

## Hash criptogràfic SHA-256

### Descripció general

El SHA-256 (secure hash algorithm, FIPS 182-2) és un hash criptogràfic de 256 bits. Es tracta d'una funció hash sense claus; o sigui, un MDC (Manipulation Detection Code).

Un missatge es processa per blocs de  $512 = 16 \times 32$  bits. Per a cada bloc es fan 64 tombos.

### 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  $\bar{\phantom{x}}$ .
- La suma d'enters mòdul  $2^{32}$ , que es denota  $A + B$ .

Les operacions es fan sempre sobre paraules de 32 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.
- $ShR(A, n)$  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:

$$\begin{aligned}
 Ch(X, Y, Z) &= (X \wedge Y) \oplus (\bar{X} \wedge Z), \\
 Maj(X, Y, Z) &= (X \wedge Y) \oplus (X \wedge Z) \oplus (Y \wedge Z), \\
 \Sigma_0(X) &= RotR(X, 2) \oplus RotR(X, 13) \oplus RotR(X, 22), \\
 \Sigma_1(X) &= RotR(X, 6) \oplus RotR(X, 11) \oplus RotR(X, 25), \\
 \sigma_0(X) &= RotR(X, 7) \oplus RotR(X, 18) \oplus ShR(X, 3), \\
 \sigma_1(X) &= RotR(X, 17) \oplus RotR(X, 19) \oplus ShR(X, 10),
 \end{aligned}$$

i les 64 paraules binàries,  $K_i$ , donades pels 32 primers bits de la part fraccionària de les arrels cúbiques dels primers 64 nombres primers:

0x428a2f98	0x71374491	0xb5c0fbcf	0xe9b5dba5	0x3956c25b	0x59f111f1	0x923f82a4	0xab1c5ed5
0xd807aa98	0x12835b01	0x243185be	0x550c7dc3	0x72be5d74	0x80deb1fe	0x9bdc06a7	0xc19bf174
0xe49b69c1	0xefbe4786	0x0fc19dc6	0x240ca1cc	0x2de92c6f	0x4a7484aa	0x5cb0a9dc	0x76f988da
0x983e5152	0xa831c66d	0xb00327c8	0xbf597fc7	0xc6e00bf3	0xd5a79147	0x06ca6351	0x14292967
0x27b70a85	0x2e1b2138	0x4d2c6dfc	0x53380d13	0x650a7354	0x766a0abb	0x81c2c92e	0x92722c85
0xa2bfe8a1	0xa81a664b	0xc24b8b70	0xc76c51a3	0xd192e819	0xd6990624	0xf40e3585	0x106aa070
0x19a4c116	0x1e376c08	0x2748774c	0x34b0bcb5	0x391c0cb3	0x4ed8aa4a	0x5b9cca4f	0x682e6fff
0x748f82ee	0x78a5636f	0x84c87814	0x8cc70208	0x90beffff	0xa4506ceb	0xbef9a3f7	0xc67178f2

## Padding

Per aconseguir que el missatge<sup>1</sup> tingui longitud múltiple de 512 bits:

- primer s'afegeix un bit 1,
- després s'afegeixen  $k$  bits 0 essent  $k$  el menor enter positiu solució de  $l + 1 + k \equiv 448 \pmod{512}$  on  $l$  és la longitud del missatge original,
- la longitud del missatge original  $l < 2^{64}$ , es representa amb exactament 64 bits i s'omplen amb això les 64 darreres posicions.

El *padding* sempre es posa encara que l'entrada tingui longitud múltiple de 512.

## Descomposició d'un bloc en subblocs

Per a cada bloc  $M \in \{0, 1\}^{512}$  es construeixen 64 paraules de 32 bits:

- les 16 primeres s'obtenen descomponent  $M$  en blocs de 32 bits

$$M = W_1 \| W_2 \| \dots \| W_{15} \| W_{16}$$

- les restants s'obtenen aplicant la fórmula

$$W_i = \sigma_1(W_{i-2}) + W_{i-7} + \sigma_0(W_{i-15}) + W_{i-16}, \quad 17 \leq i \leq 64.$$

## Calcul del hash

- Es comença inicialitzant vuit variables donades pels 32 primers bits de la part fraccionària de les arrels dels primers 8 nombres primers:

$$\begin{aligned} H_1^{(0)} &= 0x6a09e667 & H_2^{(0)} &= 0xbbb67ae85 & H_3^{(0)} &= 0x3c6ef372 & H_4^{(0)} &= 0xa54ff53a \\ H_5^{(0)} &= 0x510e527f & H_6^{(0)} &= 0x9b05688c & H_7^{(0)} &= 0x1f83d9ab & H_8^{(0)} &= 0x5be0cd19 \end{aligned}$$

- A continuació es processen els blocs de missatge  $M^{(1)}, M^{(2)}, \dots, M^{(N)}$ , un per un:

Per  $t = 1$  fins  $N$

- a partir de  $M^{(t)}$  es construeixen els 64 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 64 voltes, consistents en:

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

---

<sup>1</sup>Suposarem que el tamany es pot representar amb un enter de 64 bits.

– 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

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

## Implementació: signatures

Implementeu el hash criptogràfic descrit. Definiu la classe `sha256` amb el mètode:

```
public static BigInteger hash(byte[] M)
```

entrada: `M` és una llista de bytes de longitud arbitrària;

sortida: un enter positiu de l'interval  $[0, 2^{256})$  que és el hash que representa el missatge `M`.

## Valors de test

Els valors següents, donats en hexadecimal, permeten comprovar la implementació del hash.

entrada	61 62 63
hash	ba7816bf 8f01cfea 414140de 5dae2223 b00361a3 96177a9c b410ff61 f20015ad
entrada	61 62 63 64 62 63 64 65 63 64 65 66 64 65 66 67 65 66 67 68 66 67 68 69 67 68 69 6a 68 69 6a 6b 69 6a 6b 6c 6a 6b 6c 6d 6b 6c 6d 6e 6c 6d 6e 6f 6d 6e 6f 70 6e 6f 70 71
hash	248d6a61 d20638b8 e5c02693 0c3e6039 a33ce459 64ff2167 f6ecedd4 19db06c1
entrada	Un milió de 61
hash	cdc76e5c 9914fb92 81a1c7e2 84d73e67 f1809a48 a497200e 046d39cc c7112cd0