, 1605 и 1630 бит.

Из этого чудного многообразия есть возможность протестировать корректность только для примеров длиной 0 и 1600 бит, так как эти числа кратны восьми и можно передавать их байтами.

Тест с пустым сообщением выдал такой же хеш как и в официальном примере. Но для примера в 1600 бит одинакового результата не получилось.

По самим примерам нереально даже понять правильно ли мною скопированы входные данные, поэтому для проверки корректности результата я воспользовался официальной библиотекой sha3 для языка го и дополнительно проверил корректность аналогичной библиотекой для питона.