Funciones de hash criptográficas

Función de un espacio de posibles mensajes a un espacio de mensajes de largo fijo:

$$h : \mathcal{M} \to \mathcal{H}$$

$\mathcal{M}$ es el espacio de mensajes y $\mathcal{H}$ es el espacio de posibles valores de la función de hash

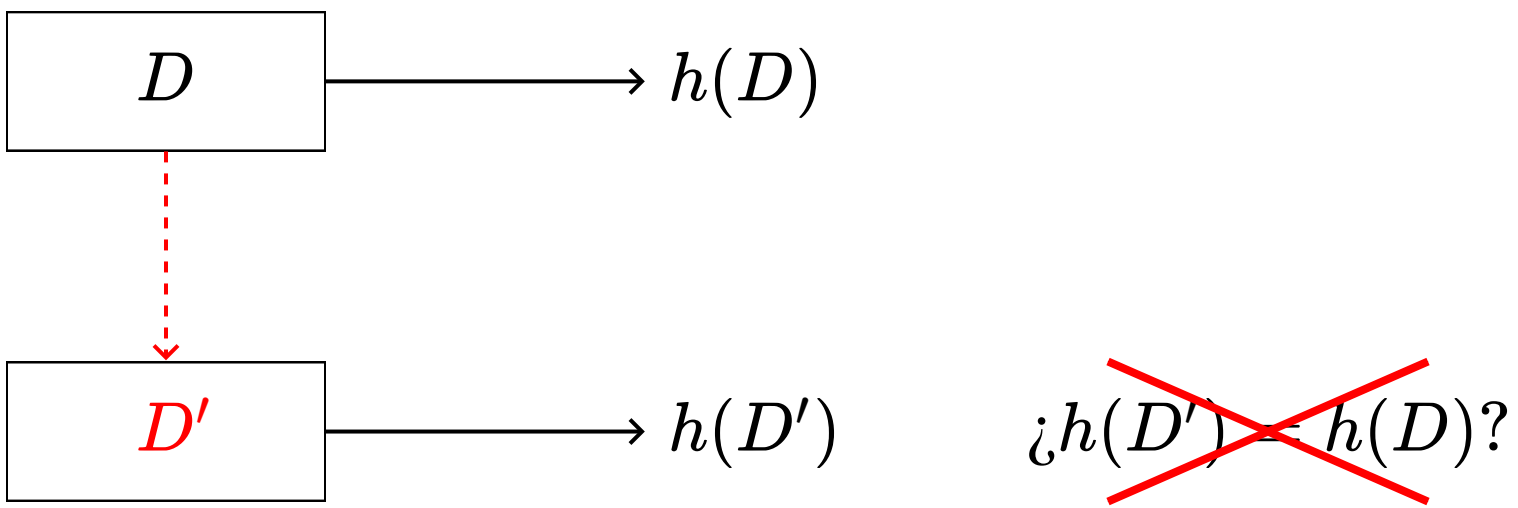
- Por ejemplo, $\mathcal{M} = \{0,1\}^*$ y $\mathcal{H} = \{0,1\}^{128}$
- Decimos que $h(m)$ es el hash de un mensaje m

Una primera propiedad fundamental de las funciones de hash

- Debe existir un algoritmo eficiente que, dado $m \in \mathcal{M}$, calcula $h(m)$
- No debe existir un algoritmo eficiente que, dado $x \in \mathcal{H}$, encuentra $m \in \mathcal{M}$ tal que $h(m) = x$

Esta propiedad se denota como ser resistente a preimagen

Una primera aplicación: integridad de un documento



¿Por qué insistimos en el adjetivo "criptográficas"?

Considere la siguiente función de hash:

$$h(m) \ = \ (A \cdot m + B) \mod C$$

Suponemos que los mensajes son números naturales

- A, B y C son constantes, C es un número primo

¿Es esta función resistente a preimagen?

La función anterior no es resistente a preimagen

Considere $h(m) = (13 \cdot m + 97) \bmod 641$

Suponga que tiene el valor de hash 200. ¿Puede encontrar un mensaje m tal que $h(m) = 200$?

Una combinación de herramientas de aritmética modular nos pueden dar una respuesta rápida: 15244

$$h(15244) = (13 \cdot 15344 + 97) \bmod 641 = 200$$

Una segunda propiedad fundamental de las funciones de hash

No debe existir un algoritmo eficiente que pueda encontrar $m_1, m_2 \in \mathcal{M}$ tales que $m_1 \neq m_2$ y $h(m_1) = h(m_2)$

Esta propiedad se denota como ser resistente a colisiones

La formalización de esta idea necesita de un poco de trabajo ...

Pero antes veamos a las funciones de hash en la práctica

```python
import hashlib

if __name__ == "__main__":
    h1 = hashlib.md5()
    h2 = hashlib.md5()
    h3 = hashlib.md5()

    h1.update(b"este es mi primer mensaje")
    h2.update(b"El objetivo del curso es introducir al alumno a "
              b"los conceptos fundamentales de criptografia y "
              b"seguridad computacional, poniendo enfasis tanto en "
              b"los aspectos formales necesarios para definir")
    h3.update(b"este es mi prXmer mensaje")

    print(h1.hexdigest())
    print(h2.hexdigest())
    print(h3.hexdigest())
```

Output: 30635f74755bfb8c9faeac3ab106c2ab

c0e1fe34f764e458463c4fd8a91355d0

2c5956c357577eed8e76608cf40e79ee

Pero antes veamos a las funciones de hash en la práctica

```python
import hashlib

if __name__ == "__main__":
    h1 = hashlib.sha256()
    h2 = hashlib.sha256()
    h3 = hashlib.sha256()

    h1.update(b"este es mi primer mensaje")
    h2.update(b"El objetivo del curso es introducir al alumno a "
              b"los conceptos fundamentales de criptografia y "
              b"seguridad computacional, poniendo enfasis tanto en "
              b"los aspectos formales necesarios para definir")
    h3.update(b"este es mi prXmer mensaje")

    print(h1.hexdigest())
    print(h2.hexdigest())
    print(h3.hexdigest())
```

Output: 105f0a373501caffc828ce3da6b0b9c7569c68194f1db1057831fa8b3844cc8c

5a022e6e371bf654c33025642eb147a432f26dc3c3206ec992fc7725799c3868

5d94508d4fb9a1daeade09995904a281ccbdd165d2dc7a24798fcff9a80c96e7

Una definición formal de función de hash

Una función de hash es un par (Gen, h) de algoritmos de tiempo polinomial tales que dado un parámetro de seguridad 1^n:

- Gen toma como entrada 1^n y retorna una llave s
- h toma como entrada s y un mensaje $m \in \{0,1\}^*$, y retorna un hash $h^s(m) \in \{0,1\}^{\ell(n)}$, donde ℓ es un polinomio fijo

Una definición formal de función de hash

Si $m \in \{0,1\}^{\ell'(n)}$ para un polinomio fijo ℓ' tal que $\ell'(n) > \ell(n)$, entonces (Gen, h) es una función de hash de largo fijo

En una definición más general pedimos que (Gen, h) sea un par de algoritmos aleatorizados de tiempo polinomial

Una noción necesaria

Una función $f : \mathbb{N} \to \mathbb{N}$ es despreciable si:

$$(\forall \text{ polinomio } p)(\exists n_0 \in \mathbb{N})(\forall n \geq n_0)\left(f(n) < \frac{1}{p(n)}\right)$$

1. Muestre que 2^{-n} y $n^{-\log(n)}$ son funciones despreciables
2. Demuestre que si f y g son funciones despreciables y p es un polinomio, entonces $f + g$ y $f \cdot p$ son funciones despreciables