5.8. Хэш-функция SHA-2

В 2001 году NIST принял в качестве стандарта три хэш-функции с существенно большей длиной хэш-кода. Часто эти хэш-функции называют *SHA-2* или SHA-256, SHA-384 и SHA-512 (соответственно, в названии указывается длина создаваемого ими хэш-кода). Эти алгоритмы отличаются не только длиной создаваемого хэш-кода, но и длиной обрабатываемого блока, длиной слова и используемыми внутренними функциями. Сравним характеристики этих хэш-функций.

Алгоритм	Длина сообщения (в битах)	Длина блока (в битах)	Длина слова (в битах)	Длина дайджеста сообщения (в битах)	Безопасность (в битах)
SHA-1	$<2^{64}$	512	32	160	80
SHA-256	$<2^{64}$	512	32	256	128
SHA-384	$<2^{128}$	1024	64	384	192
SHA-512	$<2^{128}$	1024	64	512	256

Под безопасностью здесь понимается стойкость к атакам типа "парадокса дня рождения".

В данных алгоритмах размер блока сообщения равен m бит. Для SHA-256 m = 512, для SHA-384 и SHA-512 m = 1024. Каждый алгоритм оперирует с w-битными словами. Для SHA-256 w = 32, для SHA-384 и SHA-512 w = 64. В алгоритмах используются обычные булевские операции над словами, а также сложение по модулю 2^w, правый сдвиг на n бит $SHR^n(x)$, где x - w-битное слово, и циклические (ротационные) правый и левый сдвиги на n бит $ROTR^n(x)$ и $ROTL^n(x)$, где x - w-битное слово.

SHA-256 использует шесть логических функций, при этом каждая из них выполняется с 32-битными словами, обозначенными как x, y и z. Результатом каждой функции тоже является 32-битное слово.

$Ch(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$

$Maj(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$

$\Sigma_0^{\{256\}}(x) = ROTR^2(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x)$

$\Sigma_1^{\{256\}}(x) = ROTR^6(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x)$

$\sigma_0^{\{256\}}(x) = ROTR^7(x) \oplus ROTR^{18}(x) \oplus SHR^3(x)$

$\sigma_1^{\{256\}}(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x)$

SHA-384 и SHA-512 также используют шесть логических функций, каждая из которых выполняется над 64-битными словами, обозначенными как x, y и z. Результатом каждой функции является 64-битное слово.

$Ch(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$

$Maj(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$

$\Sigma_0^{\{512\}}(x) = ROTR^{28}(x) \oplus ROTR^{34}(x) \oplus ROTR^{39}(x)$

$\Sigma_1^{\{512\}}(x) = ROTR^{14}(x) \oplus ROTR^{18}(x) \oplus ROTR^{41}(x)$

$\sigma_0^{\{512\}}(x) = ROTR^1(x) \oplus ROTR^8(x) \oplus SHR^7(x)$

$\sigma_1^{\{512\}}(x) = ROTR^{19}(x) \oplus ROTR^{61}(x) \oplus SHR^6(x)$

Предварительная подготовка сообщения, т.е. добавление определенных битов до целого числа блоков и последующее разбиение на блоки выполняется аналогично тому, как это делалось в *SHA-1* (конечно, с учетом длины блока каждой хэш-функции). После этого каждое сообщение можно представить в виде последовательности N блоков $M^{(1)}, M^{(2)}, \dots, M^{(N)}$.

Рассмотрим SHA-256. В этом случае инициализируются восемь 32-битных переменных, которые послужат промежуточным значением хэш-кода:

a, b, c, d, e, f, g, h

Основой алгоритма является модуль, состоящий из 64 циклических обработок каждого блока $M^{(i)}$:

$T_1 = h + \Sigma_1^{\{256\}}(e) + Ch(e, f, g) + K_t^{\{256\}} + W_t$

$T_2 = \Sigma_0^{\{256\}}(a) + Maj(a, b, c)$

$h = g$

$g = f$

$f = e$

$e = d + T_1$

$d = c$

$c = b$

$b = a$

$a = T_1 + T_2$

где $K_i^{\{256\}}$ - шестьдесят четыре 32-битных константы, каждая из которых является первыми 32-мя битами дробной части кубических корней первых 64 простых чисел.

W_t вычисляются из очередного блока сообщения по следующим правилам:

$W_t = M_t^{(i)}$, $0 \le t \le 15$

$W_t = \sigma_1^{\{256\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{256\}}(W_{t-15}) + W_{t-16}$, $16 \le t \le 63$

i-ое промежуточное значение хэш-кода $H^{(t)}$ вычисляется следующим образом:

$H_0^{(i)} = a + H_0^{(i-1)}$

$H_1^{(i)} = b + H_1^{(i-1)}$

$H_2^{(i)} = c + H_2^{(i-1)}$

$H_3^{(i)} = d + H_3^{(i-1)}$

$H_4^{(i)} = e + H_4^{(i-1)}$

$H_5^{(i)} = f + H_5^{(i-1)}$

$H_6^{(i)} = g + H_6^{(i-1)}$

$H_7^{(i)} = h + H_7^{(i-1)}$

Теперь рассмотрим SHA-512. В данном случае инициализируются восемь 64-битных переменных, которые будут являться промежуточным значением хэш-кода:

a, b, c, d, e, f, g, h

Основой алгоритма является модуль, состоящий из 80 циклических обработок каждого блока $M^{(i)}$:

$T_1 = h + \Sigma_1^{\{512\}}(e) + Ch(e, f, g) + K_t^{\{512\}} + W_t$

$T_2 = \Sigma_0^{\{512\}}(a) + Maj(a, b, c)$

$h = g$

$g = f$

$f = e$

$e = d + T_1$

$d = c$

$c = b$

$b = a$

$a = T_1 + T_2$

где $K_i^{\{512\}}$ - восемьдесят 64-битных констант, каждая из которых является первыми 64-мя битами дробной части кубических корней первых восьмидесяти простых чисел.

W_t вычисляются из очередного блока сообщения по следующим правилам:

$W_t = M_t^{(i)}$, $0 \le t \le 15$

$W_t = \sigma_1^{\{512\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{512\}}(W_{t-15}) + W_{t-16}$, $16 \le t \le 79$

i-ое промежуточное значение хэш-кода H(t) вычисляется следующим образом:

$H_0^{(i)} = a + H_0^{(i-1)}$

$H_1^{(i)} = b + H_1^{(i-1)}$

$H_2^{(i)} = c + H_2^{(i-1)}$

$H_3^{(i)} = d + H_3^{(i-1)}$

$H_4^{(i)} = e + H_4^{(i-1)}$

$H_5^{(i)} = f + H_5^{(i-1)}$

$H_6^{(i)} = g + H_6^{(i-1)}$

$H_7^{(i)} = h + H_7^{(i-1)}$

Рассмотрим SHA-384. Отличия этого алгоритма от SHA-512:

Другой начальный хэш-код $H^{(0)}$.

384-битный дайджест получается из левых 384 битов окончательного хэш-кода $H^{(N)}$: $H_0^{(N)} \| H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)} \| H_5^{(N)}$.