ГОСТ Р 34.11-94

Группа П85

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ИНФОРМАЦИОННАЯ ТЕХНОЛОГИЯ

КРИПТОГРАФИЧЕСКАЯ ЗАЩИТА ИНФОРМАЦИИ

ФУНКЦИЯ ХЭШИРОВАНИЯ

INFORMATION TECHNOLOGY. CRYPTOGRAPHIC DATA SECURITY. HASHING FUNCTION

ОКСТУ 5002

Дата введения 1995-01-01

ПРЕДИСЛОВИЕ

1 РАЗРАБОТАН Главным управлением безопасности связи Федерального агентства правительственной связи и информации и Всероссийским научно-исследовательским институтом стандартизации

ВНЕСЕН Техническим комитетом по стандартизации ТК 22 "Информационная технология" и Федеральным агентством правительственной связи и информации

2 ПРИНЯТ И ВВЕДЕН В ДЕЙСТВИЕ Постановлением Госстандарта России от 23.05.94 N 154

3 ВВЕДЕН ВПЕРВЫЕ

ВВЕДЕНИЕ

Расширяющееся применение информационных технологий при создании, обработке, передаче и хранении документов требует в определенных случаях сохранения конфиденциальности их содержания, обеспечения полноты и достоверности.

Одним из эффективных направлений защиты информации является криптография (криптографическая защита), широко применяемая в различных сферах деятельности в государственных и коммерческих структурах.

Криптографические методы защиты информации являются объектом

серьезных научных исследований и стандартизации на национальных, региональных и международных уровнях.

Настоящий стандарт определяет процедуру вычисления хэш-функции для любой последовательности двоичных символов.

Функция хэширования заключается в сопоставлении произвольного набора данных в виде последовательности двоичных символов и его образа фиксированной небольшой длины, что позволяет использовать эту функцию в процедурах электронной цифровой подписи для сокращения времени подписывания и проверки подписи. Эффект сокращения времени достигается за счет вычисления подписи только под образом подписываемого набора данных.

1 ОБЛАСТЬ ПРИМЕНЕНИЯ

Настоящий стандарт определяет алгоритм и процедуру вычисления хэш-функции для любой последовательности двоичных символов, которые применяются в криптографических методах обработки и защиты информации, в том числе для реализации процедур электронной цифровой подписи (ЭЦП) при передаче, обработке и хранении информации в автоматизированных системах.

Определенная в настоящем стандарте функция хэширования используется при реализации систем электронной цифровой подписи на базе асимметричного криптографического алгоритма по ГОСТ Р 34.10.

2 НОРМАТИВНЫЕ ССЫЛКИ

В настоящем стандарте использованы ссылки на следующие стандарты:

ГОСТ 28147-89 Системы обработки информации. Защита криптографическая. Алгоритмы криптографического преобразования.

ГОСТ Р 34.10-94 Информационная технология. Криптографическая защита информации. Процедуры выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма.

3 ОБОЗНАЧЕНИЯ

В настоящем стандарте используются следующие обозначения:

B^* - множество всех конечных слов в алфавите $B = (0,1)$. Чтение слов и нумерация знаков алфавита (символов) осуществляются справа налево (номер правого символа в слове равен единице, второго справа - двум и т.д.).

$|A|$ - длина слова $A \in B^*$.

$V_k(2)$ - множество всех бинарных слов длины k.

$A\|B$ - конкатенация слов $A, B \in B^*$ - слово длины $|A|+|B|$, в котором левые $|A|$ символов образуют слово A, а правые $|B|$ символов образуют слово B. Можно также использовать обозначение $A\|B = AB$.

A^k - конкатенация k экземпляров слова $A\,(A \in B^*)$.

$\langle N \rangle_k$ - слово длины k, содержащее двоичную запись вычета $N \pmod{2^k}$ неотрицательного целого числа N.

$\hat{A}$ - неотрицательное целое число, имеющее двоичную запись $A\,(A \in B^*)$.

$\oplus$ - побитовое сложение слов одинаковой длины по модулю 2.

$\oplus'$ - сложение по правилу $A \oplus' B = \langle \hat{A} + \hat{B} \rangle_k$, $(k = |A| = |B|)$.

M - последовательность двоичных символов, подлежащая хэшированию (сообщение в системах ЭЦП), $M \in B^*$.

h - хэш-функция, отображающая последовательность $M \in B^*$ в слово $h(M) \in V_{256}(2)$.

$E_K(A)$ - результат зашифрования слова A на ключе K с использованием алгоритма шифрования по ГОСТ 28147 в режиме простой замены $(K \in V_{256}(2),\ A \in V_{64}(2))$.

H - стартовый вектор хэширования.

$e := g$ - присвоение параметру e значения g.

4 ОБЩИЕ ПОЛОЖЕНИЯ

Под хэш-функцией h понимается зависящее от параметра [стартового вектора хэширования H, являющегося словом из $V_{256}(2)$] отображение

$$h : B^* \dashrightarrow V_{256}(2).$$

Для определения хэш-функции необходимы:

- алгоритм вычисления шаговой функции хэширования χ, т.е. отображения

$$\chi : V_{256}(2) \times V_{256}(2) \text{ ------- } \rightarrow V_{256}(2);$$

- описание итеративной процедуры вычисления значения хэш-функции h.

5 ШАГОВАЯ ФУНКЦИЯ ХЭШИРОВАНИЯ

Алгоритм вычисления шаговой функции хэширования включает в себя три части, реализующие последовательно:

- генерацию ключей - слов длины 256 битов;

- шифрующее преобразование - зашифрование 64-битных подслов слова H на ключах K_i (i = 1, 2, 3, 4) с использованием алгоритма по ГОСТ 28147 в режиме простой замены;

- перемешивающее преобразование результата шифрования.

5.1 Генерация ключей

Рассмотрим $X = (b_{256}, b_{255}, \ldots, b_i) \in V_{256}(2)$.

Пусть
$$X = x_4 \| x_3 \| x_2 \| x_1 =$$
$$= \eta_{16} \| \eta_{15} \| \ldots \| \eta_1 =$$
$$= \xi_{32} \| \xi_{31} \| \ldots \| \xi_1$$

где $x_i = (b_{i \times 64}, \ldots, b_{(i-1) \times 64+1}) \in V_{64}(2), \ i = \overline{1,4}$;

$\eta_j = (b_{j \times 16}, \ldots, b_{(j-1) \times 16+1}) \in V_{16}(2), \ j = \overline{1,16}$;

$\xi_k = (b_{k \times 8}, \ldots, b_{(k-1) \times 8+1}) \in V_8(2), \ k = \overline{1,32}$.

Обозначают $A(X) = (x_1 \oplus x_2) \| x_4 \| x_3 \| x_2$.

Используют преобразование $P : V_{256}(2) \text{ --------} \rightarrow V_{256}(2)$
слова $\xi_{32} \| \ldots \| \xi_1$ в слово, $\xi_{\varphi(32)} \| \ldots \| \xi_{\varphi 1)}$,

где $\varphi(i+1+4(k-1)) = 8i+k, \ i = 0 \div 3, \ k = 1 \div 8$.

Для генерации ключей необходимо использовать следующие исходные

данные:

- слова $H, M \in V_{256}(2)$;

- параметры: слова C_i (i=2, 3, 4), имеющие значения

$C_2 = C_4 = 0^{256}$ и $C_3 = 1^8 0^8 1^{16} 0^{24} 1^{16} 0^8 (0^8 1^8)^2 1^8 0^8 (0^8 1^8)^4 (1^8 0^8)^4$.

При вычислении ключей реализуется следующий алгоритм:

1 Присвоить значения

$i := 1$, $U := H$, $V := M$.

2 Выполнить вычисление

$W = U \boxplus V$, $K_1 = P(W)$

3 Присвоить $i := i + 1$.

4 Проверить условие $i = 5$.

При положительном исходе перейти к шагу 7. При отрицательном - перейти к шагу 5.

5 Выполнить вычисление

$U := A(U) \boxplus C_i$, $V := A(A(V))$, $W := U \boxplus V$, $K_i = P(W)$.

6 Перейти к шагу 3.

7 Конец работы алгоритма.

5.2 Шифрующее преобразование

На данном этапе осуществляется зашифрование 64-битных подслов слова H на ключах K_i (i = 1, 2, 3, 4).

Для шифрующего преобразования необходимо использовать следующие исходные данные:

$H = h_4 \| h_3 \| h_2 \| h_1$, $h_i \in V_{64}(2)$, $i = \overline{1,4}$

и набор ключей K_1, K_2, K_3, K_4.

Реализуют алгоритм зашифрования и получают слова

$s_i = E_{K_i}(h_i)$, где i=1, 2, 3, 4.

В результате данного этапа образуется последовательность

$S = s_4 \| s_3 \| s_2 \| s_1$.

5.3 Перемешиваюшее преобразование

На данном этапе осуществляется перемешивание полученной последовательности с применением регистра сдвига.

Исходными данными являются:

слова H, $M \in V_{256}(2)$ и слово $S \in V_{256}(2)$.

Пусть отображение

$\Psi : V_{256}(2) \text{--------} \rightarrow V_{256}(2)$

преобразует слово

$\eta_{16} \| \dots \| \eta_1$, $\eta_1 \in V_{16}(2)$, $i = \overline{1,16}$

в слово

$\eta_1 \oplus \eta_2 \oplus \eta_3 \oplus \eta_4 \oplus \eta_{13} \oplus \eta_{16} \| \eta_{16} \| \dots \| \eta_2$.

Тогда в качестве значения шаговой функции хэширования принимается слово

$\chi(M, H) = \Psi^{61}(H \oplus \Psi(M \oplus \Psi^{12}(S)))$,

где Ψ^i - i-я степень преобразования Ψ.

6 ПРОЦЕДУРА ВЫЧИСЛЕНИЯ ХЭШ-ФУНКЦИИ

Исходными данными для процедуры вычисления значения функции h является подлежащая хэшированию последовательность $M \in B^*$. Параметром является стартовый вектор хэширования H - произвольное фиксированное слово из $V_{256}(2)$.

Процедура вычисления функции h на каждой итерации использует следующие величины:

$M \in B^*$ - часть последовательности M, не прошедшая процедуры хэширования на предыдущих итерациях;

$H \in V_{256}(2)$ - текущее значение хэш-функции;

$\sum \in V_{256}(2)$ - текущее значение контрольной суммы;

$L \in V_{256}(2)$ - текущее значение длины обработанной на предыдущих итерациях части последовательности M.

Алгоритм вычисления функции h включает в себя этапы:

Этап 1

Присвоить начальные значения текущих величин

1.1 $M := M$

1.2 $H := H$

1.3 $\sum := 0^{256}$

1.4 $L := 0^{256}$

1.5 Переход к этапу 2

Этап 2

2.1 Проверить условие $|M| > 256$.

При положительном исходе перейти к этапу 3.

В противном случае выполнить последовательность вычислений:

2.2 $L := \langle \hat{L} + |M| \rangle_{256}$

2.3 $M' := 0^{256-|M|} \| M$

2.4 $\sum := \sum \boxplus' M'$

2.5 $H := \chi(M', H)$

2.6 $H := \chi(L, H)$

2.7 $H := \chi(\sum, H)$

2.8 Конец работы алгоритма

Этап 3

3.1 Вычислить подслово $M_s \in V_{256}(2)$ слова M $(M = M_p \| M_s)$. Далее выполнить последовательность вычислений:

3.2 $H := \chi(M_s, H)$

3.3 $L := \langle L + 256 \rangle_{256}$

3.4 $\sum := \sum \boxplus' M_s$

3.5 $M := M_p$

3.6 Перейти к этапу 2.

Значение величины H, полученное на шаге 2.7, является значением функции хэширования $h(M)$.

Проверочные примеры для вышеизложенной процедуры вычисления хэш-функции приведены в приложении А.

ПРИЛОЖЕНИЕ А

(справочное)

ПРОВЕРОЧНЫЕ ПРИМЕРЫ

Заполнение узлов замены π_1, π_2,..., π_8 и значение стартового вектора хэширования H, указанные в данном приложении, рекомендуется использовать только в проверочных примерах для настоящего стандарта.

А.1 Использование алгоритма ГОСТ 28147

В качестве шифрующего преобразования в приводимых ниже примерах используется алгоритм ГОСТ 28147 в режиме простой замены.

При этом заполнение узлов замены π_1, π_2,..., π_8 блока подстановки π следующее:

	8	7	6	5	4	3	2	1
0	1	D	4	6	7	5	E	4
1	F	B	B	C	D	8	B	A

2	D	4	A	7	A	1	4	9
3	0	1	0	1	1	D	C	2
4	5	3	7	5	0	A	6	D
5	7	F	2	F	8	3	D	8
6	A	5	1	D	9	4	F	0
7	4	9	D	8	F	2	A	E
8	9	0	3	4	E	E	2	6
9	2	A	6	A	4	F	3	B
10	3	E	8	9	6	C	8	1
11	E	7	5	E	C	7	1	C
12	6	6	9	0	B	6	0	7
13	B	8	C	3	2	0	7	F
14	8	2	F	B	5	9	5	5
15	C	C	E	2	3	B	9	3

В столбце с номером j, $j = \overline{1,8}$, в строке с номером i, $i = \overline{0,15}$, приведено значение $\pi_j(i)$ в шестнадцатеричной системе счисления.

А.2 Представление векторов

Последовательности двоичных символов будем записывать как строки шестнадцатеричных цифр, в которых каждая цифра соответствует четырем знакам ее двоичного представления.

А.3 Примеры вычисления значения хэш-функции

В качестве стартового вектора хэширования принимают, например, нулевой вектор

H	=	00000000	00000000	00000000	00000000
		00000000	00000000	00000000	00000000

А.3.1 Пусть необходимо выполнить хэширование сообщения

M	=	73657479	62203233	3D687467	6E656C20
		2C656761	7373656D	20736920	73696854

Выполняют присвоение начальных значений:

текста

M	=	73657479	62203233	3D687467	6E656C20
		2C656761	7373656D	20736920	73696854

хэш-функции

H	=	00000000	00000000	00000000	00000000
		00000000	00000000	00000000	00000000

суммы блоков текста

Σ	=	00000000	00000000	00000000	00000000
		00000000	00000000	00000000	00000000

длина текста

L = 00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000100

Так как длина сообщения, подлежащего хэшированию, равна 256 битам (32 байтам),

L = 00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000100

$M = M$ = 73657479 62203233 3D687467 6E656C20

2C656761 7373656D 20736920 73696854, то

нет необходимости дописывать текущий блок нулями,

$\Sigma = M'$ = 73657479 62203233 3D687467 6E656C20

2C656761 7373656D 20736920 73696854

Переходят к вычислению значения шаговой функции хэширования $\chi(M, H)$.

Вырабатывают ключи

K_1 = 733D2C20 65686573 74746769 79676120

626E7373 20657369 326C6568 33206D54

K_2 = 110C733D 0D166568 130E7474 06417967

		1D00626E	161A2065	090D326C	4D393320
K_3	=	80B111F3	730DF216	850013F1	C7E1F941
		620C1DFF	3ABAE91A	3FA109F2	F513B239
K_4	=	A0E2804E	FF1B73F2	ECE27A00	E7B8C7E1
		EE1D620C	AC0CC5BA	A804C05E	A18B0AEC

Осуществляют зашифрование 64-битных подслов блока H с помощью алгоритма по ГОСТ 28147.

Блок h_1 = 00000000 00000000 зашифровывают на ключе K_1 и получают s_1 = 42ABBCCE 32BC0B1B.

Блок h_2 = 00000000 00000000 зашифровывают на ключе K_2 и получают s_2 = 5203EBC8 5D9BCFFD.

Блок h_3 = 00000000 00000000 зашифровывают на ключе K_3 и получают s_3 = 8D345899 00FF0E28.

Блок h_4 = 00000000 00000000 зашифровывают на ключе K_4 и получают s_4 = E7860419 0D2A562D.

Получают

S	=	E7860419	0D2A562D	8D345899	00FF0E28
		5203EBC8	5D9BCFFD	42ABBCCE	32BC0B1B

Выполняют перемешивающее преобразование с применением регистра сдвига и получают

$\sum = \chi(M, H)$ = CF9A8C65 505967A4 68A03B8C 42DE7624

D99C4124 883DA687 561C7DE3 3315C034

Полагают $H = \Xi$, вычисляют $\chi(L, H)$:

K_1 = CF68D956 9AA09C1C 8C3B417D 658C24E3

50428833 59DE3D15 6776A6C1 A4248734

K_2 = 8FCF68D9 809AA09C 3C8C3B41 C7658C24

BB504288 2859DE3D 666676A6 B3A42487

K_3 = 4E70CF97 3C8065A0 853C8CC4 57389A8C

CABB50BD E3D7A6DE D1996788 5CB35B24

K_4 = 584E70CF C53C8065 48853C8C 1657389A

EDCABB50 78E3D7A6 EED19867 7F5CB35B

S = 66B70F5E F163F461 468A9528 61D60593

E5EC8A37 3FD42279 3CD1602D DD783E86

Ξ = 2B6EC233 C7BC89E4 2ABC2692 5FEA7285

DD3848D1 C6AC997A 24F74E2B 09A3AEF7

Вновь полагают $H = \Xi$ и вычисляют $\chi(\sum, H)$:

K_1	=	5817F104	0BD45D84	B6522F27	4AF5B00B
		A531B57A	9C8FDFCA	BB1EFCC6	D7A517A3
K_2	=	E82759E0	C278D950	15CC523C	FC72EBB6
		D2C73DA8	19A6CAC9	3E8440F5	C0DDB65A
K_3	=	77483AD9	F7C29CAA	EB06D1D7	841BCAD3
		FBC3DAA0	7CB555F0	D4968080	0A9E56BC
K_4	=	A 1157965	2D9FBC9C	088C7CC2	46FB3DD2
		7684ADCB	FA4ACA06	53EFF7D7	C0748708
S	=	2AEBFA76	A85FB57D	6F164DE9	2951A581
		C31E7435	4930FD05	1F8A4942	550A582D
Ξ	=	FAFF37A6	15A81669	2CFF3EF8	B68CA247
		E09525F3	9F811983	2EB81975	D366C4B1

Таким образом, результат хеширования есть

H	=	FAFF37A6	15A81669	1CFF3EF8	B68CA247
		E09525F3	9F811983	2EB81975	D366C4B1

А.3.2. Пусть необходимо выполнить хэширование сообщения

M = 7365 74796220 3035203D 20687467 6E656C20 73616820 65676173

73656D20 6C616E69 6769726F 20656874 2065736F 70707553

Так как длина сообщения, подлежащего хэшированию, равна 400 битам (50 байтам), то разбивают сообщение на два блока и второй (старший) блок дописывают нулями. В процессе вычислений получают:

ШАГ 1

H = 00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000000

M = 73616820 65676173 73656D20 6C616E69

6769726F 20656874 2065736F 70707553

K_1 = 73736720 61656965 686D7273 20206F6F

656C2070 67616570 616E6875 73697453

K_2 = 14477373 0C0C6165 1F01686D 4F002020

4C50656C 04156761 061D616E 1D277369

K_3 = CBFF14B8 6D04F30C 96051FFE DFFFB000

35094CAF 72F9FB15 7CF006E2 AB1AE227

K_4	=	EBACCB00	F7006DFB	E5E16905	B0B0DFFF
		BA1C3509	FD118DF9	F61B830F	F8C554E5
S	=	FF41797C	EEAADAC2	43C9B1DF	2E14681C
		EDDC2210	1EE1ADF9	FA67E757	DAFE3AD9
Ξ	=	F0CEEA4E	368B5A60	C63D96C1	E5B51CD2
		A93BEFBD	2634F0AD	CBBB69CE	ED2D5D9A

ШАГ 2

H	=	F0CEEA4E	368B5A60	C63D96C1	E5B51CD2
		A93BEFBD	2634F0AD	CBBB69CE	ED2D5D9A
M'	=	00000000	00000000	00000000	00007365
		74796220	3035203D	20687467	6E656C20
K_1	=	F0C6DDEB	CE3D42D3	EA968D1D	4EC19DA9
		36E51683	8BB50148	5A6FD031	60B790BA
K_2	=	16A4C6A9	F9DF3D3B	E4FC96EF	5309C1BD
		FB68E526	2CDBB534	FE161C83	6F7DD2C8

K_3	=	C49D846D	1780482C	9086887F	C48C9186
		9DCB0644	D1E641E5	A02109AF	9D52C7CF
K_4	=	BDB0C9F0	756E9131	E1F290EA	50E4CBB1
		1CAD9536	F4E4B674	99F31E29	70C52AFA
S	=	62A07EA5	EF3C3309	2CE1B076	173D48CC
		6881EB66	F5C7959F	63FCA1F1	D33C31B8
Ξ	=	95BEA0BE	88D5AA02	FE3C9D45	436CE821
		B8287CB6	2CBC135B	3E339EFE	F6576CA9

ШАГ 3

H	=	95BEA0BE	88D5AA02	FE3C9D45	436CE821
		B8287CB6	2CBC135B	3E339EFE	F6576CA9
L	=	00000000	00000000	00000000	00000000
		00000000	00000000	00000000	00000190
K_1	=	95FEB83E	BE3C2833	A09D7C9E	BE45B6FE

		88432CF6	D56CBC57	AAE8136D	02215B39
K_2	=	8695FEB8	1BBE3C28	E2A09D7C	48BE45B6
		DA88432C	EBD56CBC	7FABE813	F292215B
K_3	=	B9799501	141B413C	1EE2A062	0CB74145
		6FDA88BC	D0142A6C	FA80AA16	15F2FDB1
K_4	=	94B97995	7D141B41	C21EE2A0	040CB741
		346FDA88	46D0142A	BDFA81AA	DC1562FD
S	=	D42336E0	2A0A6998	6C65478A	3D08A1B9
		9FDDFF20	4808E863	94FD9D6D	F776A7AD
Ξ	=	47E26AFD	3E7278A1	7D473785	06140773
		A3D97E7E	A744CB43	08AA4C24	3352C745

ШАГ 4

H	=	47E26AFD	3E7278A1	7D473785	06140773
		A3D97E7E	A744CB43	08AA4C24	3352C745

Σ	=	73616820	65676173	73656D20	6061E1CE
		DBE2D48F	509A88B1	40CDE7D6	DED5E173
K_1	=	340E7848	83223B67	025AAAAB	DDA5F1F2
		5B6AF7ED	1575DE87	19E64326	D2BDF236
K_2	=	03DC0ED0	F4CD26BC	8B595F13	F5A4A55E
		A8B063CB	ED3D7325	6511662A	7963008D
K_3	=	C954EF19	D0779A68	ED37D3FB	7DA5ADDC
		4A9D0277	78EF765B	C4731191	7EBB21B1
K_4	=	6D12BC47	D9363D19	1E3C696F	28F2DC02
		F2137F37	64E4C18B	69CCFBF8	EF72B7E3
S	=	790DD7A1	066544EA	2829563C	3C39D781
		25EF9645	EE2C05DD	A5ECAD92	2511A4D1
Ξ	=	0852F562	3B89DD57	AEB4781F	E54DF14E
		EAFBC135	0613763A	0D770AA6	57BA1A47

Таким образом, результат хэширования есть

H	=	0852F562	3B89DD57	AEB4781F	E54DF14E
		EAFBC135	0613763A	0D770AA6	57BA1A47

Текст документа сверен по:
официальное издание
М.: Издательство стандартов, 1994