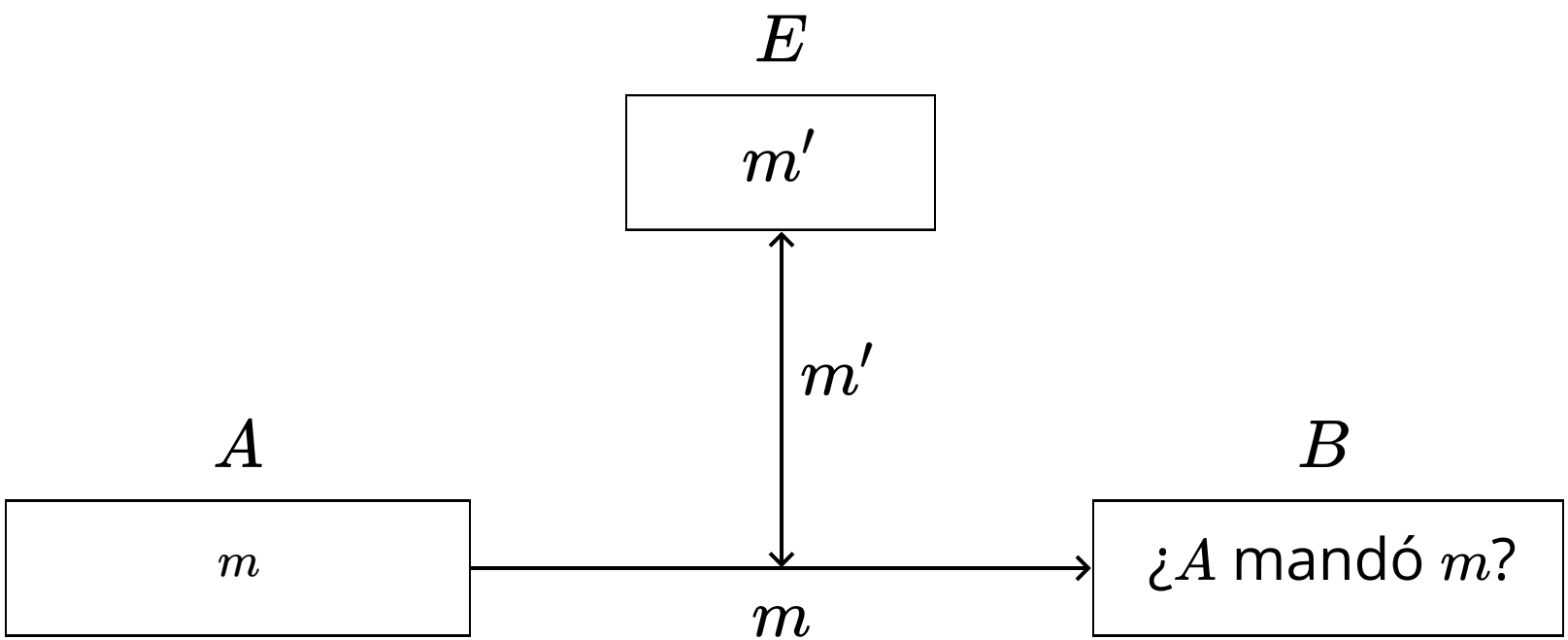
IIC3253

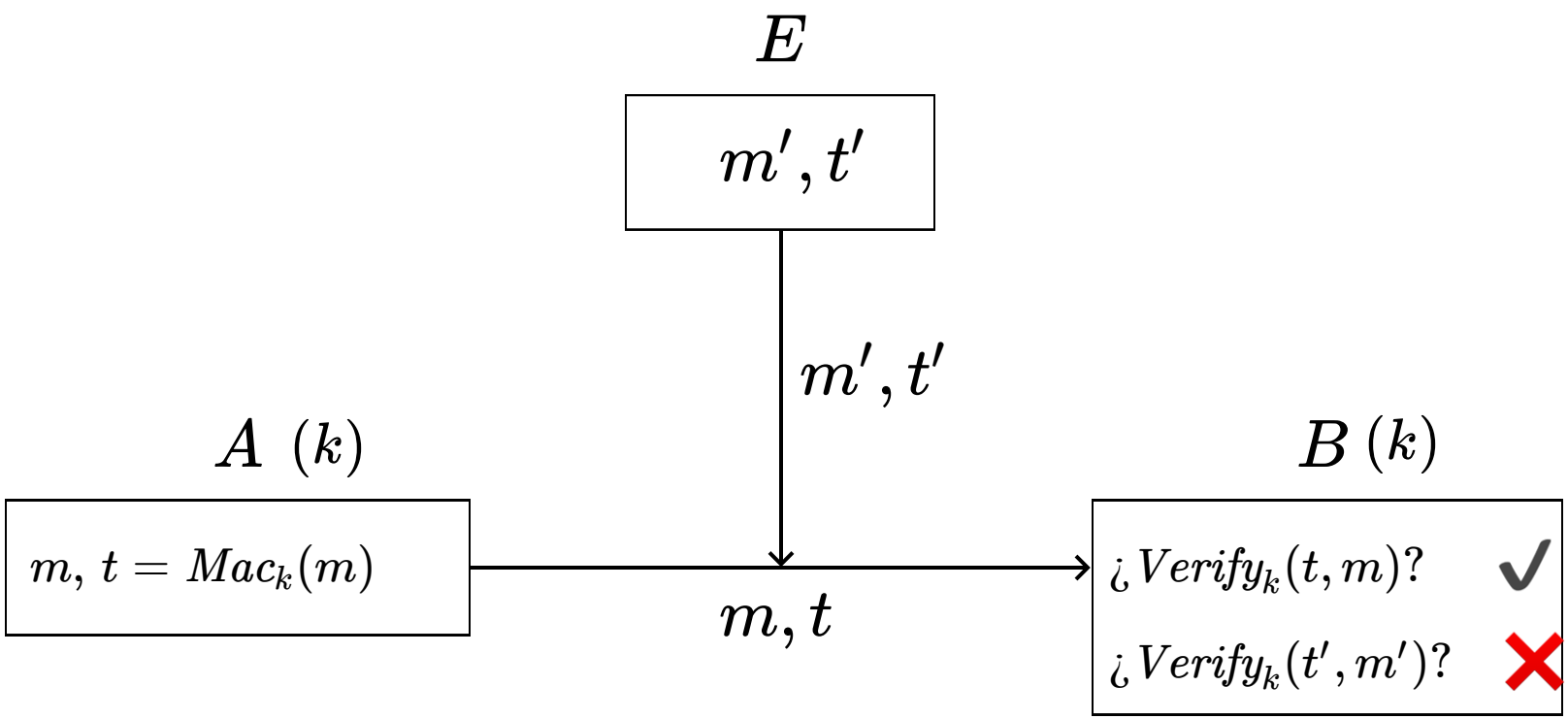
MACs

(≠ Media Access Control, como en MAC address)

Autentificación de mensajes



¿Cómo lo hacemos?

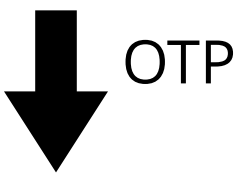


¿Qué relación tiene la encriptación
con la autentificación de mensajes?

¿No basta con encriptar?


WRONG

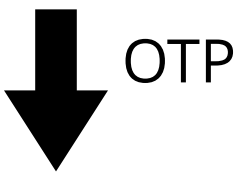
```
{
   from_account: 239478456,
   to_account: 579821324,
   amount: 10000,
   to_name: "Marcelo Arenas",
   to_bank: "BancoMarcelo",
   pinPass: 765432
}
```



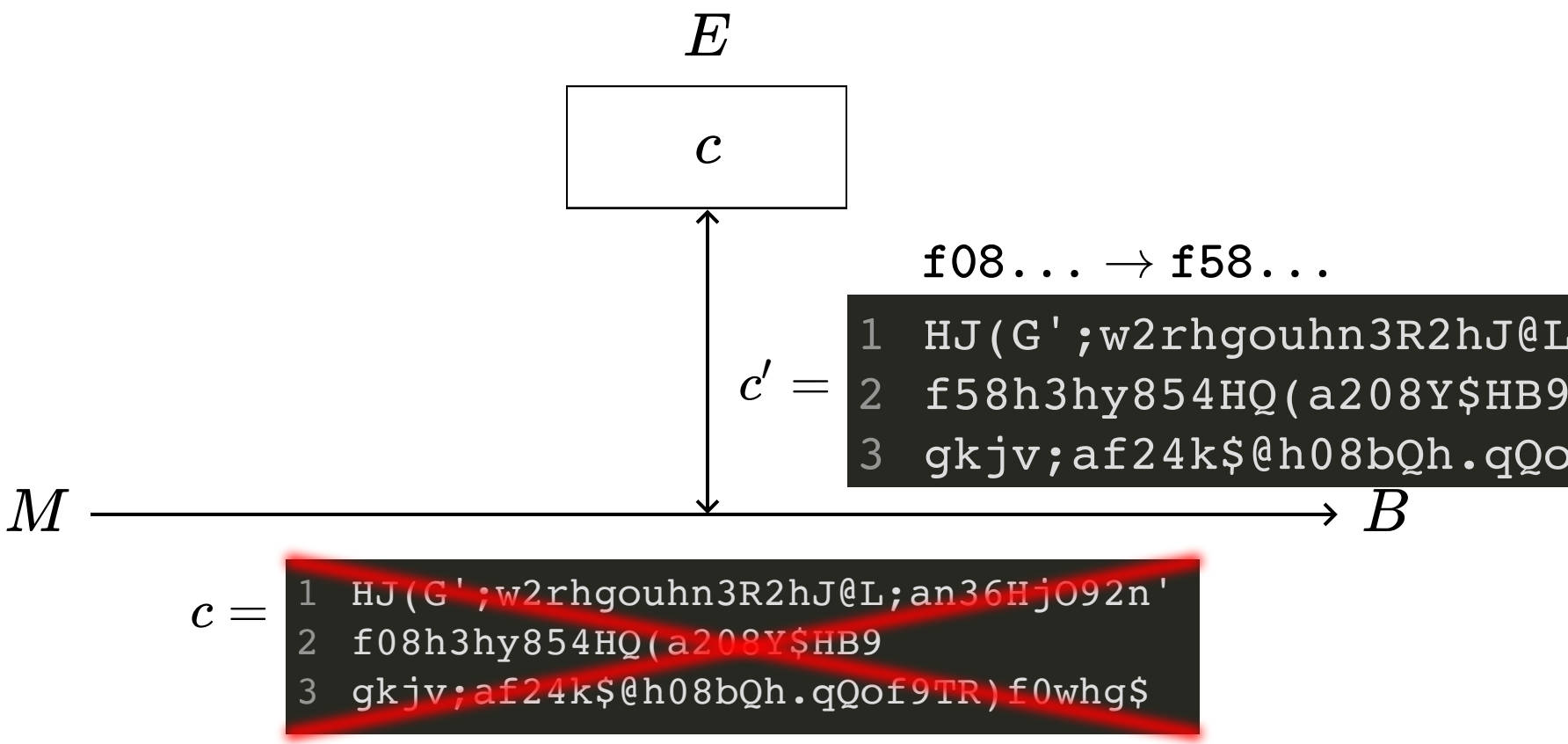

```
asdgh09o8hGH3jkG$q325hyghRG
08hyg;no3i5hghwaser
;'hqr0eihnj5asfdg08hq0[q3rw08gh00qasdfglkh
4oih,@h9hu0hb
```

El monto está encriptado a partir del caracter número 62...

```
{
  from_account: 579821324,
  to_account: 239478456,
  amount: 40000,
  to_name: "Martín Ugarte",
  to_bank: "BancoMartín",
  pinPass: 785613
}
```



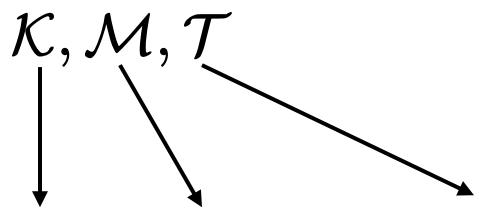

```
HJ(G';w2rhgouhn3R2hJ@L;an36HjO92n'
f08h3hy854HQ(a208Y$HB9
gkjv;af24k$@h08bQh.qQof9TR)f0whg$
```



A.K.A. Bit flip attack

A menos que un esquema criptográfico esté diseñado para ser autentificado, jamás debemos suponer que lo es.

Formalizando...



Espacio de llaves, mensajes y tags

Un esquema de autentificación de mensajes
es una tupla $(Gen, Mac, Verify)$ tal que:

Gen es un algoritmo aleatorizado tal que, dado 1^n,
genera una llave en $\mathcal{K}$ de largo mayor o igual a n

Mac es una familia de funciones $Mac_k : \mathcal{M} \to \mathcal{T}$

$Verify$ es una familia de algoritmos $Verify_k : \mathcal{T} \times \mathcal{M} \to \{0, 1\}$

¿Qué esperamos?

Para todo $k \in \mathcal{K}$ y $m \in \mathcal{M}$, se cumple que

$$Verify_k(Mac_k(m), m) = 1$$

y dado $m \in \mathcal{M}$ con $m' \neq m$ tenemos

$$Verify_k(Mac_k(m'), m) = 0$$

?

A jugar...

$(Gen, Mac, Verify)$

Definimos el juego $Forge_{MAC}(n)$:

1. El verificador invoca $Gen(1^n)$ para generar k
2. El adversario envía $m_0 \in \mathcal{M}$
3. El verificador responde $Mac_k(m_0)$
4. Los pasos 2 y 3 se repiten tantas veces como quiera el adversario
5. El adversario envía (m, t), siendo m un mensaje que no había enviado antes

¿Cuándo gana el adversario?

$$Verify_k(t, m) = 1$$

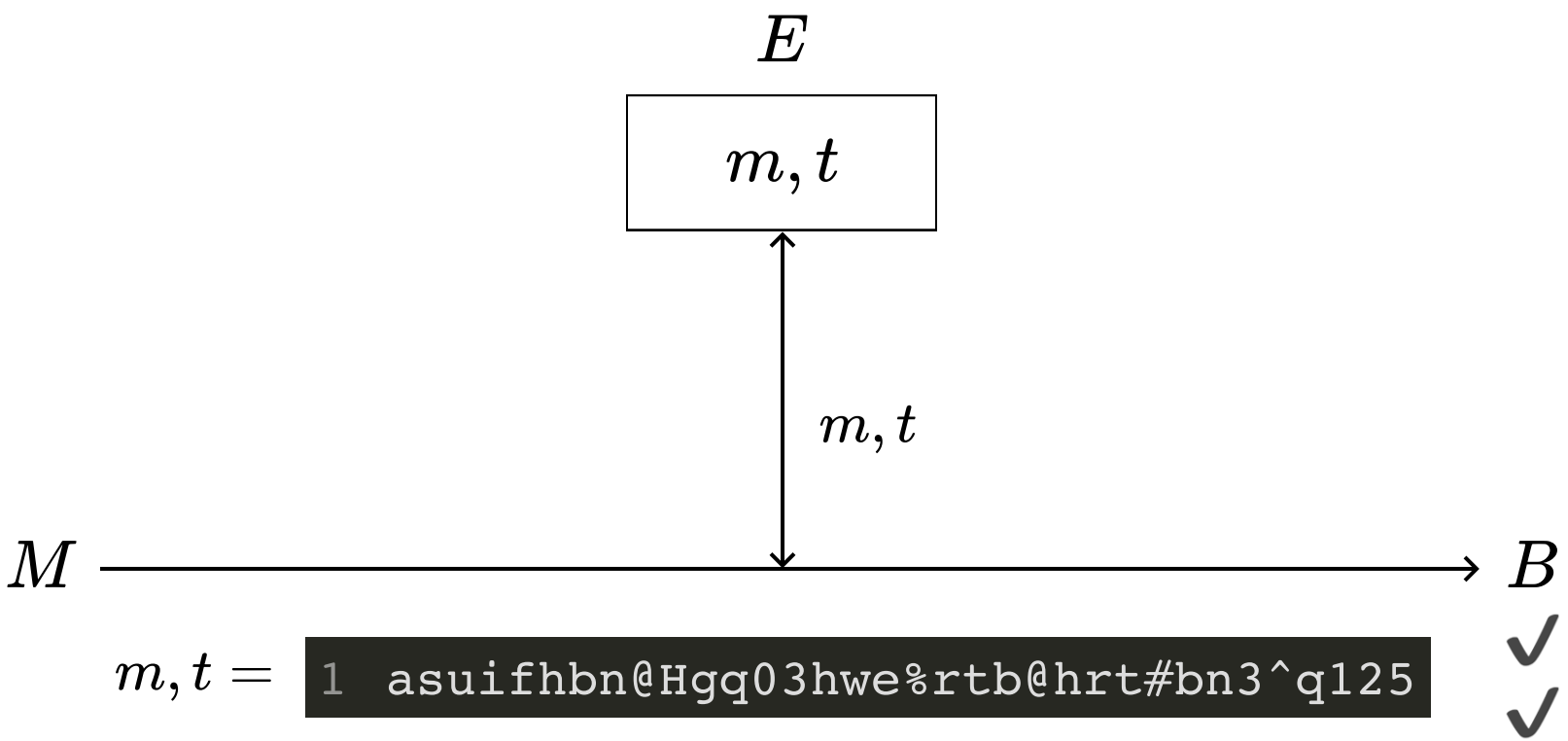
¿Cuándo decimos que MAC es un esquema de autentificación de mensajes **seguro?**

Todo adversario que juega en tiempo polinomial (en n) tiene una probabilidad despreciable (en n) de ganar el juego

En general se usa *seguro* = *unforgeable*

¿Algún ataque más?

```
{
  from_account: 579821324,
  to_account: 239478456,
  amount: 40000,
  to_name: "Martín Ugarte",
  to_bank: "BancoMartín",
  pinPass: 785613
}
```



A.K.A. replay attack

Un esquema de autentificación de mensajes **no tiene por qué** mitigar este tipo de ataques. En general esto se maneja a nivel de aplicación.

¿Podemos construir un MAC en base a una función de hash?

$$¿Mac_k(m) = h(k\|m)?$$

¿Dado $h(k\|m)$, estamos seguros de que nadie puede calcular $h(k\|m\|m')$?

Ataques de extensión