

Criptografia FIB

11. Funcions Hash

Anna Rio

Departament de Matemàtica Aplicada II • Universitat Politècnica de Catalunya



FUNCIONS HASH

- Integritat de missatges (què?)
- Signatura digital (qui?)
- Timestamping (quan?)

*Una propietat interessant del segells de temps digitals és que **el document a segellar no cal entregar-lo a ningú** per tal d'obtenir el segell de temps. El creador del document calcula ell mateix el **valor de hash** i l'envia al servei de segellat. El document en sí només es necessita quan es vol verificar el segell de temps. Això resulta molt útil en diverses situacions (com per exemple protegir **quelcom que es vol patentar**).*



*It sometimes happens that **the connection between a person and his or her public signature key must be revoked**. For example, the user's private key may accidentally be compromised, or the key may belong to a job or role in an organization that the person no longer holds. Therefore the person-key connection must have time limits, and **the signature verification procedure should check that the record was signed at a time when the signer's public key was indeed in effect**. And thus when a user signs a record that may be checked some time later - perhaps after the user's key is no longer in effect - the combination of the record and its signature should be certified with a secure digital timestamping service.*



*There is another situation in which a user's public key may be revoked. Consider **the case of the signer of a particularly important document who later wishes to repudiate his signature.** By dishonestly reporting the compromise of his private key, so that all his signatures are called into question, the user is able to disavow the signature he regrets. However, if the document in question was **digitally timestamped together with its signature** (and key-revocation reports are timestamped as well), then the signature cannot be disavowed in this way. This is the **recommended procedure, therefore, in order to preserve the non-reputability desired of digital signatures for important documents.***



Funcions hash

$$H : \mathcal{M} \longrightarrow I$$

funció matemàtica que proporciona un **resum** del missatge
(**message digest**)

En general, la sortida ha d'esser quelcom de longitud més curta que l'entrada

$\mathcal{M} = \{0, 1\}^*$ missatges = cadenes de bits de longitud arbitrària finita

$\mathcal{M} = \mathbb{N}$ missatges = nombres naturals

1011101111000111001010010110011101 = 12601566621



$$H(m) = m \pmod{2^{256}}$$

- $I = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, \dots, 2^{256} - 1\}$

11579208923731619542357098500868790785326

9984665640564039457584007913129639936

valors de hash possibles

- la probabilitat que $H(m) = H(m')$ és $1/2^{256} \approx 8.6 \times 10^{-78}$
- donat m , trobem fàcilment missatges m' amb el mateix hash:
 $m' = m + 2^{256} t, \quad t \in \mathbb{N}$



Funcions hash

$$H(m) = m \pmod{2^{256}}$$

- $I = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, \dots, 2^{256} - 1\}$

11579208923731619542357098500868790785326

9984665640564039457584007913129639936

valors de hash possibles

- la probabilitat que $H(m) = H(m')$ és $1/2^{256} \approx 8.6 \times 10^{-78}$
- donat m , trobem fàcilment missatges m' amb el mateix hash:
 $m' = m + 2^{256} t, \quad t \in \mathbb{N}$



$$H(m) = m \pmod{2^{256}}$$

- $I = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, \dots, 2^{256} - 1\}$

11579208923731619542357098500868790785326

9984665640564039457584007913129639936

valors de hash possibles

- la probabilitat que $H(m) = H(m')$ és $1/2^{256} \approx 8.6 \times 10^{-78}$
- donat m , trobem fàcilment missatges m' amb el mateix hash:
 $m' = m + 2^{256} t, \quad t \in \mathbb{N}$

- 1 Compressió
- 2 Unidireccionalitat
- 3 Difusió
- 4 Feblement lliure de col.lisions
- 5 Fortament lliure de col.lisions

PROPIETAT 1: Compressió

Normalment es demana que els $H(m)$ tinguin tots **una longitud fixada**

Es pot pensar, doncs, que les funcions hash fan una compressió del missatge en un únic bloc, d'una determinada longitud

$I = [0, 2^{256} - 1]$ (enters positius de 256 bits) és un conjunt de

$$2^{256} = 1157920892373161954235709850086879078532 \\ 69984665640564039457584007913129639936$$

elements. Si la funció H **equiparteix** els elements de \mathcal{M} , la probabilitat que dos missatges tinguin el mateix hash és

$$\frac{1}{2^{256}} \approx 8.6 \times 10^{-78}$$

PROPIETAT 1: Compressió

Normalment es demana que els $H(m)$ tinguin tots **una longitud fixada**

Es pot pensar, doncs, que les funcions hash fan una compressió del missatge en un únic bloc, d'una determinada longitud

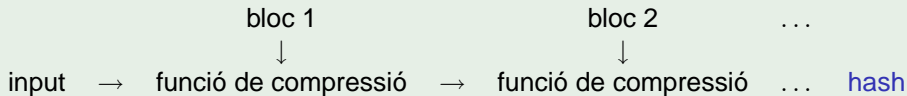
$I = [0, 2^{256} - 1]$ (enters positius de 256 bits) és un conjunt de

$$2^{256} = 1157920892373161954235709850086879078532 \\ 69984665640564039457584007913129639936$$

elements. Si la funció H **equiparteix** els elements de \mathcal{M} , la probabilitat que dos missatges tinguin el mateix hash és

$$\frac{1}{2^{256}} \approx 8.6 \times 10^{-78}$$

Estructura iterativa de les funcions hash



Exemple: SHA-256

PROPIETAT 2: Unidireccionalitat

- Donat un text, és computacionalment fàcil/ràpid calcular el hash
- Donat un hash, és computacionalment impossible reconstruir el text

Així doncs, es busquen funcions $H : \mathcal{M} \longrightarrow I$ **unidireccionals**



El resum $H(m)$ ha d'ésser una **funció complexa** de tots els bits del missatge.

*With good diffusion, flipping an input bit should change each output bit with a probability of one half (**avalanche effect**)*

In the case where the hash function lacks strong diffusion capabilities, an adversary could create collisions very easily since he could predict the response of the hash function to alterations in the input.



PROPIETATS 4 i 5: Lliure de col.lisions

Feblement

Donat m , és computacionalment impossible trobar $m' \neq m$ tal que $H(m') = H(m)$

Fortament

És computacionalment impossible trobar un parell m i m' tal que $H(m) = H(m')$

Fortament lliure de colisions \Rightarrow unidireccional.



Atac usant la paradoxa de l'aniversari

En un grup de 23 persones, la probabilitat que dues facin anys el mateix dia és $> 1/2$

Si una funció pren k valors equiprobables, aleshores s'espera obtenir una col·lisió després d'aproximadament $1.2\sqrt{k}$ avaluacions

$$1.2\sqrt{365} = 22.926$$

Si usem un hash de 160 bits, per tenir probabilitat superior al 50% de trobar dos missatges amb el mateix resum $H(m)$ no caldrà buscar entre 2^{160} elements sinó entre

$$2^{80} = 1208925819614629174706176$$

Amb un hash de 256 bits, caldria fer 2^{128} proves

Atac usant la paradoxa de l'aniversari

En un grup de 23 persones, la probabilitat que dues facin anys el mateix dia és $> 1/2$

Si una funció pren k valors equiprobables, aleshores s'espera obtenir una col·lisió després d'aproximadament $1.2\sqrt{k}$ avaluacions

$$1.2\sqrt{365} = 22.926$$

Si usem un **hash de 160 bits**, per tenir probabilitat superior al 50% de trobar dos missatges amb el mateix resum $H(m)$ no caldrà buscar entre 2^{160} elements sinó entre

$$2^{80} = 1208925819614629174706176$$

Amb un **hash de 256 bits**, caldria fer 2^{128} proves

Algoritmes hash

- MD2** (Message Digest 2) Ron Rivest, 1989. Dissenyat per a processadors de 8 bits. Encara s'usa.
- MD4** (Message Digest 4) Ron Rivest, 1990. Per a processadors de 32-bits. Dobbertin: amb un PC es poden trobar col.lisions en menys d'un minut. Es considera trencat.
- MD5** Ron Rivest 1991. Milliores al MD4 i MD2. Tots tres fan hash de 128 bits. S'ha utilitzat en SSL i PGP. Dobbertin (1996): va crear col.lisions i obtenir missatges amb el mateix hash que un donat.
- RIPEND** Dobbertin et al., Projecte de la Comunitat Europea, 1992. Resum de 160 bits.
- Panama** John Daemen, Craig Clapp, 1998. Resums de 256 bits. Funció hash o algoritme de xifrat en flux.



- RFC 1321 - The MD5 Message-Digest Algorithm
<http://www.faqs.org/rfcs/rfc1321.html>
- H. Dobbertin, *Cryptanalysis of MD5 Compress* Eurocrypt, 1996

SHA: Secure Hash Algorithm

- NIST, 1994 (FIPS 180-1) Similar a MD5 però amb hash de 160 bits.
- Agost 2002: FIPS 180-2 (vigent des de Febrer 2003)

SHA-1 i SHA-256

- missatges de longitud menor que 2^{64}
- blocs de 512 bits, paraules de 32 bits
- longitud del hash: 160 i 256, respectivament

SHA-384 i SHA-512

- missatges de longitud menor que 2^{128}
- blocs de 1024 bits, paraules de 64 bits
- longitud del hash: 384 i 512, respectivament

SHA: Secure Hash Algorithm

- X. Wang, Y. Yin, H. Yu **Finding Collisions in the Full SHA-1** Crypto'05 (14–18 Agost, Santa Barbara, California, USA)
- B. Schneier, *SHA-1 Broken*, Crypto-Gram Newsletter (15 Març)

On Tuesday Evening(8/16/2005), it was announced that it is possible to find a collision in SHA-1 **in 2^{63} operations**. This research result is due to Professor Xiaoyun Wang of Tsinghua University in Beijing, together with Professors Andrew Yao and Frances Yao. It extends the work of Wang, Yin, and Yu, which demonstrated that a collision could be found **in 2^{69} operations**. The new result was announced by Adi Shamir at the Crypto 2005 conference in Santa Barbara, on behalf of Professor Wang, who was unable to secure a visa for the conference.



Status of the Attack

This work estimates the difficulty of an attack, rather than producing an actual collision. **No actual collision for SHA-1 has been exhibited to date.** However 2^{63} is within reach of a distributed computing effort. It will not be surprising if further improvements to SHA-1 collision attacks appear in the coming months.

Practical Ramifications

This research has ramifications for applications which require collision resistant hash functions: for example digital signatures. Practically, this cryptanalytic result suggests the **acceleration of upgrading software** which uses hash functions.

Three viable approaches for improving the security of applications are:

- **Replace the hash function with a stronger one.** The most commonly suggested approach is to simply employ SHA-256, possibly truncating the output to 160 bits.
- **Alter the protocol** so that it no longer requires that the hash function be collision resistant. A recent proposal suggests **adding randomness** to hash functions. To implement this, the application must have a good source of randomness and must alter the protocol.
- **Implement simple message pre-processing** to convert plaintext messages into a form which renders all existing collision attacks inapplicable.

Bruce Schneier (Febrer 2005)

I'd like to see NIST orchestrate a **worldwide competition for a new hash function**, like they did for the new encryption algorithm, AES, to replace DES. NIST should issue a call for algorithms, and conduct a series of analysis rounds, where the community analyzes the various proposals with the intent of establishing a new standard.

Most of the hash functions we have, and all the ones in widespread use, are based on the general principles of MD4. Clearly we've learned a lot about hash functions in the past decade, and I think we can start applying that knowledge to **create something even more secure**.

