Достоинства и недостатки алгоритма MD5. Сравнение MD5 с SHA.

Группа *016*

Студент *Жилкин М. В.*

Содержание

Алгоритм MD5

Общие сведения

Алгоритм MD5 был создан в 1991 году профессором Массачусетского Технологического Института (MIT, Massachusetts Institute of Technlogy) Рональдом Райвестом (Ronald Rivest) в целях создания цифровых подписей. Он предназначен для использования на 32-битных машинах и является более безопасным, нежели алгоритм MD4, который был сломан. MD5 - односторонняя хэш-функция, то есть, зная лишь результат преобразования, невозможно восстановить исходную информацию.

Таким образом, алгоритм MD5 преобразует исходную информацию в число фиксированной длины, называемое "дайджестом сообщения" (message digest). Использование MD5 позволяет сравнить дайджест сообщения с опубликованным, чтобы убедиться, что данное сообщение полностью совпадает с оригинальным, то есть, не было повреждено или изменено. Данная процедура сравнения называется "проверка хэша" (hashcheck).

Пример.

Проверочная сумма MD5 (MD5 checksum) пустого сообщения:

d41d8cd98f00b204e9800998ecf8427e

Цифровая подпись

Цифровая подпись - базовый элемент в криптографии, играющий фундаментальную роль в аутентификации и авторизации. Задача цифровой подписи - идентифицировать действующее лицо с передаваемой информацией. Процесс подписи заключается в преобразовании исходного сообщения и некоторой секретной информации в собственно цифорвую подпись.

Как и обычная подпись, цифровая подпись должна гарантировать, что сообщение было отправлено именно тем, кто называет себя отправителем. Цифровые подписи особенно важны в электронной коммерции и являются ключевыми элементами в различных схемам аутентификации.

Цифровая подпись должна обладать определенной устойчивостью, иначе целесообразность ее использования будет под вопросом. Алгоритм MD5 является

одним из многих алгоритмов, призванных гарантировать необходимый уровень безопасности.

Проверка целостности

Проверочные суммы MD5 широко используются для проверки целостности скачанных файлов. Сравнивая сумму MD5 скачанного файла с опубликованным значением, пользователь имеет возможность убедиться, что файл, который он получил, абсолютно идентичен оригиналу, значит, не содержит вирусов.

Пример утилиты для проверки сумм MD5 можно найти по адресу www.fastsum.com

Устойчивость MD5

Алгоритм MD5 широко использовался и используется, и изначально считалось, что он абсолютно криптоустойчив. Однако, в 1994 году была открыта уязвимость, которая поставила под вопрос дальнейшее использование алгоритма. Было показано, что возможно создавать пары сообщений, имеющие одну и ту же проверочную сумму. Правда, в отличие от MD4, по-прежнему считается, что крайне сложно создать сообщение с заданной проверочной суммой. В 2004 году был запущен распределенный проект под названием MD5CRK (www.md5crk.com) с целью обнаружения уязвимостей алгоритма.

Алгоритм работы MD5 детально рассмотрен в Приложении 2.

MAC

Определение

MAC (Message Authentication Code) - последовательность бит, генерируемая криптографической хэш-функцией как из самих данных, так и из секретного ключа и используемая для аутентификации.

MD5-MAC

MD5-MAC является простым и эффективным алгоритмом получения MAC с помощью MD5. Размер ключа - 128 бит. Итоговый размер MAC - 64 бита.

Из 16-байтного ключа K получается три 16-байтных ключа: K_0, K_1 и K_2. K_1 разбивается на четыре 32-битные подстроки: $K_1[0]$, $K_1[1]$, $K_1[2]$ и $K_1[3]$. Далее MD5-MAC вычисляется по обычному алгоритма MD5 со следующими изменениями:

1. Четвертое слово начального буфера в алгоритме MD5 заменяется на K_0.
2. Значение $K_1[i]$ добавляется по модулю 2^{32} ко всем константам в i-том раунде.
3. После последнего блока, который содержит незначащие биты и значение длины исходного сообщения, добавляется следующий 64-байтный блок:

$$K_2; K_2 + T_0; K_2 + T_1; K_2 + T_2$$

T_i - "магическая строка", заданная последовательность битов.

4. Значение MD5-MAC состоит из начальных 64 битов полученного результата.

Коллизии в MD5

Определение

Под коллизией (collision) хэш-функции понимается получение одного и того же значения для разных сообщений при идентичном начальном буфере. Если же начальные буферы различаются, то совпадение выходных значений (как для разных сообщений, так и для одинаковых) называется псевдоколлизией (pseudo-collision).

Уязвимость MD5

В 1993 году Bert den Boer и Antoon Bosselaers показали, как можно обнаружить псевдоколлизии в алгоритме MD5. Matt Robshaw так прокомментировал эту атаку:

"На самом деле псевдоколлизия возникает при инициализации буфера из четырех слов при запуске алгоритма MD5 двумя разными значениями. Эти значения различаются только своими старшими разрядами в каждом слове. Для обоих буферов используется одно и то же сообщение, при этом получается одинаковый дайджест сообщения.

Если бы можно было выбрать одно и то же стартовое значение для буфера (необязательно то, что применяется в алгоритме), и затем выбрать два разных сообщения, возможно, различающихся лишь несколькими битами в каком-нибудь слове, таким образом, что получался бы один и тот же дайджест, то это было бы намного более серьезной уязвимостью."

Алгоритм SHA-1

Общие сведения

Алгоритм SHA-1 (Secure Hash Algorithm) был разработан Национальным Агентством Безопасности (NSA, National Security Agency) и опубликован Национальным Институтом Стандартов и Технологии (NIST, National Institute of Standards and Technology). Этот алгоритм сопоставляет сообщению с максимальной длиной 2^{64} бита дайджест длиной 160 бит.

История

Первоначальная спецификация алгоритма была опубликована в 1993 году под названием Secure Hash Standard, FIPS PUBS 180 (Federal Information Processing Standards Publications). Данную версию теперь часто называют SHA-0. NSA отказалась от нее вскоре после публикации и заменила улучшенной версией, опубликованной в 1995 году в FIPS PUBS 180-1 и обычно называемой SHA-1. Согласно заявлениям NSA, это было сделано в целях исправления ошибки в оригинальном алгоритме, которая уменьшала его криптографическую устойчивость. Однако, NSA не сообщила никакой дополнительной информации. Много спустя, на конференции Crypto в 1998 году два французских исследователя (F. Chabaud и A. Joux) представили атаку на алгоритм SHA-0, которая не работала на алгоритме SHA-1. Возможно, это и была ошибка, открытая NSA. Алгоритм SHA-1 был тщательно изучен криптографическим сообществом и пока не было найдено никаких уязвимостей. Таким образом, он считается вполне безопасным.

Вариации SHA

NIST также опубликовал три дополнительных вариации алгоритма SHA с более длинными дигетами. Их названия соответствуют длине дигета: SHA-256, SHA-384 и SHA-512. Они были впервые опубликованы в 2001 году в черновом варианте FIPS PUBS 180-2. Официальная версия FIPS PUBS 180-2, которая также включает SHA-1, была выпущена в качестве официального стандарта в 2002 году. Новые хэш-функции еще не были изучены с той же тщательностью, что SHA-1, поэтому их криптоустойчивость пока не подтверждена.

Различия MD5 и SHA

Алгоритм MD5 является последователем MD4 с улучшенным побитовым хэшированием, дополнительным раундом и улучшенным "лавинным эффектом" (avalanche effect). Более подробно различия MD4 и MD5 рассмотрены в Приложении 3.

Алгоритм SHA также происходит от MD4 и отличается от последнего расширенной трансформацией, дополнительным раундом и улучшенным "лавинным эффектом".

Несмотря на то, что до сих пор не был найден способ атаки MD5 на практике, тот факт, что такая атака возможна, внушает опасения. SHA-1 с более длинным дайджестом и устойчивостью к подобным атакам выглядит предпочтительнее.

Преимущество MD5 перед SHA-1 - производительность (см. Приложение 1).

Приложение 1. Сравнительная производительность MD5 и SHA-1.

Производительность измерялась в мегабайтах в секунду.

Алгоритм	Pentium 90 MHz	Power Mac 80 MHz	SPARC 4 110 MHz	DEC Alpha 200 MHz
MD5	13.1	3.1	5.1	8.5
SHA-1	2.5	1.2	2.0	3.3

Приложение 2. Описание работы алгоритма MD5.

Шаг 1. Добавление незначащих битов.

Исходное сообщение дополняется битами таким образом, чтобы его длина была сравнима с 448 по модулю 512. Если это условие уже выполняется, то все равно к сообщению приписываются 512 битов. Первый бит имеет значение 1, остальные - нули. Таким образом, приписывается как минимум один бит, максимум - 512.

Шаг 2. Добавление длины.

Сообщение дополняется 64 битами, выражающими длину оригинального сообщения. Таким образом, длина сообщения становится кратной 512.

Шаг 3. Инициализация MD буффера.

Для вычисления дайджеста сообщения используется буфер, состоящий из четырех слов. Ниже приведены их значения в шестнадцатиричном формате.

Слово A: 01 23 45 67

Слово B: 89 AB CD EF

Слово C: FE DC BA 98

Слово D: 76 54 32 10

Шаг 4. Обработка сообщения блоками по 16 слов.

Сначала определим четыре вспомогательные функции, каждая из которых принимает на входе три 32-битных слова и выдает одно 32-битное слово.

$$F(X,Y,Z) = X \cdot Y \cup not(X) \cdot Z$$
$$G(X,Y,Z) = X \cdot Z \cup Y \cdot not(Z)$$
$$H(X,Y,Z) = (X)\,xor\,(Y)\,xor\,(Z)$$
$$I(X,Y,Z) = (Y)\,xor\,(X \cup not(Z))$$

Функция F действует как условие: если X, то Y, иначе Z. Интересно заметить, что если биты X, Y и Z независимы, то каждый бит на выходе функции F так же является независимым. Этим же свойством обладают и функции G, H и I.

Следующий шаг использует таблицу $T[1..64]$, состоящую из 64 элементов и построенную на основе функции синуса. Пусть $T[i]$ означает i-ый эелемент таблицы, который равен целой части от произведения 4294967296 и $abs(\sin(i))$, где i означает радианы.

Далее приведен полный алгоритм.

Обработка блоками по 16 слов:

```
  For i = 0 to N/16-1 do

    For j = 0 to 15 do
      Set X[j] to M[i*16+j].

    End of j

    AA = A
    BB = B
    CC = C
    DD = D
```

Пусть [ABCD k s i] обозначает операцию A = B + ((A + F(B, C, D) + X[k] + T[i]) <<< s):

```
    [ABCD  0  7  1]  [DABC  1 12  2]  [CDAB  2 17  3]  [BCDA  3 22  4]
    [ABCD  4  7  5]  [DABC  5 12  6]  [CDAB  6 17  7]  [BCDA  7 22  8]
    [ABCD  8  7  9]  [DABC  9 12 10]  [CDAB 10 17 11]  [BCDA 11 22 12]
    [ABCD 12  7 13]  [DABC 13 12 14]  [CDAB 14 17 15]  [BCDA 15 22 16]
```

Пусть [ABCD k s i] обозначает операцию A = B + ((A + G(B, C, D) + X[k] + T[i]) <<< s):

```
    [ABCD  1  5 17]  [DABC  6  9 18]  [CDAB 11 14 19]  [BCDA  0 20 20]
    [ABCD  5  5 21]  [DABC 10  9 22]  [CDAB 15 14 23]  [BCDA  4 20 24]
    [ABCD  9  5 25]  [DABC 14  9 26]  [CDAB  3 14 27]  [BCDA  8 20 28]
    [ABCD 13  5 29]  [DABC  2  9 30]  [CDAB  7 14 31]  [BCDA 12 20 32]
```

Пусть [ABCD k s i] обозначает операцию A = B + ((A + H(B, C, D) + X[k] + T[i]) <<< s):

```
    [ABCD  5  4 33]  [DABC  8 11 34]  [CDAB 11 16 35]  [BCDA 14 23 36]
    [ABCD  1  4 37]  [DABC  4 11 38]  [CDAB  7 16 39]  [BCDA 10 23 40]
    [ABCD 13  4 41]  [DABC  0 11 42]  [CDAB  3 16 43]  [BCDA  6 23 44]
    [ABCD  9  4 45]  [DABC 12 11 46]  [CDAB 15 16 47]  [BCDA  2 23 48]
```

Пусть [ABCD k s i] обозначает операцию A = B + ((A + I(B, C, D) + X[k] + T[i]) <<< s):

```
    [ABCD  0  6 49]  [DABC  7 10 50]  [CDAB 14 15 51]  [BCDA  5 21 52]
    [ABCD 12  6 53]  [DABC  3 10 54]  [CDAB 10 15 55]  [BCDA  1 21 56]
    [ABCD  8  6 57]  [DABC 15 10 58]  [CDAB  6 15 59]  [BCDA 13 21 60]
```

[ABCD 4 6 61] [DABC 11 10 62] [CDAB 2 15 63] [BCDA 9 21 64]

Затем прибавляем к каждому слову буфера его первоначальное значение:

A = A + AA
B = B + BB
C = C + CC
D = D + DD

End of i

Итоговый дайджест сообщения представляет собой соединение всех слов буфера.

Приложение 3. Различия алгоритмов MD4 и MD5.

- В алгоритме MD5 был добавлен один раунд в обработке блоками по 16 слов.
- На каждом шаге прибавляется уникальная добавочная константа.
- Изменена функция G для большей симметричности.
- На каждом шаге прибавляется результат предыдущего шага.
- Порядок работы с входными словами в раундах 2 и 3 был изменен для большего различия.
- Были оптимизированы сдвиги в каждом раунде.

Список использованных источников

- IP Authentication using keyed MD5 by Bart Preneel (ESAT, K.U.Leuven, Belgium) Bart.Preneel@esat.kuleuven.ac.be
- MD5 vs SHA-1, Performance and Pedigree by Masatake Ohta mohta@necom830.hpcl.titech.ac.jp
- MD5 vs SHA by Ryan Malayter rmalayter@bai.org
- www.secure-hash-algorithm-md5-sha-1.co.uk
- www.webopedia.com