Исследование и реализация хеш-функции SHA-2

Работу выполнил Попов Артём Юрьевич, 10 класс
под руководством Троицкого Игоря Ивановича

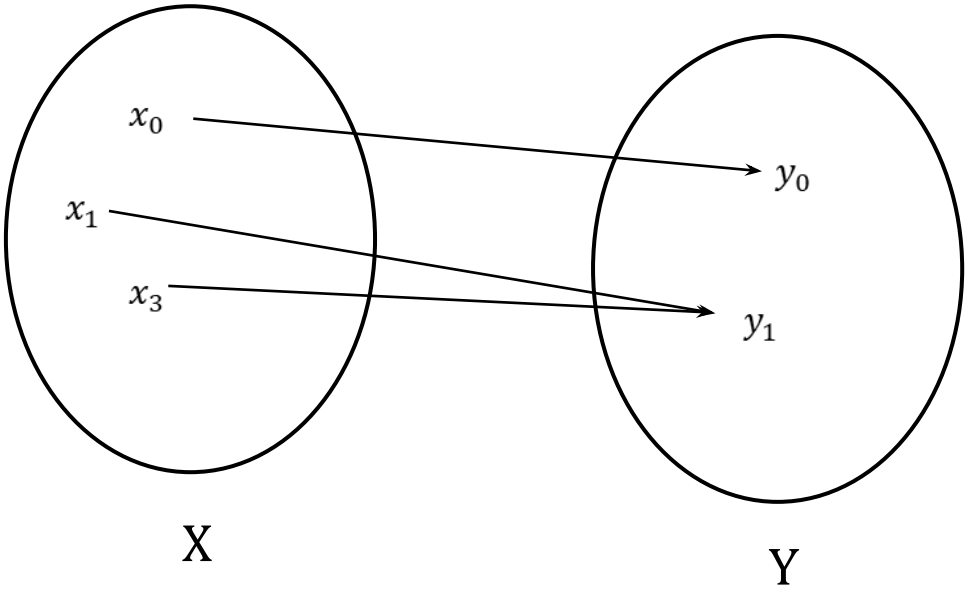
Хеш-функции. Общие сведения

$$H: X \to Y; |X| > |Y|$$

где X *– множество всевозможных сообщений (ключей),*

Y *– множество возможных значений хеш-функции.*

Коллизии



Коллизии

При равномерном распределении, количество прообразов к одному значению выходных данных можно рассчитать по формуле:

$$d = \frac{2^{\text{максимальная длина сообщения (бит)}}}{2^{\text{размер свёртки (бит)}}}$$

Типы хеш-функций

1. Хеш-функции для ускорения поиска информации. Применяется для построения хеш-таблиц – особых структур, в которых скорость поиска достигается с помощью использования свёртки, как ключа в таблице.

2. Протокол для аудита целостности информации. Используется для проверки целостности полученных данных.

3. Криптографические хеш-функции. Используются для защиты информации, о них пойдёт речь далее.

Криптографические хеш-функции

Требования к криптографическим хеш-функциям:

1. Стойкость к поиску первого прообраза (необратимость) –
Сложность восстановления исходного текста.

2. Стойкость к поиску второго прообраза (стойкость к коллизиям первого рода) —
Сложность найти, зная сообщение n, другое сообщение m, такое, что $H(n) = H(m)$.

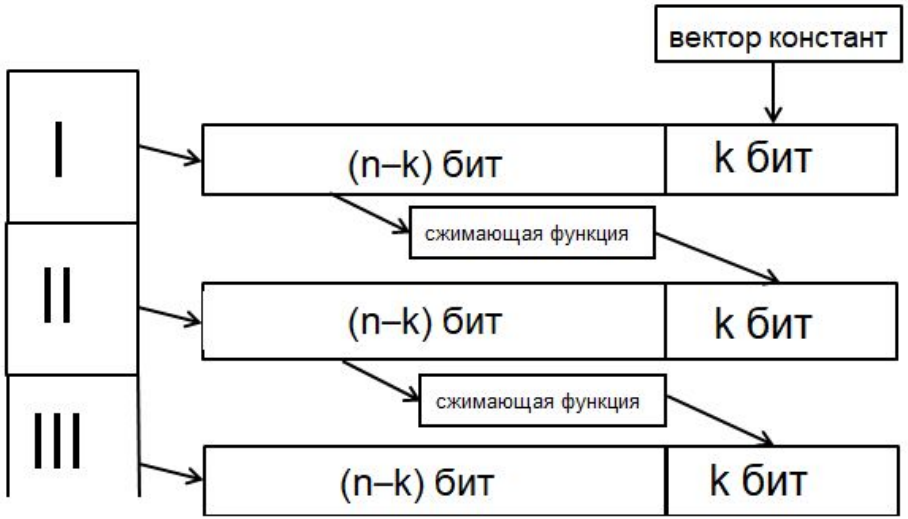
3. Стойкость к коллизиям (стойкость к коллизиям второго рода) –
Сложность подобрать такую пару n, m, что $H(n) = H(m)$.

Структура Меркла-Дамгарда

$$h = v$$
$$h_i = f(m_i, h_{i-1})\ i = 1, \dots, n$$
$$h(m) = h_n$$

где v – начальный вектор констант, m – сообщение, h – хеш-функция

Структура Меркла-Дамгарда



Сообщение поделено на блоки по *n-k* бит каждый. *k* – длина свёртки.

Применяется в таких хеш-функциях, как MD5, SHA-1, SHA-2, о которых речь пойдёт позже.

Атака методом «грубой силы»

По вышеуказанной формуле $d = \frac{2^{\text{максимальная длина сообщения (бит)}}}{2^{\text{размер свёртки (бит)}}}$ можем вычислить количество сгенерированных сообщений на примере хеш-функции SHA-256 значения будут повторяться каждое $2^{2^{64}-257}$ сообщение:

$$2^{2^{64}-257} = 10^{10^{18.74452949484047}}$$

Атака «дней рождения»

Данная атака позволяет уменьшить количество перебираемых значений хеш-функции с 2^n до $2^{\frac{n}{2}}$. Алгоритм основан на парадоксе о том, что чтоб с вероятностью 50% у двух учеников совпал день рождения, нужно взять выборку из 23 ребёнка.

$$P(n) = \prod_{i=1}^{i-1} \frac{n-i}{n} \approx e^{\frac{-m^2}{2n}}$$

тождество, использованное для аппроксимации: $1 - x \approx e^{-x}$

значит $m \approx \sqrt{2nlnln\left(\frac{1}{1-p}\right)}$, *где m –необходимое количество вычисленных значений, n – длина выходного массива, p – вероятность коллизии.*

Атака «дней рождения»

Биты	Желаемая вероятность случайной коллизии (P)									
	10^{-18}	10^{-15}	10^{-12}	10^{-9}	10^{-6}	0,1 %	1 %	25 %	50 %	75 %
256	$4{,}8\times10^{29}$	$1{,}5 \times 10^{31}$	$4{,}8 \times 10^{32}$	$1{,}5 \times 10^{34}$	$4{,}8 \times 10^{35}$	$1{,}5 \times 10^{37}$	$4{,}8 \times 10^{37}$	$2{,}6 \times 10^{38}$	$4{,}0 \times 10^{38}$	$5{,}7 \times 10^{38}$
384	$8{,}9 \times 10^{48}$	$2{,}8 \times 10^{50}$	$8{,}9 \times 10^{51}$	$2{,}8 \times 10^{53}$	$8{,}9 \times 10^{54}$	$2{,}8 \times 10^{56}$	$8{,}9 \times 10^{56}$	$4{,}8 \times 10^{57}$	$7{,}4 \times 10^{57}$	$1{,}0 \times 10^{58}$
512	$1{,}6 \times 10^{68}$	$5{,}2 \times 10^{69}$	$1{,}6 \times 10^{71}$	$5{,}2 \times 10^{72}$	$1{,}6 \times 10^{74}$	$5{,}2 \times 10^{75}$	$1{,}6 \times 10^{76}$	$8{,}8 \times 10^{76}$	$1{,}4 \times 10^{77}$	$1{,}9 \times 10^{77}$

Атака методом удлинения сообщения

Суть атаки состоит в дописывании сообщения *m*, хешированного вместе с ключом с помощью конструкции, создании новой действительной подписи, не зная секретного ключа *k:*

$$h(m \mid k)$$

Вспомним, что на этапе обработки сообщение дополняется паддингом, зависящем от длины сообщения.

сообщение | паддинг | новые_данные | новый паддинг

Атака методом удлинения сообщения

Запрос: `from=alice&to=bob&amount=30000`

Подпись запроса *h(ключ | запрос)*

Учитывая паддинг: *ключ | запрос | паддинг*

Паддинг (рассчитано помощью функции pad из программы [2] `sha256.py` в приложении):
:

`\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01`

Припишем к сообщению рассчитанный паддинг и «`00`», чтобы увеличить размер перечисляемой платы в 100 раз, отправим запрос.

Далее, алгоритм находящейся в правильном состоянии хеш-функции, обработает оставшуюся часть нового сообщения и создаст новую действительную подпись (вычислено с помощью функции `sha256` из программы [2] `sha256.py` в приложении):

`c154aa5feff7ac1db7b5dcd6dfe1038a884277bdc7d71149c748f87a449d20ae`

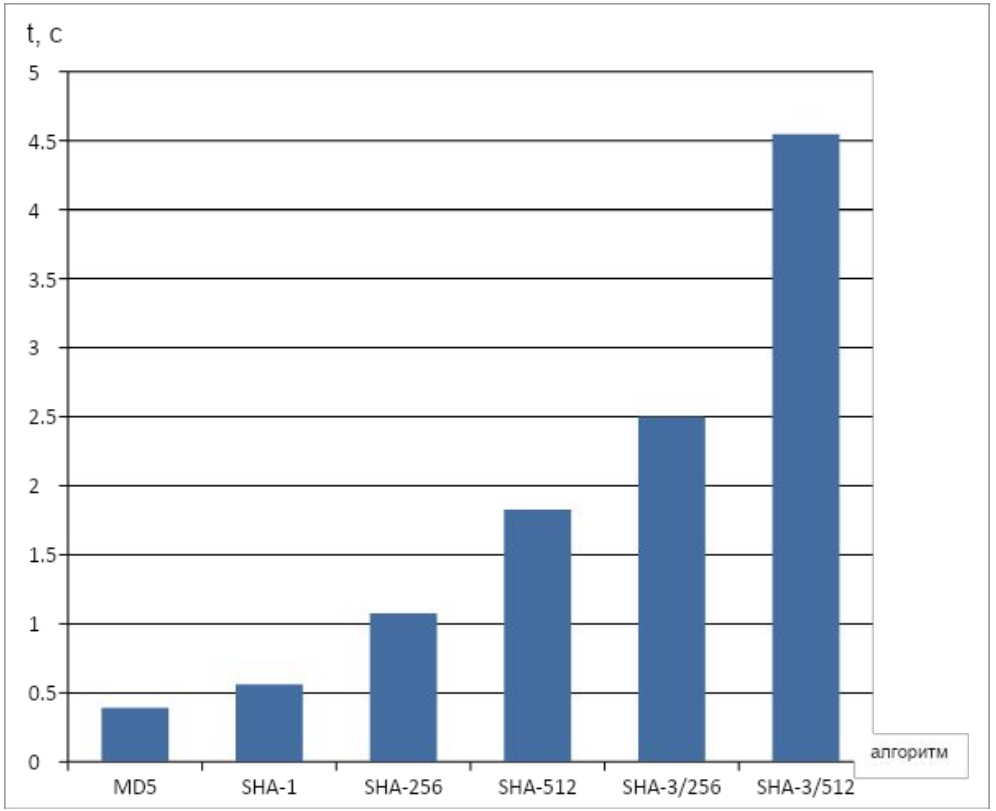
Общее сопоставление характеристик

Алгоритм	Год публикации	Размер свёртки* (бит)	Кол–во операций необх. для поиска коллизий	Уязвимость к атаке удлинения
MD5	1992	128		есть
SHA-1	1993	160		есть
SHA-2	2001 – 2012 (разные вариации)	224–512		есть (кроме SHA-512/224 SHA-512/256)
SHA-3 (Keccak)	2015	произвольный		нет

Сравнение скорости вычисления

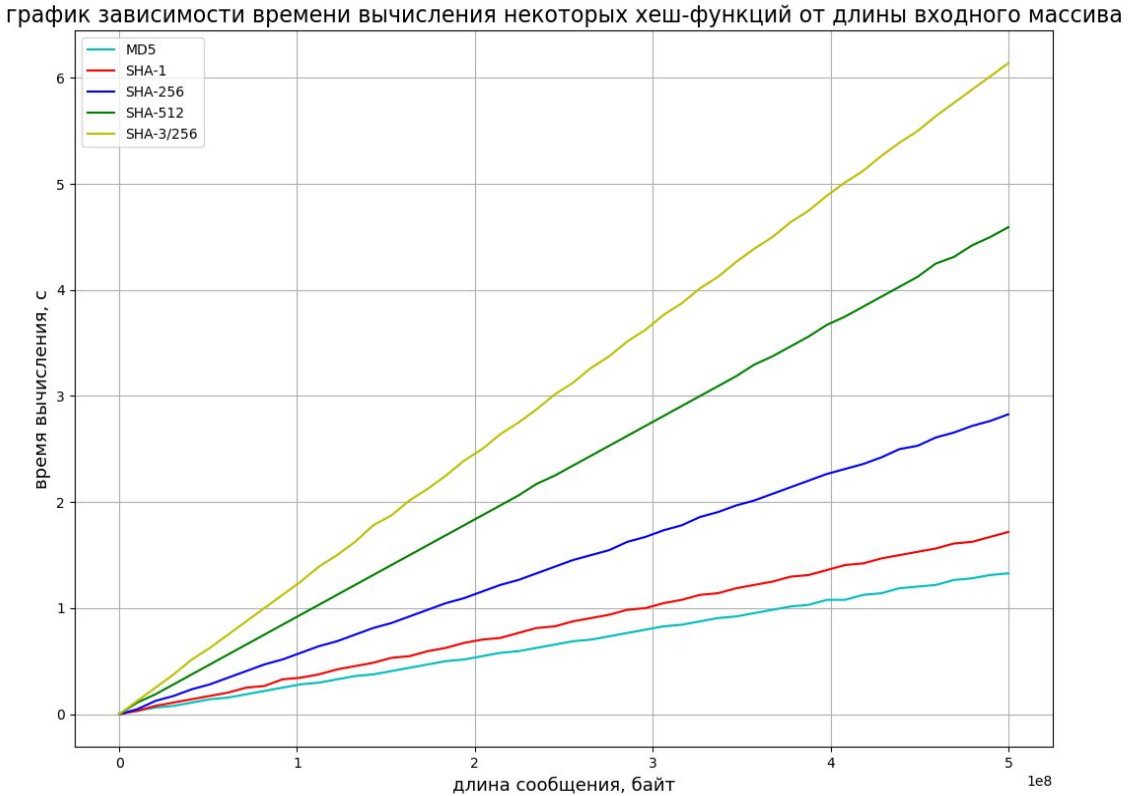
Время вычисления хеш-функции от сообщения «`The quick brown fox jumps over a lazy dog`», записанного 5120000 раз подряд.

Была выбрана фраза на латинице, т.к. кириллица в кодировке UTF-8 обозначается с помощью 2-х байт, а не одного, как ASCII символы.



Тесты скорости были проведены с помощью функции `calc_hash` из программы `[1]`, `calc_time.py`

Зависимость скорости вычисления от длины



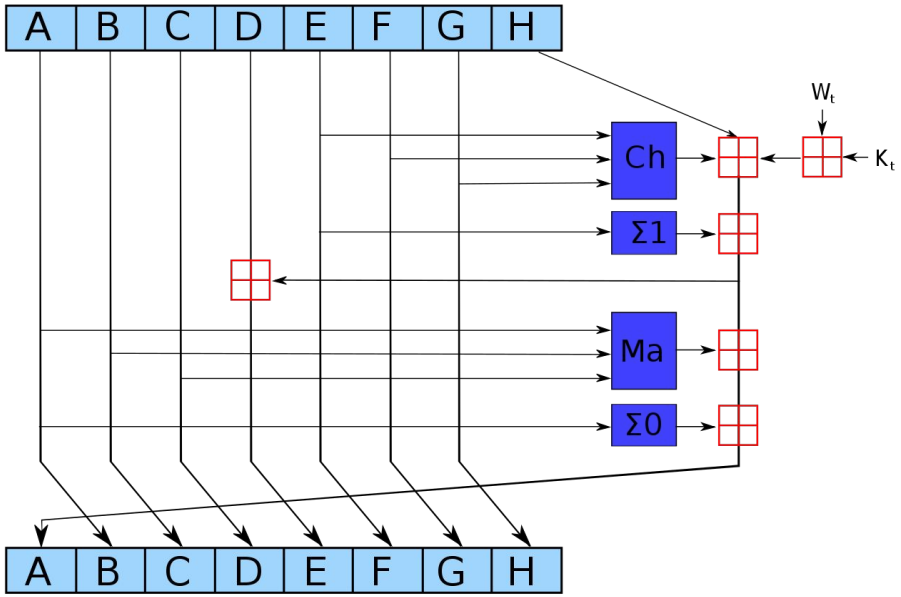
Характеристики компьютера, на котором проводились расчёты: Win10/x64, 8GB RAM, GTX 1050, Intel Core i5.

График построен с помощью программы [1], calc_time.py из приложения.

SHA-256. Предварительная обработка сообщения

В хеш-функции SHA-2 паддинг состоит из единичного бита и k нулевых бит таких, что $length + k + 1 \equiv 448 \, mod \, 512$, длина дописывается в конец.

SHA-256. Сжимающая функция



Одна итерация сжимающей функции SHA-256, где

$$Ch(E,F,G) = (E \,\&\, F) \oplus (\overline{E} \,\&\, G)$$
$$Ma(A,B,C) = (A \,\&\, B) \oplus (\mathrm{A} \,\&\, \mathrm{C}) \oplus (\mathrm{B} \,\&\, \mathrm{C})$$
$$\mathbf{\Sigma}_0(A) = (A \ggg 2) \oplus (\mathrm{A} \ggg 13) \oplus (\mathrm{A} \ggg 22)$$
$$\mathbf{\Sigma}_1(E) = (E \ggg 6) \oplus (\mathrm{E} \ggg 11) \oplus (\mathrm{E} \ggg 25)$$

сложение, обозначенное красным цветом, производится по модулю 2^{32}

Демонстрация работы хеш-функции SHA-256

Для демонстрации работы SHA-256 была использована функция sha256 из [2].

Сообщение	Свёртка
«1» 1024 раза	b4a38f5f5a21b6ba9f2008878beaa3a708866c66cbada71ba28ec359b5a53c3c
«1» 1023 раза и «0» в конце	2c5df9acaa75587d0fd3ac3b81a73e997c0b12624cb90512349328ecea42793d

Демонстрация работы хеш-функции SHA-256

Сообщение	Свёртка
«»	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855

Длина выходного массива остается прежней, даже если в качестве входных данных указать пустую строку. В таком случае хешируется лишь паддинг.

Заключение

Приложение

Все программы доступны по ссылке `github.com/ulnsig/hash-functions` .

Список используемых сточников

1. А. П. Алфёров, А.Ю. Зубов, А.С. Кузьмин, А.В. Черёмушкин «Основы Криптографии», 2-е издание – Москва, 2002.

2. Криптографические методы защиты информации: Учебно-методическое пособие: Том 1 / Бабаш А.В., - 2-е изд.

3. https://www.crypto101.io/

4. https://wikipedia.org/

5. https://habr.com/en/post/461163/