Государственный Университет Молдовы

Факультет Математики и Информатики

Департамент Информатики

“Криптография и информационная безопасность”

Индивидуальная работа 2

Проверил: Чербу Ольга

Выполнил: Чобану Артём

Кишинев 2022

**Алгоритм SHA1**

**SHA1** — алгоритм криптографического хеширования. Для входного сообщения произвольной длины генерирует 160 битное хеш-значение, называемое дайджестом, и обычно отображается как 16-ричное число длиной из 40 цифр. Используется во многих криптографических приложениях и протоколах. Также рекомендован в качестве основного для государственных учреждений США.

SHA-1 реализует [хеш-функцию](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), 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-блок Mi равен hi = f (Mi, hi-1)

Хеш-значением всего сообщения является выход последнего блока.

**Алгоритм:**

1. Инициализация

Исходное сообщение разбивается на блоки по 512 бит в каждом. Последний блок дополняется до длины, кратной 512 бит. Сначала добавляется 1 (бит), а потом — нули, чтобы длина блока стала равной 512 — 64 = 448 бит. В оставшиеся 64 бита записывается длина исходного сообщения в битах (в [big-endian](https://ru.wikipedia.org/wiki/Big-endian" \o "Big-endian) формате). Если последний блок имеет длину более 447, но менее 512 бит, то дополнение выполняется следующим образом: сначала добавляется 1 (бит), затем — нули вплоть до конца 512-битного блока; после этого создается ещё один 512-битный блок, который заполняется вплоть до 448 бит нулями, после чего в оставшиеся 64 бита записывается длина исходного сообщения в битах (в big-endian формате). Дополнение последнего блока осуществляется всегда, даже если сообщение уже имеет нужную длину.

Инициализируются пять 32-битовых переменных.

A = 0x67452301

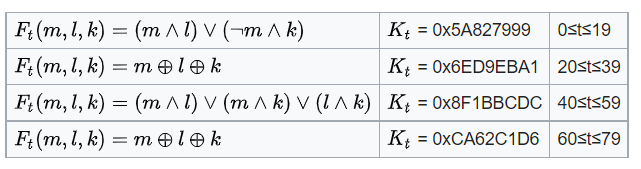
B = 0xEFCDAB89

C = 0x98BADCFE

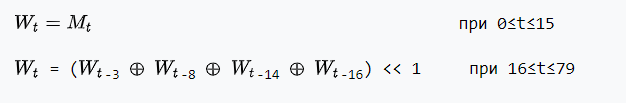
D = 0x10325476

E = 0xC3D2E1F0

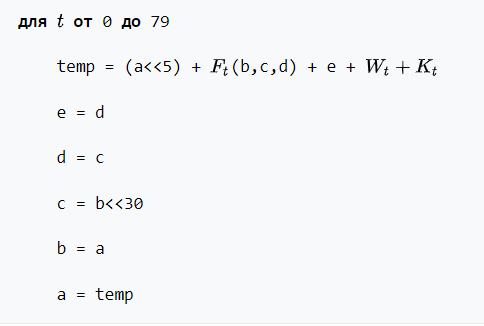
1. Определяются четыре нелинейные операции и четыре константы.



1. Главный цикл итеративно обрабатывает каждый 512-битный блок. В начале каждого цикла вводятся переменные a, b, c, d, e, которые инициализируются значениями A, B, C, D, E, соответственно. Блок сообщения преобразуется в 80 32-битных слов по правилу:



Далее:



Псевдокод:

A = 0x67452301  
B = 0xEFCDAB89  
C = 0x98BADCFE  
D = 0x10325476  
E = 0xC3D2E1F0  
  
inputBytes.Add(1)  
while (inputBytes.Length \* 8 % 512 < 448)  
{  
 inputBytes.Add(0);  
}  
inputBytes.AddRange(GetBytes(inputBytes.Length));  
  
//Везде деление или умножение на 8 т.к. измерение должно быть в битах, а не в байтах.  
for (int x = 0; x < inputBytes.Count; x += 512 / 8)  
{  
 var w = ToIntArray(inputBytes.GetRange(x, 512 / 8))  
  
 for (int i = 16; i <= 79; i++)  
 {  
 var wi = w[i - 3] ^ w[i - 8] ^ w[i - 14] ^ w[i - 16]  
  
 w.Add(wi <<< 1))  
 }  
  
 a = A;  
 b = B;  
 c = C;  
 d = D;  
 e = E;  
  
 for (int i = 0; i <= 79; i++)  
 {  
 if (0 <= i <= 19)  
 {  
 f = (b & c) | (~b & d)  
 k = 0x5A827999  
 }  
 else if (20 <= i <= 39)  
 {  
 f = b ^ c ^ d  
 k = 0x6ED9EBA1  
 }  
 else if (40 <= i <= 59)  
 {  
 f = (b & c) | (b & d) | (c & d)  
 k = 0x8F1BBCDC  
 }

else if (60 <= i <= 79)

{  
 f = b ^ c ^ d  
 k = 0xCA62C1D6  
 }  
  
 var temp = a <<< 5 + f + e + k + w[i]  
 e = d  
 d = c  
 c = b <<< 30  
 b = a  
 a = temp  
 }  
  
 A += a  
 B += b  
 C += c  
 D += d  
 E += e  
}

digest = A append B append C append D append E