Hash criptogràfic SHA-512

Descripció general

El SHA-512 (secure hash algorithm, FIPS 180-4) és un hash criptogràfic de 512 bits. Es tracta d'una funció hash sense claus; o sigui, un MDC (Manipulation Detection Code).

Un missatge es processa per blocs de $1024 = 16 \times 64$ bits, 16 paraules de 64 bits.

Operacions bàsiques

- Les operacions booleanes AND, XOR i OR, que es denoten $\wedge$, $\oplus$ i $\vee$, respectivament.
- El pas al complementari, que es denota amb $\neg$.
- La suma d'enters mòdul 2^{64}, que es denota $A + B$.

Les operacions es fan sempre sobre paraules de 64 bits. Per a la darrera operació, cal interpretar aquestes paraules binàries com a nombres enters escrits en base 2.

- $RotR(A, n)$ indica que en la paraula binària A es fa un desplaçament cíclic de n bits a la dreta: $RotR(A, n) = (A >> n) \vee (A << w - n)$.
- $ShR(A, n)$ indica que en la paraula binària A es fa un desplaçament de n bits a la dreta.
- $A \| B$ indica la concatenació de paraules binàries.

Funcions i constants

L'algorisme fa servir les funcions:

$$\begin{aligned} Ch(X,Y,Z) &= (X \wedge Y) \oplus (\overline{X} \wedge Z), \\ Maj(X,Y,Z) &= (X \wedge Y) \oplus (X \wedge Z) \oplus (Y \wedge Z), \\ \Sigma_0(X) &= RotR(X,28) \oplus RotR(X,34) \oplus RotR(X,39), \\ \Sigma_1(X) &= RotR(X,14) \oplus RotR(X,18) \oplus RotR(X,41), \\ \sigma_0(X) &= RotR(X,1) \oplus RotR(X,8) \oplus ShR(X,7), \\ \sigma_1(X) &= RotR(X,19) \oplus RotR(X,61) \oplus ShR(X,6), \end{aligned}$$

i les 80 paraules binàries, K_i, donades pels 64 primers bits de la part fraccionària de les arrels cúbiques dels primers 80 nombres primers (llegides de esquerra a dreta):

0x428a2f98d728ae22	0x7137449123ef65cd	0xb5c0fbcfec4d3b2f	0xe9b5dba58189dbbc
0x3956c25bf348b538	0x59f111f1b605d019	0x923f82a4af194f9b	0xab1c5ed5da6d8118
0xd807aa98a3030242	0x12835b0145706fbe	0x243185be4ee4b28c	0x550c7dc3d5ffb4e2
0x72be5d74f27b896f	0x80deb1fe3b1696b1	0x9bdc06a725c71235	0xc19bf174cf692694
0xe49b69c19ef14ad2	0xefbe4786384f25e3	0x0fc19dc68b8cd5b5	0x240ca1cc77ac9c65
0x2de92c6f592b0275	0x4a7484aa6ea6e483	0x5cb0a9dcbd41fbd4	0x76f988da831153b5
0x983e5152ee66dfab	0xa831c66d2db43210	0xb00327c898fb213f	0xbf597fc7beef0ee4
0xc6e00bf33da88fc2	0xd5a79147930aa725	0x06ca6351e003826f	0x142929670a0e6e70
0x27b70a8546d22ffc	0x2e1b21385c26c926	0x4d2c6dfc5ac42aed	0x53380d139d95b3df
0x650a73548baf63de	0x766a0abb3c77b2a8	0x81c2c92e47edaee6	0x92722c851482353b
0xa2bfe8a14cf10364	0xa81a664bbc423001	0xc24b8b70d0f89791	0xc76c51a30654be30
0xd192e819d6ef5218	0xd69906245565a910	0xf40e35855771202a	0x106aa07032bbd1b8
0x19a4c116b8d2d0c8	0x1e376c085141ab53	0x2748774cdf8eeb99	0x34b0bcb5e19b48a8
0x391c0cb3c5c95a63	0x4ed8aa4ae3418acb	0x5b9cca4f7763e373	0x682e6ff3d6b2b8a3
0x748f82ee5defb2fc	0x78a5636f43172f60	0x84c87814a1f0ab72	0x8cc702081a6439ec
0x90befffa23631e28	0xa4506cebde82bde9	0xbef9a3f7b2c67915	0xc67178f2e372532b
0xca273eceea26619c	0xd186b8c721c0c207	0xeada7dd6cde0eb1e	0xf57d4f7fee6ed178
0x06f067aa72176fba	0x0a637dc5a2c898a6	0x113f9804bef90dae	0x1b710b35131c471b
0x28db77f523047d84	0x32caab7b40c72493	0x3c9ebe0a15c9bebc	0x431d67c49c100d4c
0x4cc5d4becb3e42b6	0x597f299cfc657e2a	0x5fcb6fab3ad6faec	0x6c44198c4a475817

Padding

Per aconseguir que el missatge que el tamany es pot representar tingui longitud múltiple de 1024 bits:

- primer s'afegeix un bit 1,
- després s'afegeixen k bits 0 essent k el menor enter positiu solució de $l + 1 + k \equiv 896 \mod 1024$ on l és la longitud del missatge original,
- la longitud del missatge original $l < 2^{128}$, es representa amb exactament 128 bits i s'omplen amb això les 128 darreres posicions.

El *padding* sempre es posa encara que l'entrada tingui longitud múltiple de 1024.

Descomposició d'un bloc en subblocs

Per a cada bloc $M \in \{0,1\}^{1024}$ es construeixen 80 paraules de 64 bits:

- les 16 primeres s'obtenen descomponent M en blocs de 64 bits

$$M = W_0 \| W_1 \| \cdots \| W_{14} \| W_{15}$$

- les restants s'obtenen aplicant la fórmula

$$W_i = \sigma_1(W_{i-2}) + W_{i-7} + \sigma_0(W_{i-15}) + W_{i-16}, \quad 16 \leq i \leq 79.$$

Calcul del hash

- Es comença inicialitzant vuit variables donades pels 64 primers bits de la part fraccionària de les arrels del primers 8 nombres primers:

$$\begin{array}{ll} H_1^{(0)} = \texttt{0x6a09e667f3bcc908} & H_2^{(0)} = \texttt{0xbb67ae8584caa73b} \\ H_3^{(0)} = \texttt{0x3c6ef372fe94f82b} & H_4^{(0)} = \texttt{0xa54ff53a5f1d36f1} \\ H_5^{(0)} = \texttt{0x510e527fade682d1} & H_6^{(0)} = \texttt{0x9b05688c2b3e6c1f} \\ H_7^{(0)} = \texttt{0x1f83d9abfb41bd6b} & H_8^{(0)} = \texttt{0x5be0cd19137e2179} \end{array}$$

- A continuació es processen els blocs de missatge $M^{(1)}$, $M^{(2)}, \ldots, M^{(N)}$, un per un:
  `Per` $t = 1$ `fins` N
  - a partir de $M^{(t)}$ es construeixen els 80 subblocs W_i tal com s'ha explicat anteriorment
  - amb els valors de les variables $H_j^{(t-1)}$ s'inicialitzen

$$(a, b, c, d, e, f, g, h) = (H_1^{(t-1)}, H_2^{(t-1)}, H_3^{(t-1)}, H_4^{(t-1)}, H_5^{(t-1)}, , H_6^{(t-1)}, H_7^{(t-1)}, H_8^{(t-1)})$$

  - es fan 80 voltes ($i = 0..79$), consistents en:

$$\begin{aligned}
T_1 &= h + \Sigma_1(e) + Ch(e, f, g) + K_i + W_i \\
T_2 &= \Sigma_0(a) + Maj(a, b, c) \\
h &= g \\
g &= f \\
f &= e \\
e &= d + T_1 \\
d &= c \\
c &= b \\
b &= a \\
a &= T_1 + T_2
\end{aligned}$$

  - es calculen les variables $H_j^{(t)}$

$$\begin{aligned}
H_1^{(t)} &= H_1^{(t-1)} + a \\
H_2^{(t)} &= H_2^{(t-1)} + b \\
H_3^{(t)} &= H_3^{(t-1)} + c \\
H_4^{(t)} &= H_4^{(t-1)} + d \\
H_5^{(t)} &= H_5^{(t-1)} + e \\
H_6^{(t)} &= H_6^{(t-1)} + f \\
H_7^{(t)} &= H_7^{(t-1)} + g \\
H_8^{(t)} &= H_8^{(t-1)} + h
\end{aligned}$$

  `Fi per`

- El hash del missatge és la concatenació del contingut de les vuit variables $H_i^{(N)}$ després d'haver processat el darrer bloc del missatge

$$H = H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)} \| H_5^{(N)} \| H_6^{(N)} \| H_7^{(N)} \| H_8^{(N)}.$$

Valors de test

Si implementeu el vostre propi SHA-512, els valors següents, donats en hexadecimal, permeten comprovar la implementació del hash.

entrada	`61 62 63` (“abc”)
hash	`ddaf35a193617aba cc417349ae204131 12e6fa4e89a97ea2 0a9eeee64b55d39a` `2192992a274fc1a8 36ba3c23a3feebbd 454d4423643ce80e 2a9ac94fa54ca49f`

entrada	`6162636465666768 6263646566676869 636465666768696a 6465666768696a6b`
	`65666768696a6b6c 666768696a6b6c6d 6768696a6b6c6d6e 68696a6b6c6d6e6f`
	`696a6b6c6d6e6f70 6a6b6c6d6e6f7071 6b6c6d6e6f707172 6c6d6e6f70717273`
	`6d6e6f7071727374 6e6f707172737475`
	(“abcdefghbcdefghicdefghijdefghijkefghijklfghijklmghijklmnhijklmnoijklmnop jklmnopqklm-nopqrlmnopqrsmnopqrstnopqrstu”).

hash	`8e959b75dae313da 8cf4f72814fc143f 8f7779c6eb9f7fa1 7299aeadb6889018` `501d289e4900f7e4 331b99dec4b5433a c7d329eeb6dd2654 5e96e55b874be909`
entrada	Un milió de `61` (“a...a”)
hash	`e718483d0ce76964 4e2e42c7bc15b463 8e1f98b13b204428 5632a803afa973eb` `de0ff244877ea60a 4cb0432ce577c31b eb009c5c2c49aa2e 4eadb217ad8cc09b`

Consulteu el apèndix C del FIPS 180-2.

Entrega

1. Trobeu una entrada M de 512 bits tal que si escriviu el seu hash H en hexadecimal els 8 primers símbols[1] coincideixin amb el vostre DNI. Per exemple, si el vostre DNI es 31416005 llavors la entrada

   ```
   41cb9c1e3b2e4390 b7b7d31940eac981 facb5e870db5e9aa 9958b256fd96164f...
      ba7224a912276a6f e42bd883d9a44fe3 3073a1c69756a97a 3b39e4db0aed3d80
   ```

   es una solució ja que té per hash

   ```
   3141600578cc8c6c 2637540ef0a608cd 6595a6666c9cefd7 db570720c1e8833f...
       4920b0b79cd17915 582b575524a4faca 404d0d6916316b1 a585c2f614d07eed9
   ```

2. Amb la M trobada feu una estadística dels bits que canvien a la sortida quan canvieu 1 bit a la entrada:

   (a) histograma del nombre total de bits que canvien amb cada modificació,

   (b) histograma de les posicions que canvien amb cada modificació.

   Feu el mateix si canvieu 2 bits, 3 bits...

Per llegir

Satoshi Nakamoto *Bitcoin: A Peer-to-Peer Electronic Cash System*

[1] Abans de començar, feu proves amb 1, 2, 3, 4... símbols i feu una estimació del temps que trigarà.