# AUTENTICAZIONE

Gabriele Fillari

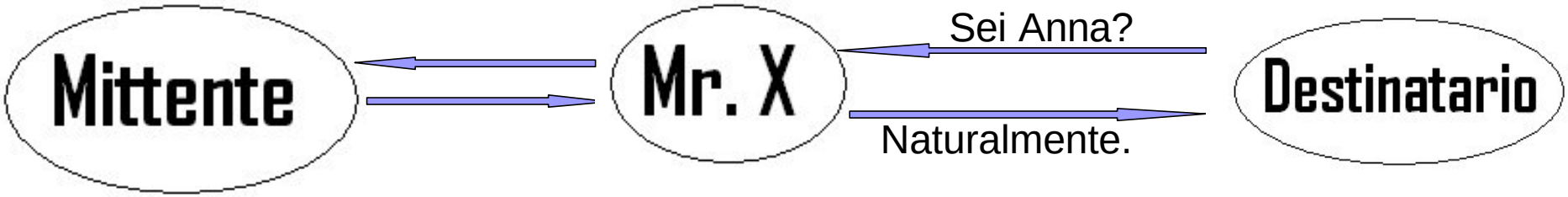
# 3 problemi di base:

- 1)  integrità del messaggio
- 2) sicurezza sull’ autenticità del mittente
- 3) identità del mittente

# 1) integrità del messaggio

# 2) Sicurezza sull' autenticità del mittente

# 3) identità del mittente



# Come possiamo ottenere integrità e autenticità?

- Il destinatario ha bisogno di uno strumento di verifica.
  - Un' informazione aggiuntiva (MAC)
    - Viene calcolato grazie a un algoritmo crittografico applicato sul messaggio **m** con una chiave **k** conosciuta da entrambe le parti.
    - Mr. X è contrastato efficacemente poiché non è in possesso della chiave k.
    - Il destinatario può solamente rendersi conto della presenza di una modifica ma non può ritornare al messaggio originale.

# MAC

$mac=f_k(m)$


$mac^* = f_k(m')$

$mac^* = mac'$ ??

# Esempio:

m = i love you

k = nazareno  (13,0,25,0,17,4,13,14)

Mac? Usiamo un cifrario di Vigenere.

mac= VLNUVCBI

Si inserisce mr. X nella comunicazione e modifica il messaggio.

m'= i hate you  mac'=VLNUVCBI

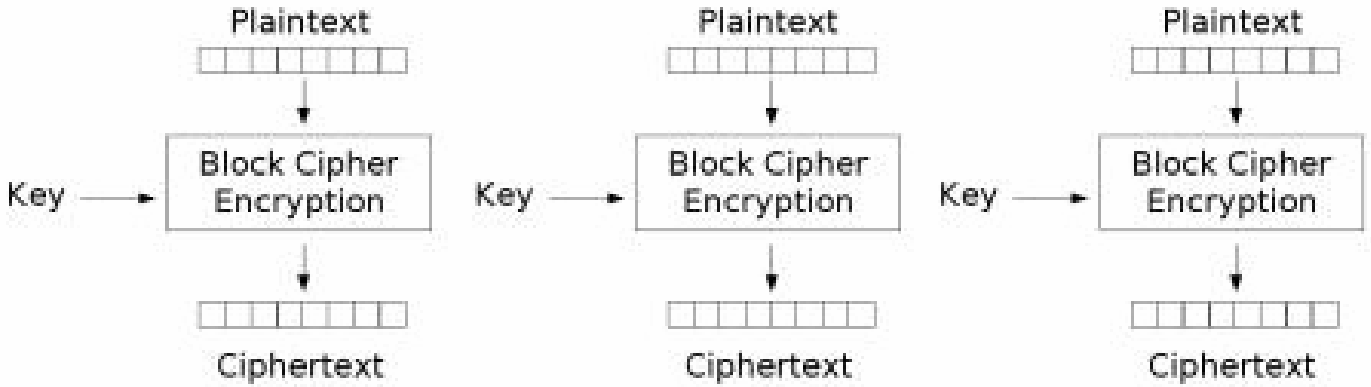
A questo punto il destinatario ricalcolerà il mac.

mac* = VHZTVCBI

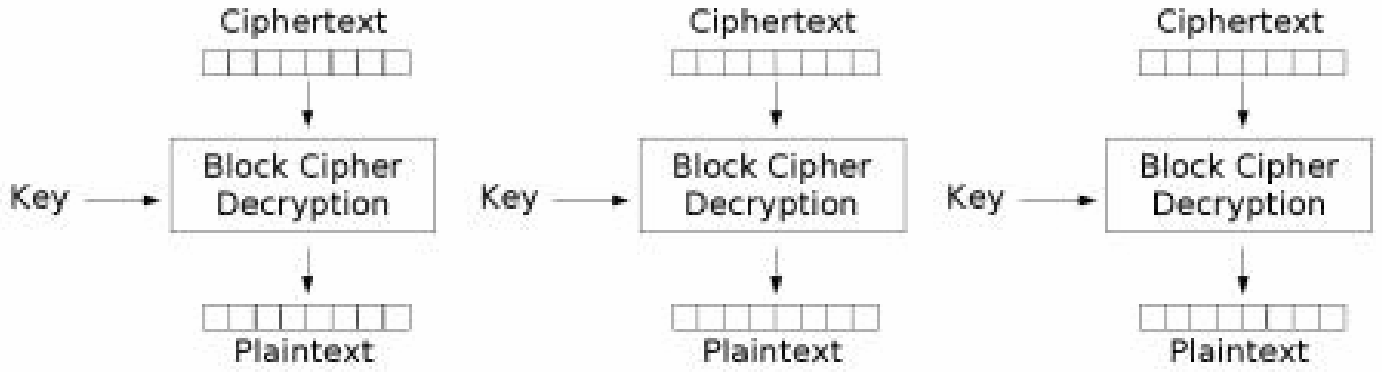
mac* != mac'  Il destinatario si accorge dell'intrusione

# Algoritmi di cifratura a blocchi

- Un algoritmo di cifratura a blocchi suddivide l'input **m** in **n** blocchi di lunghezza fissata (64,128 bit).
- Presi gli n blocchi l'algoritmo li cifra separatamente usando una chiave di lunghezza uguale a quella dei blocchi.
- L' output sarà dato dalla concatenazione dei vari blocchi in uscita.




Electronic Codebook (ECB) mode encryption

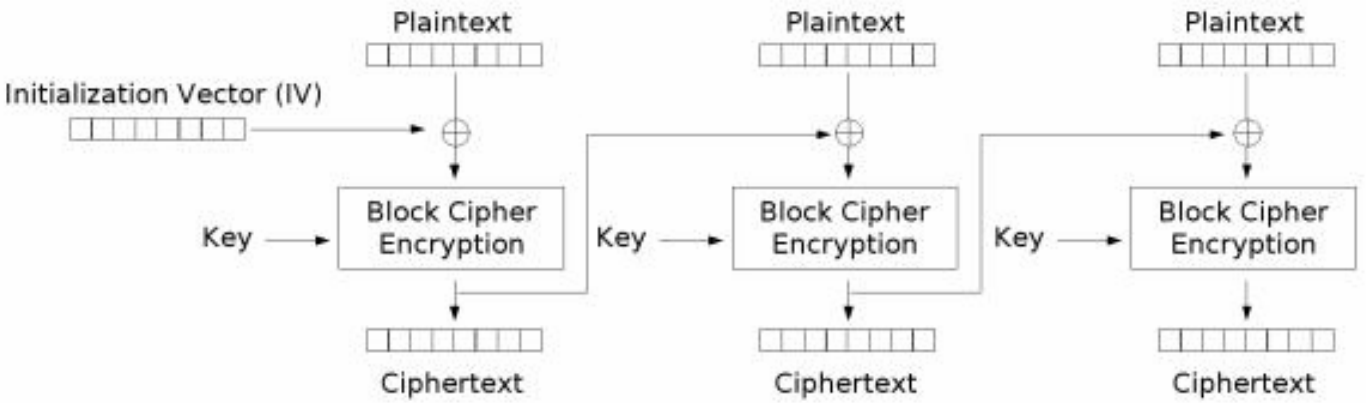



Electronic Codebook (ECB) mode decryption

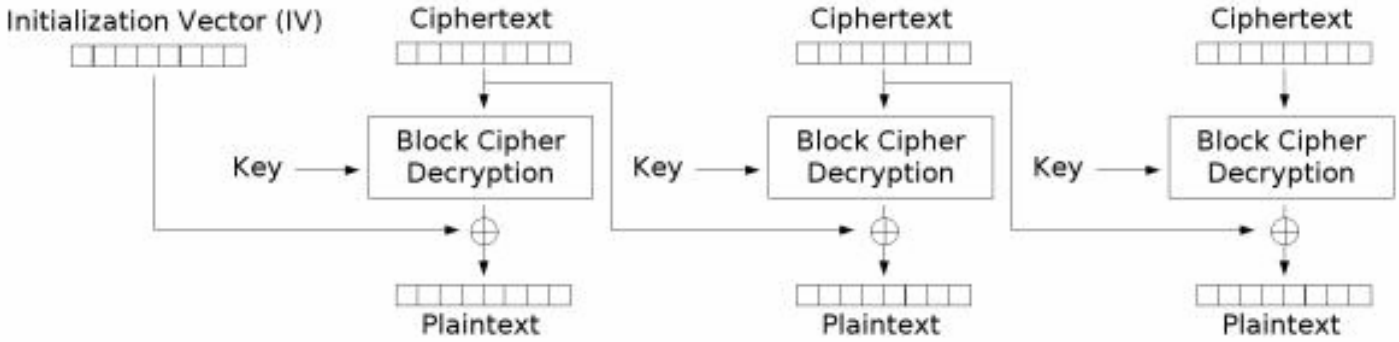
# Cipher-block-chaining mode

- Si prende un Algoritmo cifrante f che trasforma un input di n simboli in un cifrato di n simboli, usando una chiave k a sua volta di lunghezza n.
  - Divido il messaggio in blocchi di lunghezza $m_1,\ldots,m_s$
  - Calcolo $c_1 = f_k(m1)$
  - Per gli s-1 blocchi successivi :

    $c_i = f_k(c_i\text{-}1 \text{ XOR } m_i)$ $1 < i <= s$




Cipher Block Chaining (CBC) mode encryption




Cipher Block Chaining (CBC) mode decryption

# Mac con CBC

- Per ottenere un mac dall'algoritmo CBC si mette il messaggio in input, il mac sarà ottenuto dall'uscita dell ultimo blocco cifrante. mac $=c_s$
- Il mac così ottenuto avrà lunghezza n e avrà una dipendenza da tutto il messaggio

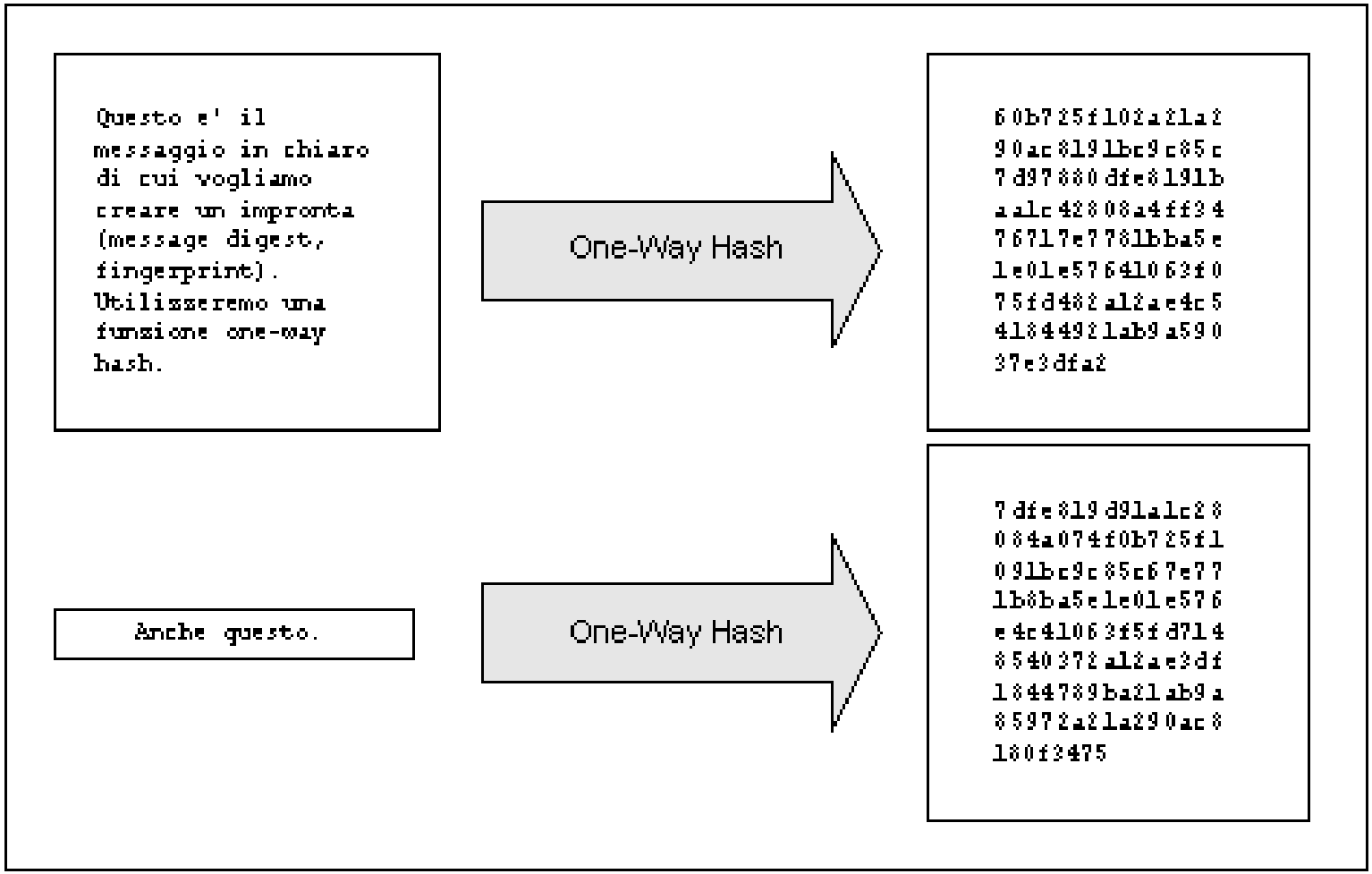
# Funzioni HASH

- Funzioni con input di lunghezza arbitraria calcolano un output di lunghezza fissata n
  - MD5 output 128 bit
  - SHA-1 output 160 bit
- Hash one-way funzioni che sono unidirezionali.
  - Impossibile dall'output trovare l'input anche sapendo la chiave

# Funzioni HASH

- Queste funzioni non sono state pensate per essere utilizzate con una chiave
  - Come possiamo utilizzarle nel computo del mac???
  - Concateniamo l'input con la chiave k in modo da far dipendere l'output da entrambi.

# Funzioni HASH



# HMAC

- Mac calcolati grazie a funzioni **hash**
- Vantaggi:
  - Tempo di calcolo inferiore.
  - Le funzioni hash sono disponibili in librerie per l'implementazione dell'algoritmo in codice.

# HMAC

- H funzione hash (MD5,SHA-1) prende input di lunghezza arbitraria e produce un output di lunghezza l (128,160 bit).
- Presa una chiave k di lunghezza max 64byte (se più corta viene allungata con 0)
- Si hanno 2 stringhe di 64 byte ipad e opad stringhe fissate: ipad 64 ripetizioni di 0x36

  opad 64 ripetizioni di 0x5c

# HMAC

- Dato un messaggio m
- Hmac = H(k XOR opad, H(k XOR ipad,m))

# Funzioni HASH

- Proprietà di una buona funzione hash
  - Collision-free
    - Impossibilità dati 2 ingressi diversi di avere la stessa uscita
  - Imprevedibilità di H
- Una funzione con queste proprietà genera un MAC sicuro