Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Исследование криптографических хеш-функций

Студент: Чистякова Ю.А.

ФИТ 3 курс 5 группа

Преподаватель: Берников В.О.

Минск 2020

## 1. Описание приложения

Приложение написано на языке программирования C# и реализиует алгоритм хеширования SHA256. Приложение обрабатывает входное сообщение, длина которого определена спецификацией

## 2. Методика выполнения поставленных задач

## Данный алгоритм имеет 64 итерации в одном цикле, принимает на вход сообщения предельно допустимого размера до 264 бит, и выдает на выходе дайджест размером 32 байт. SHA256 представляет собой однонаправленную функцию.

Программная реализация алгоритма в разработанном ПС осуществляется с помощью встроенной C# библиотеки класса System.Security.Cryptography. Рассмотрим подробнее алгоритм хеширования.

Во-первых, необходимо сгенерировать так называемую «соль» указанной длины – строка данных, которая позже будет передана хэш-функции вместе с входным массивом данных (прообразом) для вычисления хэша. Программная реализация генерации соли продемонстрирована на рисунке 2.1.

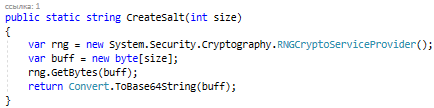


Рис. 2.1 – Генерация соли

RNG (random number generator) возвращает случайную последовательность чисел, которую мы конвертируем в строку base64. Соль используется нами для усложнения определения прообраза хэш-функции методом перебора по словарю, таким образом максимально увеличивает криптостойкость и уменьшает возможность коллизии. В данном случае использована динамическая соль, что является еще более эффективным с точки зрения профилактики атак.

Далее вызывается функция генерации хэша с помощью алгоритма SHA256, встроенная в библиотеку. Входными параметрами данной функции являются вычисленная на предыдущем шаге «соль» и входное сообщение. Программная реализация данной функции продемонстрирована на рисунке 2.2.

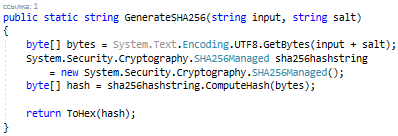


Рис. 2.2 – Реализация функции хэширования

В данном случае использована встроенная функция SHA256Managed(), позволяющая вычислить хэш. После получения результата функции, преобразуем его в 16-ричное представление с помощью разработанной нами функции ToHex().

Рассмотрим и проанализируем результаты выполнения выше указанных функций. Результат запуска консольного приложения представлен на рисунке 2.3.

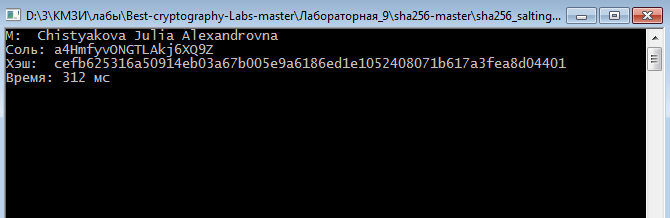


Рис. 2.3 – Результат работы приложения

Проанализировав время выполнения операции хеширования, можно предположить, что алгоритм SHA256 достаточно производительный. Это подтверждает то, что в протоколе указана максимально достиаемая скорость примерно 140 МБ/с. Ведь, если процесс вычисления хеша не достаточно быстрый, система просто не будет эффективной.

## Вывод

В ходе лабораторной работы были изучены алгоритмы хеширования и приобретены практические навыки их реализации и использования в криптографии. Также была оценена скорость вычисления кодов хеш-функции SHA256.