Задача

Вариант 4.

Реализовать алгоритм хеширования SHA-1.

Провести эксперимент, согласно варианту: придумать 3 слова-пароля, которые будут исходным сообщением x (пусть один из них будет состоять из нулей), далее с каждым: найти от него хеш h(x) взять от полученного хеша первые 8 бит, обозначив последовательность y0. Далее случайным образом сгенерировать N (N зависит от двух условий, описанных ниже) сообщений, найти от каждого хеш и взять от каждого хеша первые 8 бит, получив последовательность y1, y2,..., yN-1.

1). Нахождение второго прообраза. Необходимо найти такой yi, что yi = y0. Посчитать количество шагов, которое потребовалось, чтобы найти yi - это сложность второго прообраза.

2). Нахождение коллизий. Необходимо найти в полученной последовательности такие yi и yj, что yi = yj. Посчитать количество шагов, которое потребовалось, чтобы найти эту пару - это сложность коллизии.

Проделать оба эксперимента 1000 раз, получить средние значения сложности второго прообраза и сложности коллизии.

Повторить данные манипуляции для последовательности хешей размером 10, 12, 14 и 16 бит. Построить графики зависимости среднего значения сложности второго прообраза и коллизии от количества взятых бит. Оценить полученные графики.

Тестируемый алгоритм

SHA-1 представляет собой алгоритм хэширования 512-битных блоков данных в 160-битный хэш.

Описание алгоритма

SHA-1 реализует хеш-функцию, построенную на идее функции сжатия. Входами функции сжатия являются блок сообщения длиной 512 бит и выход предыдущего блока сообщения. Выход представляет собой значение всех хеш-блоков до этого момента. Хеш-значением всего сообщения является выход последнего блока.

В начале работы алгоритма сообщение дополняется так, чтобы его длина стала кратной 512 разрядам: в конец сообщения добавляют 1, а затем столько нулей, чтобы размер сообщения стал

кратен 448, а затем к полученному результату добавляется 64-битовое представление размера исходного сообщения. Затем особым образом инициализируются пять 32-битовых переменных и начинается главный цикл(рис.1).

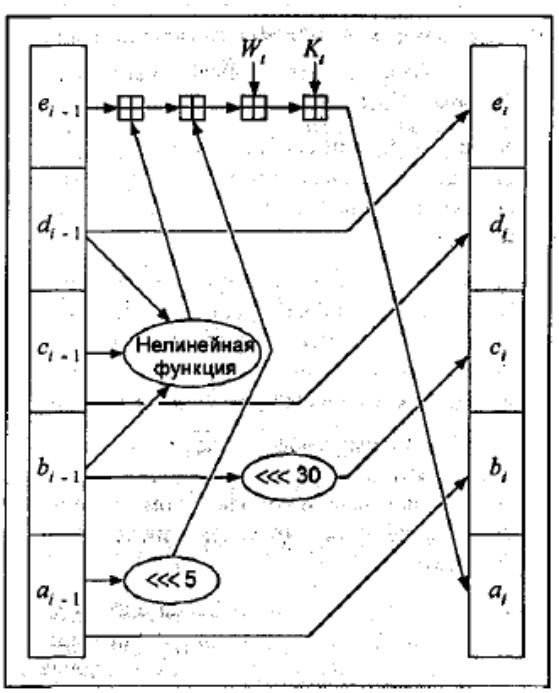

Рисунок 1. Схема одной операции SHA-1

Главный цикл состоит из 4 раундов, каждый из которых включает по 20 операций. Каждая такая операция состоит из подсчета нелинейной функции над тремя переменными из пяти. После выполнения вычислений над результатом выполняются операции сдвига и сложения.

После выполнения всех вышеперечисленных операций значения, полученные путем обработки текущего блока данных, складываются со значениями, полученными путем обработки предыдущих блоков данных, и алгоритм переходит к обработке следующего блока данных. Окончательный результат получается конкатенацией значений.

Примеры использования алгоритма

Пример 1.

Исходное сообщение: «»

Хэш: «da39a3ee 5e6b4b0d 3255bfef 95601890 afd80709»

Пример 2.

Исходное сообщение: «sha»

Хэш: «d8f45903 20e1343a 915b6394 170650a8 f35d6926»

Пример 3.

Исходное сообщение: «Sha»

Хэш: «ba79baeb 9f10896a 46ae7471 5271b7f5 86e74640»

Исследование

Для проведения исследования были взяты следующие слова-пароли: «word», «sha», «00000». Результаты исследования приведены в таблице 1:

Таблица 1. Результаты исследования

Количество взятых бит	Тест	Слово-пароль		
		word	sha	00000
6	Прообраз	39.772	10.000	48.308
	Коллизия	22.572	1.000	21.787
8	Прообраз	171.296	147.191	75.814
	Коллизия	19.617	34.532	12.224
10	Прообраз	427.387	3.000	945.801
	Коллизия	36.121	3.000	60.351
12	Прообраз	835.984	3301.323	18165.724
	Коллизия	42.878	62.935	65.092
14	Прообраз	86040.753	44774.337	52536.06
	Коллизия	53.056	42.727	24.728

По результатам исследования были построены графики зависимостей второго прообраза и коллизии от количества взятых бит (рис.2 и 3):

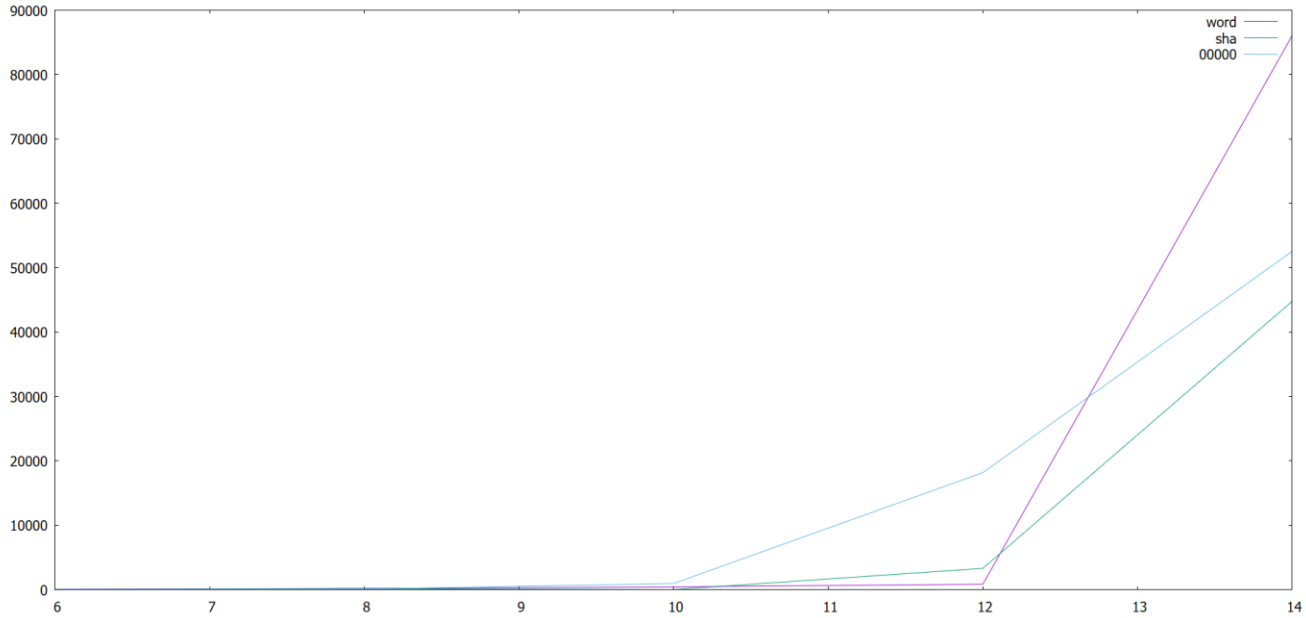

Рисунок 2. График зависимости второго прообраза от количества взятых бит

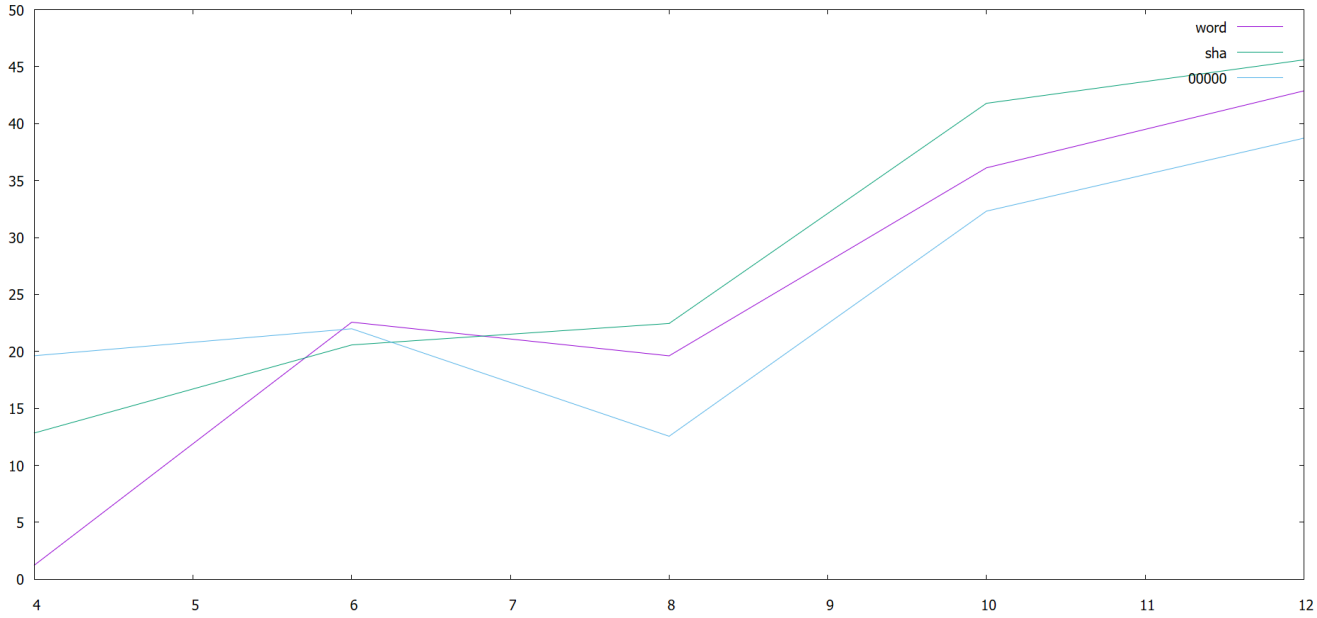

Рисунок 3. График зависимости коллизии от количества взятых бит

Вывод

В данной лабораторной работе был реализован алгоритм хэширования SHA-1, по которому было проведено исследование по нахождению второго прообраза и коллизии, на основании которого можно сделать вывод, что чем длиннее искомая последовательность, тем сложнее ее найти.

SHA-1 является наиболее распространенным из всего семейства SHA и применяется в различных широко распространенных криптографических приложениях и алгоритмах, таких как S/MIME, SSL, IPSec и т.д.

Из-за блочной и итеративной структуры алгоритмов, а также отсутствия специальной обработки в конце хеширования, все хеш-функции семейства SHA уязвимы для атак удлинением сообщения и коллизиям при частичном хешировании сообщения. Эти атаки позволяют подделывать сообщения, подписанные только хэшем путём удлинения сообщения и пересчёту хэша без знания значения ключа. Простейшим исправлением, позволяющим защититься от этих атак, является двойное хеширование.

Данный алгоритм был создан в 1995 году, а уже в 2005 году были опубликованы теоретические атаки на SHA-1, требующие менее 2^{63} операций. В 2017 году специалисты из Google и CWI объявили о практическом взломе алгоритма, опубликовав 2 PDF-файла с одинаковой контрольной суммой SHA-1.