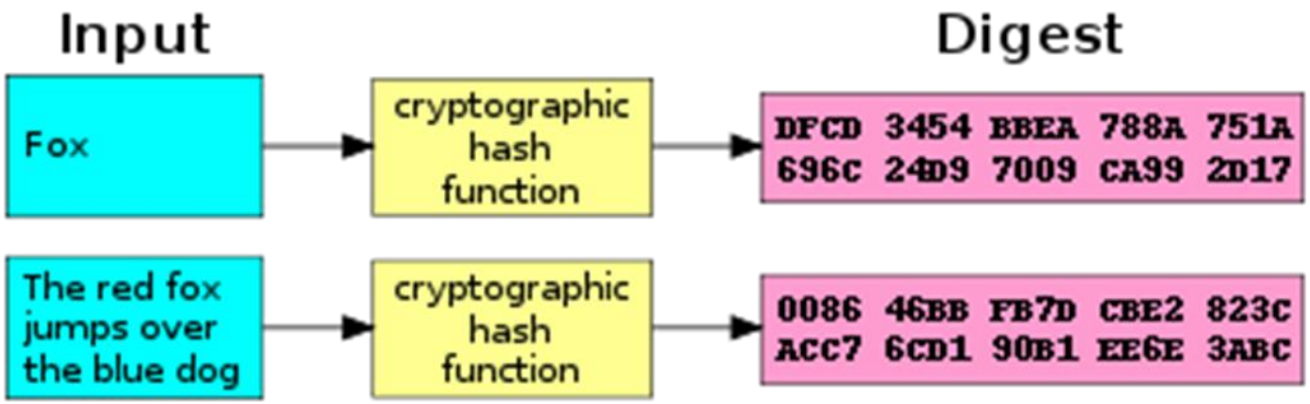
Seguridad de la Información

Introducción a Pycryptodome

Hash y HMAC

Recordatorio: Hash

- Una función hash criptográfica es un algoritmo h = hash(M) para el que resulta computacionalmente imposible:
  1. encontrar la preimagen *M* para un valor hash *h* predeterminado (propiedad: **unidireccionalidad**), y
  2. encontrar dos bloques *M* y *M'* que produzcan el mismo valor hash *h* (propiedad: **libre de colisiones**), tal que *h = h'*



Hash en Pycryptodome (SHA256)

- En el paquete **Crypto.Hash**

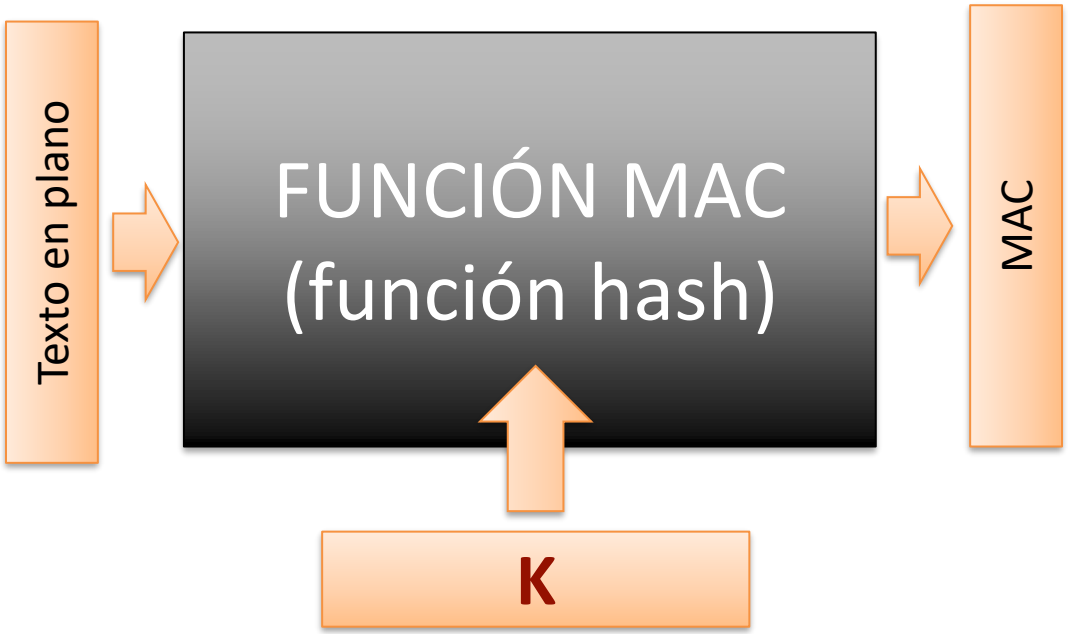
  - Crypto.Hash.SHA256.new(data=b'First')

  - Parámetros:
    - ***data***: Datos sobre los que hacer el hash
  - Devuelve: un objeto **SHA256Hash**

- Es posible incluir nuevos datos dentro del hash con el método update(data)
- El valor hash se obtiene de dos formas:
  - .digest() => hash como array de bytes
  - .hexdigest() => hash como cadena de caracteres hexadecimales

Recordatorio: HMAC

- Un código de autenticación de mensaje, o función MAC, toma como entrada un mensaje *M* y una clave *K*, y produce un valor hash (que en este caso se denomina valor **MAC**)



HMAC en Pycryptodome (SHA256)

- En el paquete **Crypto.Hash**
  - Crypto.Hash.HMAC.new(secret, msg=b'First', digestmod=SHA256)
  - Parámetros:
    - ***secret***: clave a utilizar – ***¡¡¡distinta al cifrado!!!***
    - ***msg***: Datos sobre los que hacer el hash
    - ***digestmod***: función hash a utilizar
  - Devuelve: un objeto **HMAC**
- Es posible incluir nuevos datos dentro del HMAC con el método update(data)
- El valor HMAC se obtiene de dos formas:
  - .digest() => HMAC como array de bytes
  - .hexdigest() => HMAC como cadena de caracteres hexadecimales

HMAC en Pycryptodome (SHA256)

- Ejemplo de HMAC y verificación

```
from Crypto.Hash import HMAC, SHA256
from Crypto.Random import get_random_bytes

secret = get_random_bytes(16)
hsend = HMAC.new(secret, msg=b'First', digestmod=SHA256)
mac = hsend.digest()

hrecv = HMAC.new(secret, digestmod=SHA256)
hrecv.update(b'First')
try:
  hrecv.hexverify(mac)
  print("The message is authentic")
except ValueError:
  print("The message or the key is wrong")
```

Bibliografía básica

- "Python 3 documentation"

  https://docs.python.org/3/tutorial/

- PyCryptodome

  https://pycryptodome.readthedocs.io/en/latest/src/hash/hash.html