MD5 ХЭШ-ФУНКЦИЯ. ОБЩИЕ СВЕДЕНИЯ, ПОДВОДНЫЕ КАМНИ

ИНФОБЕЗ production
Шаров А.И.
Точилкин А.И.

Общие сведения

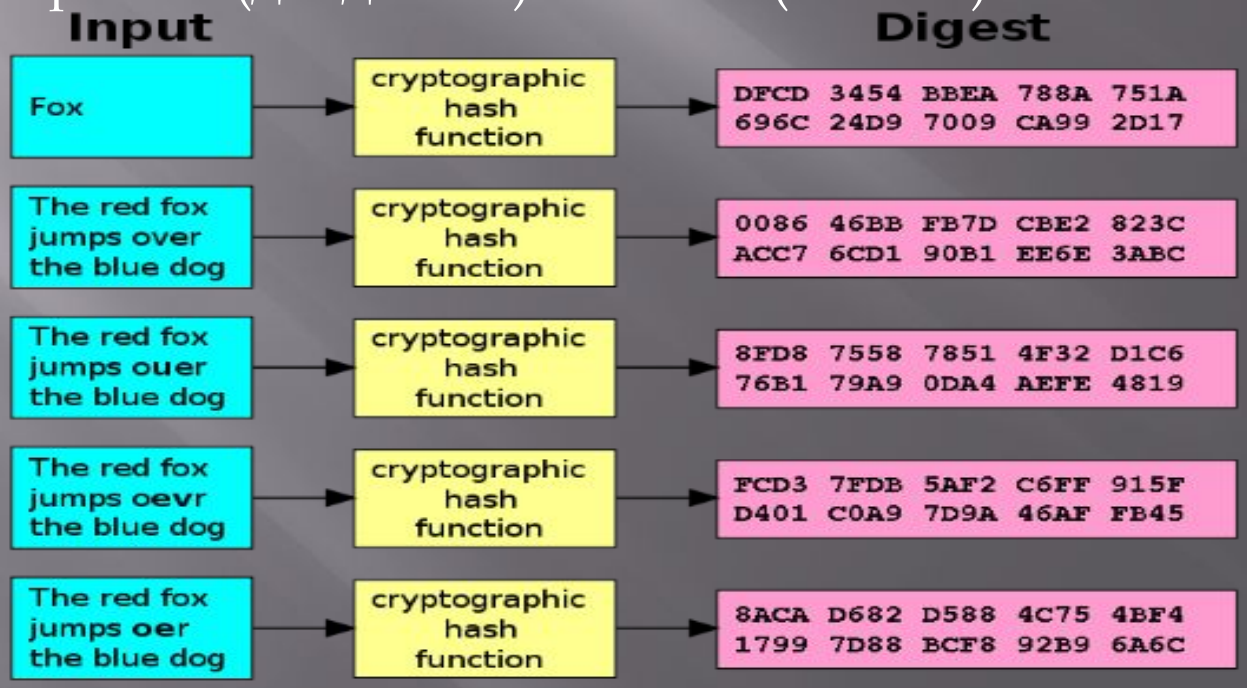
- **MD5** (англ. *Message Digest 5*) — 128-битный алгоритм хеширования, разработанный профессором Рональдом Л. Ривестом из MIT в 1991 году. Предназначен для создания «отпечатков» или «дайджестов» сообщений произвольной длины и последующей проверки их подлинности.

- **Дайджест** – результат преобразования входного сообщения произвольной длины в выходную фиксированной длины.

Принцип работы

- На вход алгоритма поступает входной поток данных, хеш которого необходимо найти. Длина сообщения может быть любой (в том числе нулевой). Далее входные данные преобразуются с помощью алгоритма хеш-функции (в нашем случае md5) и на выходе мы получаем последовательность из 32 шестнадцатиричных цифр.

  Размер хеша (дайджеста) - 128 бит (16 байт)



5 шагов алгоритма MD5. Шаги 1 и 2

- **Шаг 1. Выравнивание потока**

Сначала дописывают единичный бит в конец потока(байт 0x80), затем необходимое число нулевых бит. Входные данные выравниваются так, чтобы их новый размер L' был сравним с 448 по модулю 512 ($L' = 512 \times N + 448$). Выравнивание происходит, даже если длина уже сравнима с 448.

- **Шаг 2. Добавление длины сообщения**

В оставшиеся 64 бита дописывают 64-битное представление длины данных (количество бит в сообщении) до выравнивания. Сначала записывают младшие 4 байта. Если длина превосходит $2^{64} - 1$, то дописывают только младшие биты. После этого длина потока станет кратной 512. Вычисления будут основываться на представлении этого потока данных в виде массива слов по 512 бит.

Шаг 3

- **Шаг 3. Инициализация буфера**

```
A = 01 23 45 67;
B = 89 AB CD EF;
C = FE DC BA 98;
D = 76 54 32 10.
```

Для вычислений инициализируются 4 переменных размером по 32 бита и задаются начальные значения шестнадцатеричным числами (шестнадцатеричное представление, сначала младший

В этих переменных будут храниться результаты промежуточных вычислений. Начальное состояние ABCD называется инициализирующим вектором.

Определим ещё функции и константы, которые нам понадобятся для вычислений.

Потребуются 4 функции для четырёх раундов. Введём функции от трёх параметров — слов, результатом также будет слово.

1 раунд $FunF(X,Y,Z) = (X \wedge Y) \vee (\neg X \wedge Z)$.

2 раунд $FunG(X,Y,Z) = (X \wedge Z) \vee (\neg Z \wedge Y)$.

3 раунд $FunH(X,Y,Z) = X \oplus Y \oplus Z$.

4 раунд $FunI(X,Y,Z) = Y \oplus (\neg Z \vee X)$.

Определим таблицу констант T[1..64] — 64-элементная таблица данных, построенная следующим образом: $T[i] = int\ (4294967296 * |\ sin(i)\ |\ )$, где $4294967296 = 2^{32}$.

Выровненные данные разбиваются на блоки (слова) по 32 бита, и каждый блок проходит 4 раунда из 16 операторов. Все операторы однотипны и имеют вид: [abcd k s i], определяемый как $a = b + ((a + Fun(b,c,d) + X[k] + T[i]) <<< s)$, где X — блок данных. X[k] = M [n * 16 + k], где k — номер 32-битного слова из n-го 512-битного блока сообщения, и s — циклический сдвиг влево на s бит полученого 32-битного аргумента.

Криптоанализ

На данный момент существуют несколько видов «взлома» хешей MD5 — подбора сообщения с заданным хешем:

- Перебор по словарю

PasswordsPro, MD5BFCPF, John The Ripper.

- RainbowCrack
- Brute-force

Примеры использования

- MD5 позволяет получать относительно надёжный идентификатор для блока данных. Такое свойство алгоритма широко применяется в разных областях. Оно позволяет искать дублирующиеся файлы на компьютере, сравнивая MD5 файлов, а не их содержимое. Как пример, dupliFinder — графическая программа под Windows и Linux. Такой же поиск может работать и в интернете.
- С помощью MD5 проверяют целостность скачанных файлов — так, некоторые программы идут вместе со значением хеша. Например, диски для инсталляции.
- MD5 используется для хеширования паролей.

В системе UNIX каждый пользователь имеет свой пароль и его знает только пользователь.

Радужные таблицы

- Радужная таблица (англ. rainbow table) — специальный вариант таблиц поиска, использующий механизм уменьшения времени поиска за счет увеличения занимаемой памяти. Радужные таблицы используется для вскрытия паролей, преобразованных при помощи необратимой хеш-функции.

Генерация rainbow table

- Радужная таблица создается построением цепочек возможных паролей. Каждая цепочка начинается со случайного возможного пароля, затем подвергается действию хеш-функции и функции редукции. Данная функция преобразует результат хеш-функции в некоторый возможный пароль. Промежуточные пароли в цепочке отбрасываются и в таблицу записывается только первый и последний элементы цепочек. Создание таблиц требует времени и памяти, но они позволяют очень быстро (по сравнению с обычными методами) восстановить исходный пароль.

Поиск в rainbow table

- Для восстановления пароля данное значение хеш-функции подвергается функции редукции и ищется в таблице. Если не было найдено совпадения, то снова применяется хеш-функция и функция редукции. Данная операция продолжается, пока не будет найдено совпадение. После нахождения совпадения, цепочка содержащая его, восстанавливается для нахождения отброшенного значения, которое и будет искомым паролем.

Подробнее про генерацию радужной таблицы

- 1. Фиксируется рабочий алфавит, то есть задается множество Q всех возможных ключей. 2. Фиксируется элемент q из множества Q и вычисляется значение h хэш-функции на нем. 3. При помощи некоторой «срезающей» функции R из хэша генерируется ключ, принадлежащий множеству Q: q=R(h). Если число элементов в цепочке меньше заданного, осуществляется переход к пункту 2.
- Когда будет достигнута требуемая длина цепочки, в таблицу будут записаны первый и последний её элементы
- Далее, по вышеуказанному алгоритму, генерируется некоторое количество цепочек

Подробнее про поиск в радужной таблице

- Задается значение хэш-функции, для которого необходимо получить коллизию (нахождение соответствия). При помощи срезающей функции R строится значение ключа K0, из которого выводится по описанному алгоритму цепочка поиска длиной не более чем t элементов. Если в таблице есть искомый ключ, то один из сгенерированных элементов новой цепочки будет являться терминальным элементом нашей таблицы. Далее не составляет труда по известному начальному элементу вывести всю соответствующую терминалу цепочку, включая элемент, непосредственно предшествующий начальному значению K0, то есть ключ, который мы ищем. Но если же в сгенерированных таблицах нет коллизия, то нельзя дать точного ответа сколько нужно сгенерировать цепочек, чтобы они покрывали все множество возможных ключей.

Защита от радужных таблиц

- Метод защиты очень прост, но, почему-то, до сих пор используется далеко не везде. Заключается он в добавлении «соли» («затравки») к пользовательскому паролю, и уже потом использование хеш функции. По сути «соль» просто увеличивает длину пароля
- Хеш = MD5(пароль + соль), где + - конкатенация
- Для восстановления такого пароля взломщику необходимы таблицы для всех возможных значений затравки. При использовании такой схемы, затравка должна быть достаточно длинной (6-8 символов), случайной и различной для каждого пароля.