Хеширование строк

1 Хеш-функции

Хеш-функция предназначена для свертки входного массива любого размера в битовую строку, для MD5 длина выходной строки равна 128 битам. Для чего это нужно? К примеру у вас есть два массива, а вам необходимо быстро сравнить их на равенство, то хеш-функция может сделать это за вас, если у двух массивов хеши разные, то массивы гарантировано разные, а в случае равенства хешей — массивы скорее всего равны. [2]

Хеш-функции используются в криптографических алгоритмах, электронных подписях, кодах аутентификации сообщений, обнаружении манипуляций, сканировании отпечатков пальцев, контрольных суммах (проверка целостности сообщений), хеш-таблицах, хранении паролей и многом другом. К примеру, скачивая файл из интернета, вы часто видите рядом с ним строку вида b10a8db164e0754105b7a99be72e3fe5 — это и есть хеш, прогнав этот файл через алгоритм MD5 вы получите такую строку, и, если хэши равны, можно с большой вероятностью утверждать что этот файл действительно подлинный.

2 MD5

MD5 (англ. Message Digest 5) — 128-битный алгоритм хеширования, разработанный профессором Рональдом Л. Ривестом из Массачусетского технологического института (Massachusetts Institute of Technology, MIT) в 1991 году. Предназначен для создания «отпечатков» или дайджестов сообщения произвольной длины и последующей проверки их подлинности. Широко применялся для проверки целостности информации и хранения хешей паролей. [4]

Алгоритм производит хеш со значением в 128 битов. Широко используется для проверки целостности данных. Не подходит для использования в иных областях по причине уязвимости в безопасности MD5. (рис.1)

Алгоритм состоит из пяти шагов:

1.Append Padding Bits

В исходную строку дописывают единичный байт 0x80, а затем дописывают нулевые биты, до тех пор, пока длина сообщения не будет сравнима с 448 по модулю

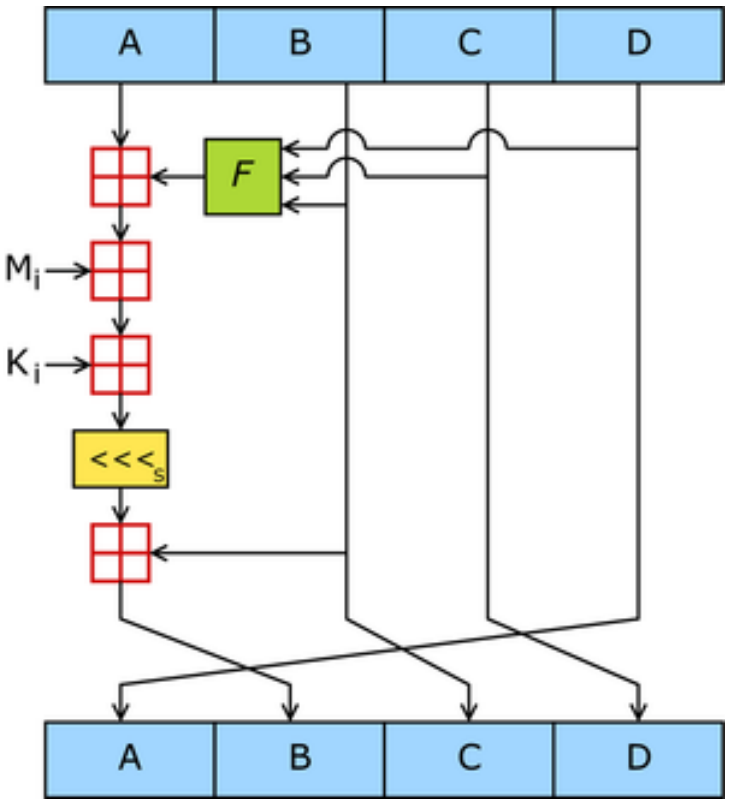

Рис. 1: Схема работы алгоритма MD5

512. То есть дописываем нули до тех пор, пока длина нового сообщения не будет равна [длина] = (512*N+448), где N — любое натуральное число, такое, что это выражение будет наиболее близко к длине блока.

2.Append Length

Далее в сообщение дописывается 64-битное представление длины исходного сообщения.

3.Initialize MD Buffer

На этом шаге инициализируется буффер

word A: 01 23 45 67

word B: 89 ab cd ef

word C: fe dc ba 98

word D: 76 54 32 10

Как можно заметить буффер состоит из четырех констант, предназначенный для сбора хэша.

4.Process Message in 16-Word Blocks

На четвертом шаге в первую очередь определяется 4 вспомогательные логические функции, которые преобразуют входные 32-битные слова, в, как ни странно, в 32-битные выходные.

$$F(X,Y,Z) = XYvnot(X)Z \tag{1}$$

$$G(X,Y,Z) = XZvYnot(Z) \tag{2}$$

$$H(X,Y,Z) = XxorYxorZ \tag{3}$$

$$I(X,Y,Z) = Yxor(Xvnot(Z)) \tag{4}$$

5.Output

Выводя побайтово буффер ABCD начиная с A и заканчивая D получим наш хеш. [3]

В листинге 1 представлен пример MD5-хеширования.

Листинг 1: MD5

```
import hashlib
hash_object = hashlib.md5(b'Hello World')
print(hash_object.hexdigest())
```

3 SHA-1

Secure Hash Algorithm 1 — алгоритм криптографического хеширования. Описан в RFC 3174. Для входного сообщения произвольной длины (длина сообщения варьируется от 160 до 512 бит) алгоритм генерирует 160-битное (20 байт) хеш-значение, называемое также дайджестом сообщения, которое обычно отображается как шестнадцатеричное число длиной в 40 цифр. [1] Используется во многих криптографических приложениях и протоколах. Также рекомендован в качестве основного для государственных учреждений в США. (рис.2)

Код 2 принимает строку "Hello World"и выводит дайджест HEX данной строки. hexdigest возвращает строку HEX, что представляет хеш, и в случае, если вам нужна последовательность байтов, нужно использовать дайджест. [5]

Листинг 2: SHA-1

```
import hashlib
hash_object = hashlib.sha1(b'Hello World')
hex_dig = hash_object.hexdigest()
print(hex_dig)
```

Предположим, нам нужно хешировать строку "Hello Word"с помощью функции SHA-1. Тогда результатом будет 0a4d55a8d778e5022fab701977c5d840bbc486d0.

Список литературы

[1] Шнайер Б. *Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си.* Litres, 2002.

[2] Вирт Н. *Алгоритмы и структуры данных.* Litres, 1989.

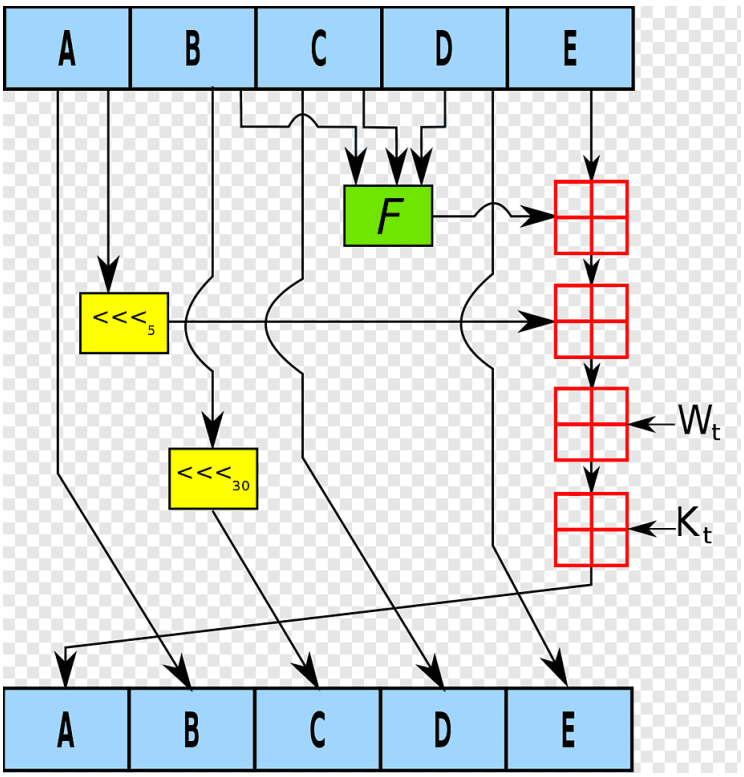

Рис. 2: Схема работы алгоритма MD5

[3] Риверст Р. *The MD5 Message-Digest Algorithm.* Litres, 1992.

[4] Ван Сяоюнь. *How to Break MD5 and Other Hash Functions.* Litres, 2005.

[5] Н. Фергюсон and Б. Шнайер. *Практическая криптография.* Litres, 2004.