Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра информационных систем и технологий

**Отчет к лабораторной работе №11**:

Исследование криптографических хеш-функций

Выполнила:

студентка 3 курса 6 группы

Трошко В.Н.

Преподаватель:

Нистюк О.А.

**2022 г.**

1. **Теоретические сведения**

Определение 1. Хеш-функция – математическая или иная функция, h = H(М), которая принимает на входе строку символов М, называемую также прообразом, переменной длины n и преобразует ее в выходную строку фиксированной (обычно – меньшей) длины, l.

Определение 2. Хеширование (или хэширование, англ. hashing ) – это преобразование входного массива данных определенного типа и произвольной длины (практически) в выходную битовую строку фиксированной длины.

Преобразования называются хеш-функциями или функциями свертки, а их результаты называют хешем, хеш-кодом, хеш-таблицей или дайджестом сообщения (анг. message digest).

Все существующие функции хеширования можно разделить на два больших класса:

* бесключевые хеш-функции, зависящие только от сообщения,
* хеш-функции с секретным ключом, зависящие как от сообщения, так и от секретного ключа.

Определение 3. Криптографическая хеш-функция – это специальный класс хеш-функций, который имеет различные свойства, необходимые для решения задач в области криптографии.

Основные задачи, решаемые с помощью хеш-функций:

* аутентификация (хранение паролей),
* проверка целостности данных,
* защита файлов,
* обнаружение зловредного ПО,
* криптовалютные технологии.

К основным свойствам хеш-функций можно отнести следющие.

Свойство 1. Детерминированность: независимо от того, сколько раз вычисляется H(M), M – const, при использовании одинакового алгоритма код хеш-преобразования h всегда должен быть одинаковым.

Свойство 2. Скорость вычисления хеша h: если процесс вычисления h не достаточно быстрый, система просто не будет эффективной.

Свойство 3: Сложность обратного вычисления: для известного H (М) невозможно (практически) определить М. Это важнейшее свойство хеш-функции для криптографических применений – односторонности преобразования.

Это означает, что по хеш-коду должно быть практически невозможным восстановление входной строки М.

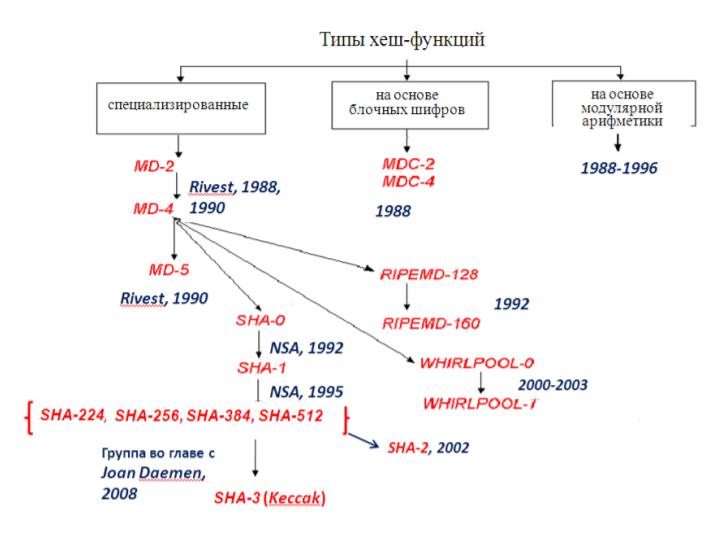


Рисунок 1.1 – Основные классы хеш-функций в соответствии с используемым внутренним преобразованием

Большинству даже простых пользователей известно, что наиболее распространенная область применения хеширования – хранение паролей. К примеру, если пользователь забыл пароль и пытается воспользоваться доступными функциями-сервисами восстановления пароля, то в этом случае, такой сервис может выдать пароль, как правило, в двух случаях:

* информационная система, в которую входит пользователь с помощью пароля, на самом деле, не хеширует этот пароль,
* используемая система восстановления пароля использует некоторую базу данных, содержащую простые, наиболее часто используемые пароли (например, 123456 или qwerty); примером такой системы восстановления пароля является Online Reverse Hash Lookup.

В плане односторонности хешей на основе блочных шифров отметим одно обстоятельство. Блочный шифр необратим по ключу шифрования, и, если в качестве такого ключа на текущем шаге преобразования используется выход предыдущего шага, а в качестве шифруемого сообщения – очередной блок сообщения (или наоборот), то можно получить хеш-функцию с хорошими криптографическими характеристиками с точки зрения односторонности.

Такой подход использовался, например, в российском стандарте хеширования – ГОСТ Р 34.11-94.

Основным недостатком хеш-функций на основе блочных шифров является сравнительно невысокая производительность.

Свойство 4. Даже минимальные изменения в хешируемых данных (М ≠ М') должны изменять хеш: Н(M) ≠ Н(М').

Определение 4. Коллизией хеш-функции Н называют ситуацию, при которой различным входам (в общем случае – х и у или М ≠ М') соответствует одинаковый хеш-код: H(x) = H(y) или H(М) = H(М').

Свойство 5. Коллизионная устойчивость (стойкость).

Зная М, трудно найти такое М' (М ≠ М'), для которого H(М) = H(М').

Если последнее равенство выполняется, то говорят о коллизии 1- го рода.

Если случайным образом выбраны два сообщения (М и М’), для которых H(М) = H(М'), говорят о коллизии 2-го рода.

Мерой криптостойкости хеш-функции считается вычислительная сложность нахождения коллизии.

Для хеш-функций одним из основных средств поиска коллизий является метод, основанный на известной статистической задаче – «парадоксе дня рождения».

В более общем случае: для того, чтобы хеш-функция H(M) считалась криптографически стойкой, она должна удовлетворять трем основным требованиям: необратимостью вычислений (свойство 3), устойчивостью к коллизиям первого рода и устойчивостью к коллизиям второго рода (свойство 5).

Основной постулат парадокса «дней рождения» гласит: в группе минимум из 23 человек с вероятностью более 0,5 день рождения одинаков. Парадоксом является высокая (как кажется на первый взгляд) вероятность наступления указанного события. При этом предполагается, что:

* в этой группе нет близнецов;
* люди рождаются независимо друг от друга, т. е. дата (день) рождения любого человека не влияет на дату рождения другого;
* люди рождаются равномерно и случайно, т. е. люди с равной вероятностью могут рождаться в любой день года; с формальной точки зрения это означает, что вероятность р1 рождения отдельно выбранного члена группы (как и любого человека) в любой выбранный день равна р1 =1/365 (хотя известно, что в реальности рождение людей не совсем соответствует такому предположению).

Определение 5. Хеш-функция – это функция, выполняющая отображение из множества М в число, находящееся в интервале в интервале [0, m–1]: h: M → [0, m–1].

Мы ранее отмечали, что стойкость хеш-преобразования к коллизии означает, что трудно найти такие Мi и Мj (Мi, Мj М), при которых h(Мi) = h(Мj), i ≠ j, 1 ≤ i, j ≤ n

Для выполнения анализа атаки на основе парадокса «дней рождения» будем использовать те же принципы, которые мы применяли для вероятностной оценки дней рождения.

В атаке «дней рождения» m соответствует количеству календарных дней в году, а М – множеству людей, составляющих группу. Люди «хешируются» в их дни рождения, которые могут быть одним из значений m.

Допустим (переходя в информационную область), нам нужно найти коллизию с вероятностью 0,99 (Рс(Аn) = 0.99). Мы хотим определить наименьшее n, при котором хеш двух значений из Аn будет «одним днем рождения», что в интересующей нас плоскости означает, что два входных набора данных (Мi, Мj М) хешируются в одинаковое значение: h(Мi) = h(Мj). Допустим далее, что все входные данные хешируются в m выходных хеш-кодов.

При атаке «дней рождения» злоумышленник будет случайным образом подбирать Мi и Мj и сохранять пары их хешей, пока не найдет двух значений, при которых h(Мi) = h(Мj). Нам нужно определить, сколько раз атакующему нужно повторить эту операцию, пока не будет обнаружена коллизия.

Иначе говоря, стоит задача отыскания наименьшего n, при котором хеши двух значений m будут «одним днём рождения».

n = (2m\* ln 100)1/2.

Если хеш имеет длину l бит, то m = 2 l . И в соответствии с (9.7) для поиска коллизии с вероятностью 0.99 нужно выполнить 2l/2 операций хеширования различных входных сообщений.

На рис. 1.2 приведены вероятностные оценки появления коллизии для хеш-функций различной длины (в приведенной таблице параметр N соответствует принятому нами обозначению l).

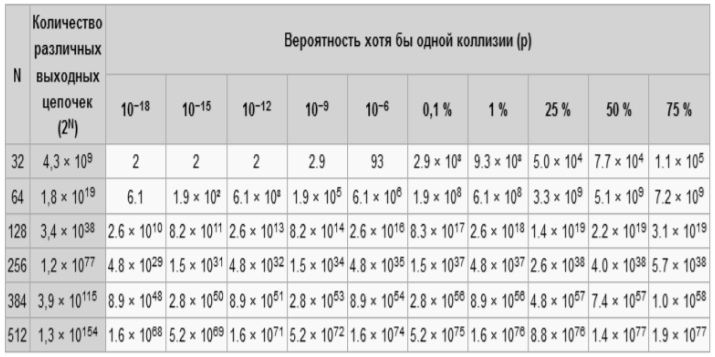


Рисунок 1.2 – Вероятностные оценки появления коллизии для хеш-кодов различной длины l

Алгоритмы семейства MD-x (2/4/5/6) являются творениями Р. Ривеста; MD – Message Digest. Алгоритм MD6, в отличие от предыдущих версий алгоритма этого семейства, не стандартизован.

Алгоритмы семейства SHA (SHA – Secure Hash Algorithm) являются в настоящее время широко распространенными. Во многих случаях завершился переход от SHA-1 к стандартам версии SHA-2. SHA-2 – собирательное название алгоритмов SHA-224, SHA-256, SHA-384 и SHA-512. SHA-224 и SHA384 являются, по сути, аналогами SHA-256 и SHA-512 соответственно.

Известен также алгоритм хеширования, долгое время использовавшийся в качестве национального стандарта (ГОСТ 34.11- 94) России.

Алгоритмы семейства MD входные сообщения максимальной длины 264-1 бит (в общем случае – L бит) преобразуют в хеш длиной l = 128 бит. Исключением является последняя – 6 – из версий алгоритма, где длина результирующего хеша может изменяться от 1 до 512 бит.

Максимальный объем хешируемых сообщений для алгоритмов SHA-1, SHA-256, SHA-224 такой же, как и для алгоритмов MD. Однако длина хешей разная: в SHA-1 – 160 бит; во алгоритмах, относящихся к семейству SHA-2 – соответствует числу, дополняющему через дефис название алгоритма. Максимальная же длина входных сообщений в алгоритмах SHA-512, SHA-384, SHA-512/256, SHA-512/224 составляет 2 128-1 бит.

Базовые алгоритмы обоих рассматриваемых семейств (MD и SHA) условно можно разделить на 5 стадий:

* расширение входного сообщения;
* разбивка расширенного сообщения на блоки;
* инициализация начальных констант;
* обработка сообщения поблочно (основная процедура алгоритма хеширования);
* вывод результата.

Входное сообщение «дополняется» (расширяется) так, чтобы его длина (в битах) была конгруэнтной к 448 по модулю 512. Это значит, что сообщение начальной длиной L бит расширяется так, что остаются незаполненными всего лишь 64 бита, чтобы итоговая длина L' была кратной 512. В указанные 64 бита записывается двоичная длина.

Расширение происходит всегда, даже если длина сообщения уже соответствует 448, по модулю 512. Эта операция выполняется следующим образом: один бит «1» добавляется к сообщению, а затем добавляются биты «0», так что длина в битах дополненного сообщения стала конгруэнтной 448 по модулю 512. Добавляется не менее одного бита, но не более 448 бит.

Как было отмечено выше, основная операция заключается в циклической (пораундовой или поэтапной) обработке 512-битных блоков. Таких циклов может быть 3 (как в MD-4), или 4 (как в MD-4), или более. В каждом цикле используется своя нелинейная функция (обычно обозначаемая по порядку F, G, H,…), зависящая от текущего состояния 4 (в MD), 5 (в SHA-1), 8 (SHA-256) и т. д. переменных, начальные состояния которых известны, а текущие – зависят от выполненных операций над хешируемым сообщением.

В алгоритмах MD-5 и SHA-1результат текущего действия прибавляется к результату предыдущего. Это направлено на усиление лавинного эффекта. Этой же цели служит то обстоятельство, что значения циклического сдвига влево на каждом этапе были приближенно оптимизированы: четыре сдвига, используемые на каждом этапе, отличаются от значений, используемых на других этапах.

1. **Практическая часть**

В данной лабораторной работе необходимо разработать оконное приложение, реализующее алгоритм хеширования MD5. При этом можно воспользоваться доступными готовыми библиотеками.

Так, для реализации этого алгоритма была использована библиотека System.Security.Cryptography, реализующая алгоритм хеширования MD5. Для этого была написана следующая функция, представленная на рисунке 2.1.

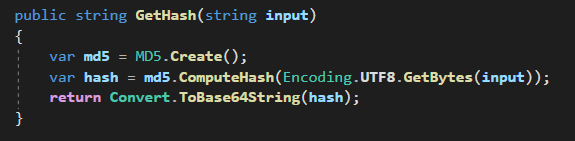


Рисунок 2.1 – Реализация функции хеширования

Функция принимает в себя один параметр – исходный текст, хэш которого необходимо получить. Затем создается экземпляр встроенного класса MD5, который содержит в себе метод ComputeHash, позволяющий получить хэш сообщения. В метод необходимо передать сообщение в двоичном виде.

Также программа позволяет узнавать время выполнения работы алгоритма Работа программы продемонстрирована на рисунке 2.2.

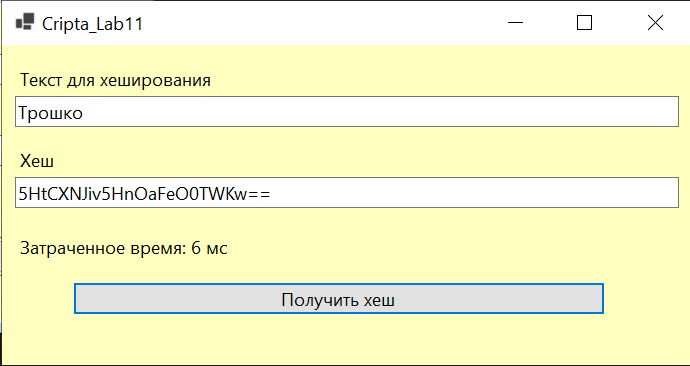


Рисунок 2.2 – Работа программы с входной строкой «Трошко»

Основное правило хеширования гласит, что разные строки должны образовывать разный хеш, результат работы приложения, подтверждающий это правило представлен на рисунке 2.3:

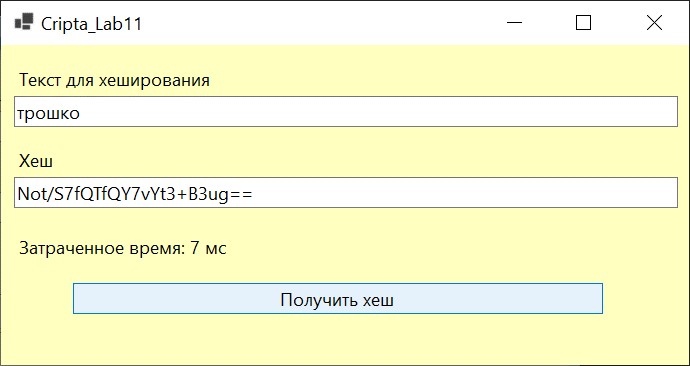


Рисунок 2.3 – Работа программы с входной строкой «трошко»

Тем самым мы опытным путём подтверждаем это правило т.к. строки «Трошко» и «трошко» между собой схожи, но по факту разные.

**Вывод:** таким образом, в данной лабораторной работе я закрепила теоретические знания по алгебраическому описанию, алгоритмам реализации операций вычисления однонаправленных хэш-функций, разработала приложение для реализации алгоритма хеширования MD5, а также оценила скорость вычисления кодов хеш-функций.