

# COMPUTER FORENSICS

## Lezione 9: Protocolli Crittografici *Funzioni di hash* (1ª parte)

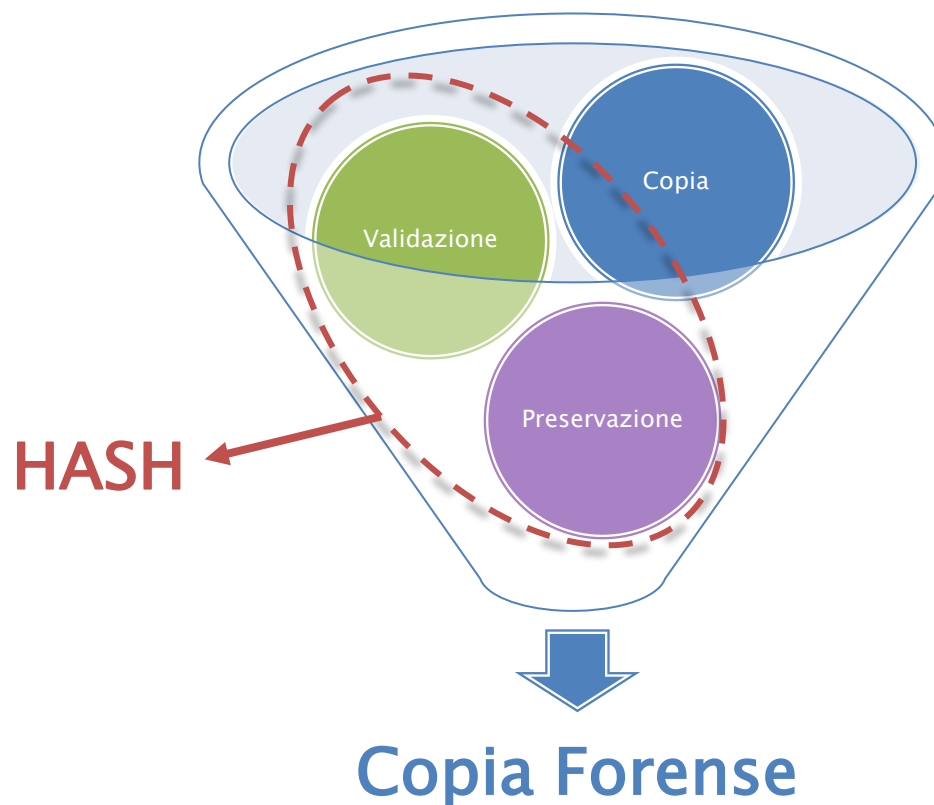


A.A. 2021/22

Dott. Lorenzo LAURATO



# Nella puntata precedente...



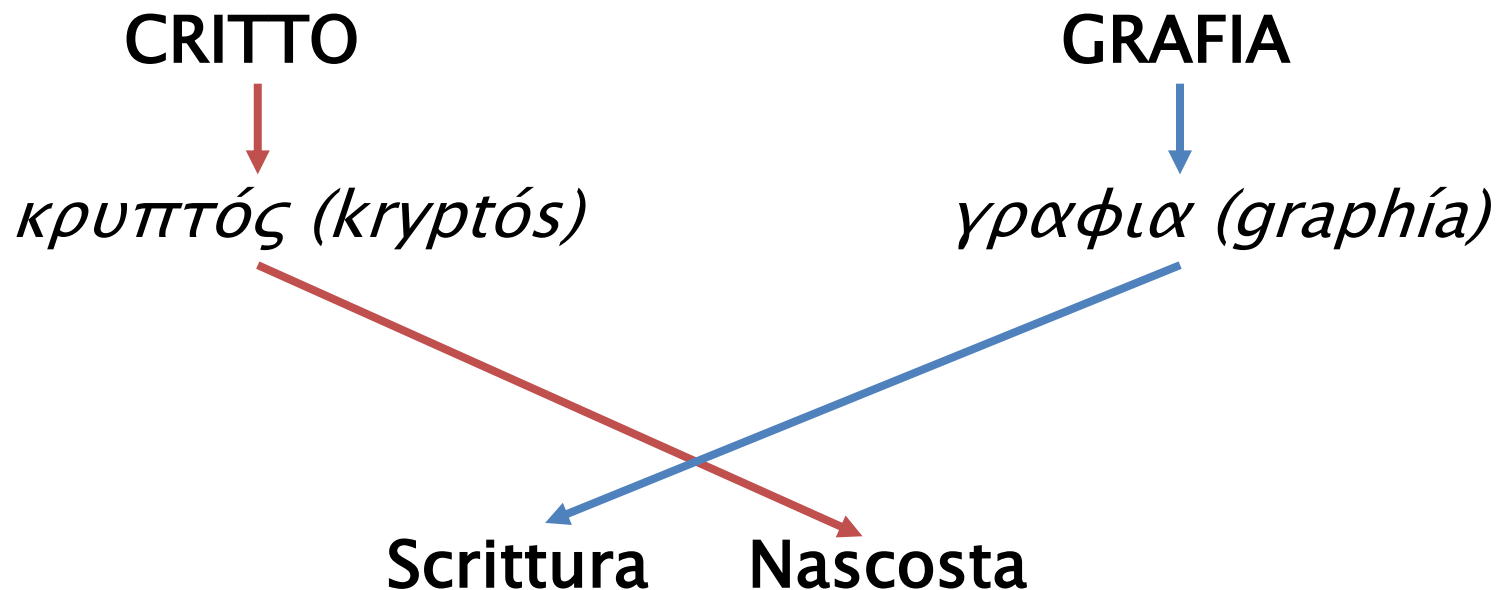
# Crittografia

## »» Introduzione



# La crittografia...

- Rendere oscuro ciò che scrivi o vuoi comunicare.



# La crittologia:

- ▶ scienza che si occupa della comunicazione in forma sicura e di solito segreta.



## Crittografia

studio e applicazione dei principi e delle tecniche per rendere l'informazione inintelligibile a tutti tranne che al destinatario

**VS**

## Crittoanalisi

scienza e arte di risolvere i crittosistemi per recuperare l'informazione nascosta

# La crittografia: storia

- ▶ utilizzo dall'antichità:
  - comunicazioni private
  - arte/religione
  - usi Militari e Diplomatici

# La crittografia: storia

## *scritture segrete*

- ▶ trasformazione delle parole per renderne incomprensibile il significato

### Geroglifico

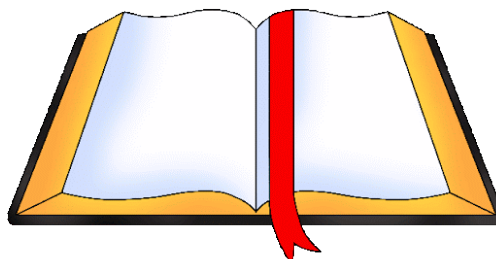


- conferire dignità e onorificenza al defunto (incisione funebre)
- mistero
- senso dell'arcano
- conferire potere magico alle parole

# La crittografia: storia *testi sacri*

## Bibbia

*(tre tecniche di cifratura a sostituzione)*



- ▶ **Atbash:** alfabeto rovesciato ( $a \rightarrow Z$ ;  $b \rightarrow Y$ ;  $c \rightarrow X$ ; ... ;  $m \rightarrow N$ )
- ▶ **Albam:** alfabeto diviso in due metà ( $a \rightarrow N$ ;  $b \rightarrow O$ ;  $c \rightarrow P$ ; ... ;  $m \rightarrow Z$ )
- ▶ **Atbah:** relazione numerica fra le lettere ( $a=1$ ;  $b=2$ ; ... ;  $z=26$ )
  - Prime 9 lettere ( $a-i$ ):  $10 - \text{lettera} \Rightarrow \text{lettera sostituyente}$ ;
  - Successive 9 lettere ( $j-s$ ):  $28 - \text{lettera} \Rightarrow \text{lettera sostituyente}$ ;
  - Ultime 8 lettere ( $t-z$ ):  $45 - \text{lettera} \Rightarrow \text{lettera sostituyente}$



# La crittografia: storia

## *cifrario di Cesare*

- ▶ lettera di Cesare a Cicerone (100–44 a.c.)

RPQLD JDOOLD HVW GLYLVD LQ  
SDUWHV WUHV




- *cifratura a sostituzione*
- relazione numerica fra le lettere (A=1; B=2; ...; Z=26)

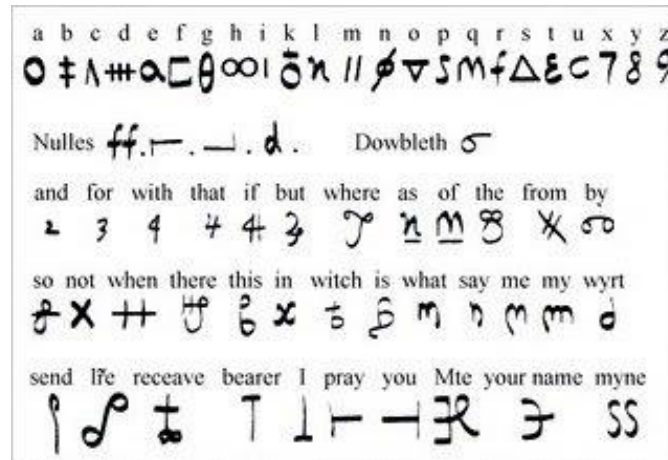
$$C(M_i) \Rightarrow M_{i+3 \bmod 26}$$

OMNIA GALLIA EST DIVISA IN  
PARTES TRES

# La crittografia: storia *congiura di Babington*

- ▶ Maria Stuarda (Regina di Scozia) e Anthony Babington stavano cospirando contro Elisabetta I
- ▶ I messaggi venivano nascosti in botti di birra e cifrati: 

- *Lettera = Simbolo;*
- *Simbolo = doppia lettera;*
- *4 simboli falsi;*
- *35 simboli per parole/frasi;*



- ▶ Il corriere (Gilbert Gifford) consegnava i messaggi anche al nemico, poi decifrati da Thomas Phelippes;

# La crittografia: storia *macchina cifrante*

## ENIGMA

*(dispositivo elettromeccanico per cifrare e decifrare messaggi)*



- ▶ Sviluppata da Arthur Scherbius [1918]
- ▶ Usata nella II Guerra Mondiale dai tedeschi.
- ▶ Decifrata nel 1932 da un gruppo di matematici polacchi.

# La crittografia: storia *i tre stadi*

## ▶ Primo stadio

- Dalle prime civiltà fino al secolo scorso
- Algoritmi sviluppati e implementati “a mano”

## ▶ Secondo stadio

- Seconda guerra mondiale
- Apparizione delle prime macchine cifranti

## ▶ Terzo stadio

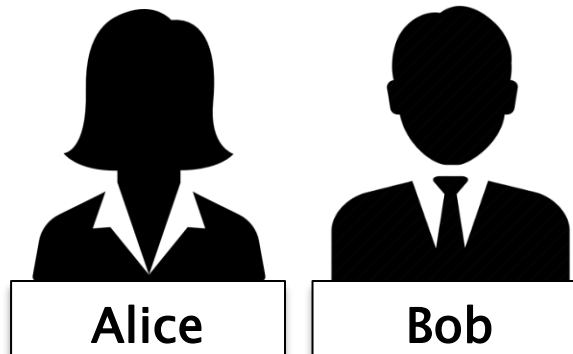
- Ultimi 50 anni
- Utilizzo di computer
- Fondamenti matematici

# La crittografia: storia *età moderna*

- ▶ Avvento dei computer come strumento di calcolo:
  - nuovi algoritmi di crittografia;
  - nuove tecniche di crittoanalisi;
- ▶ nuovi campi di applicazione
- ▶ è fondata su solide basi matematiche
- ▶ si occupa anche del progetto e della valutazione di metodi e tecniche per la protezione dell'informazione

# La crittografia: protocolli

- ▶ Un protocollo o schema definisce le interazioni fra le parti per ottenere le proprietà di sicurezza desiderate.
- ▶ **Parti:** entità coinvolte nello schema;



- ▶ **Proprietà di sicurezza:** segretezza, autenticità

# La crittografia: primitive

- ▶ I protocolli di basano su una serie di protocolli più semplici detti primitive crittografiche:
  - risolvono problemi “facili”
  - possono essere usate per risolvere problemi più complessi
- ▶ Fonti:
  - Costrutti (*Es. DES*) ;
  - Problemi matematici (*Es. teoria dei numeri*) ;

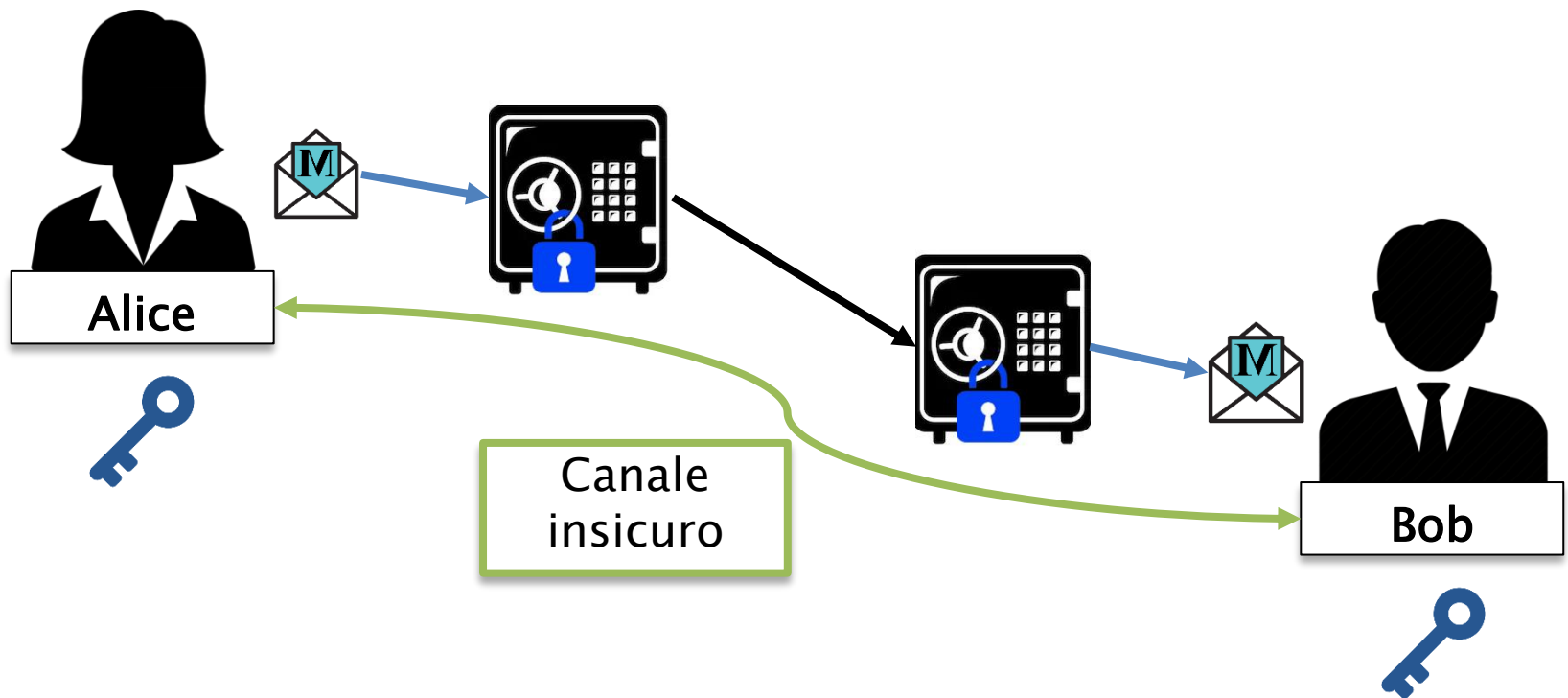
# La crittografia: primitive

- ▶ risolvono i seguenti problemi:
  - **Cifratura:** *cifrari simmetrici o asimmetrici o a chiave pubblica;*
  - **Autenticazione ed integrità:** *Funzioni Hash e MAC*
  - **Firme digitali**
  - **Altro:** *generazione di numeri pseudo-casuale, prove zero-knowledge, etc.*



# Le primitive crittografiche

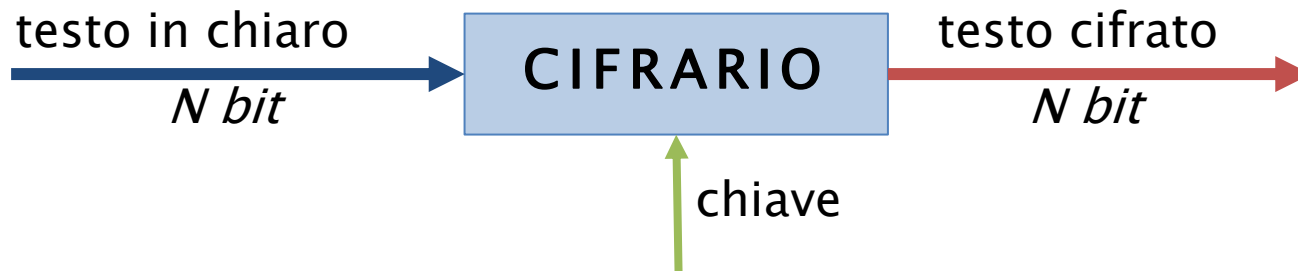
## *cifrario simmetrico*



Condividono la stessa chiave

# Le primitive crittografiche

## *cifrario simmetrico*

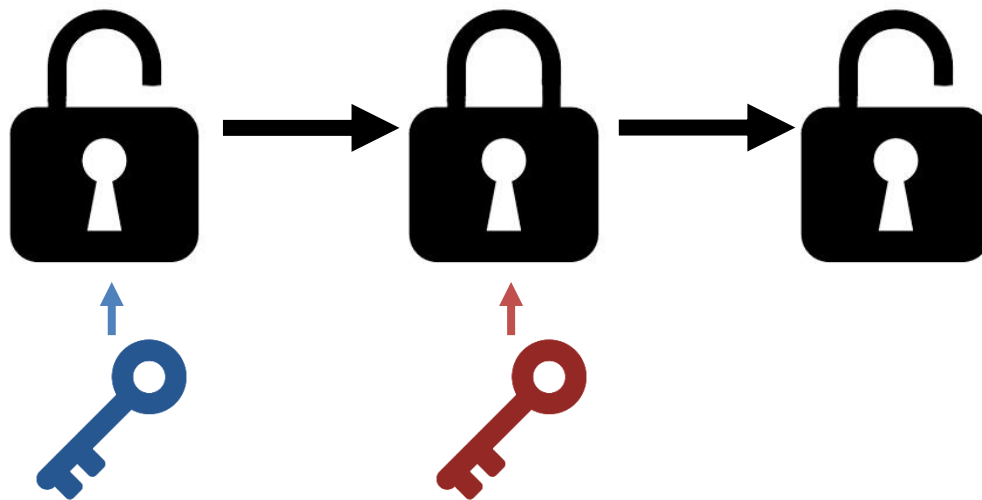


- ▶ Data Encryption Standard (DES)
  - *DES triplo, modalità di cifratura*
- ▶ RC2, RC4, RC5, RC6
- ▶ Advanced Encryption Standard (AES)
- ▶ Blowfish

# Le primitive crittografiche

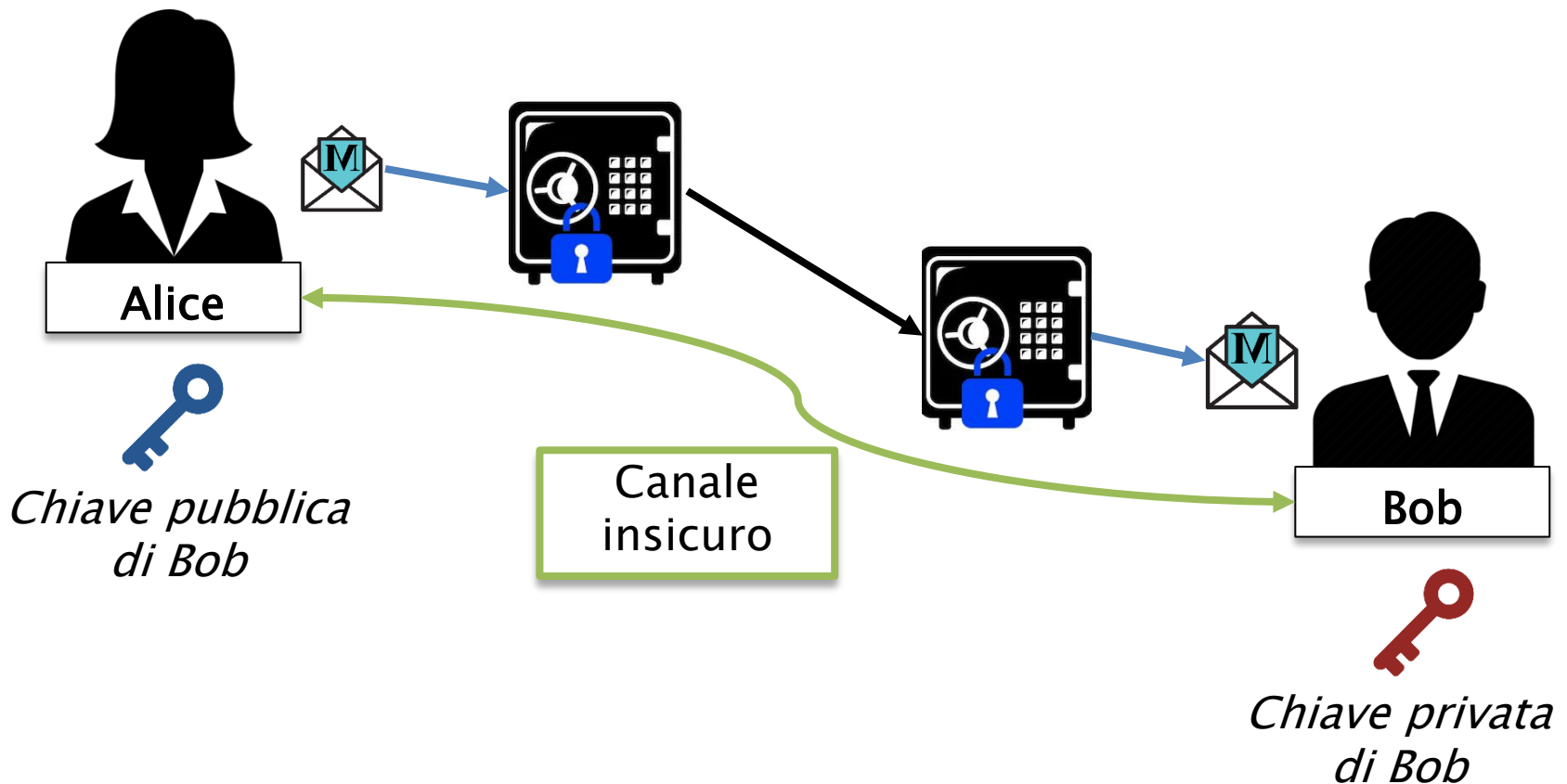
## *cifrario asimmetrico*

- ▶ Si impiegano due chiavi differenti:
  - **Chiave Pubblica**: impiegata per cifrare
  - **Chiave Privata**: impiegata per decifrare



# Le primitive crittografiche

## *cifrario asimmetrico*



# Le primitive crittografiche

## *firma digitale*

- ▶ *Apposizione di una firma ad un documento digitale*



- ▶ **Proprietà:**
  - La firma digitale deve poter essere facilmente prodotta dal legittimo firmatario.
  - Nessun utente deve poter riprodurre la firma di altri.
  - Chiunque può facilmente verificare una firma
- ▶ **Algoritmi:**
  - RSA
  - Digital Signature Standard (DSS)

# Le primitive crittografiche

## *funzione hash*

- ▶ il valore hash  $h(M)$  è una rappresentazione non ambigua e non falsificabile del «*messaggio  $M$* »



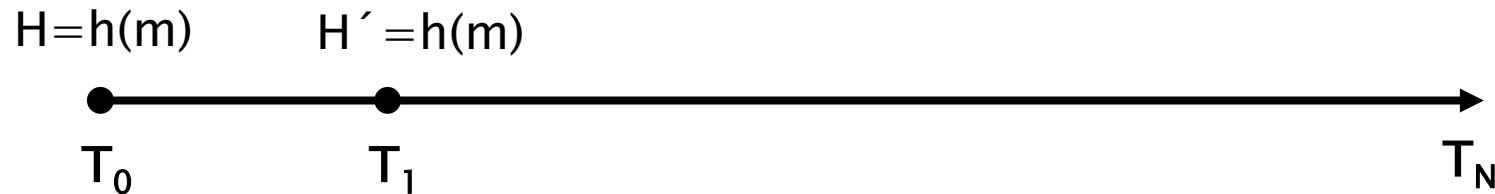
- ▶ **Impiego:**
  - Firma digitale
  - Integrità dei dati
  - Certificazione del tempo

# Le primitive crittografiche

## *funzione hash (integrità)*

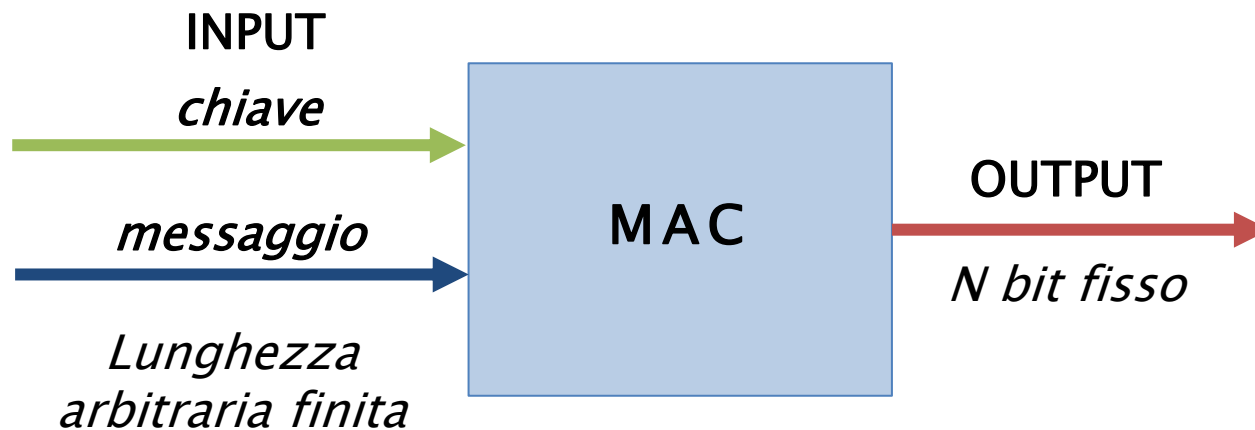
- ▶ Computo al tempo  $T_0$  il valore hash del file  $M$ :  $H = h(M)$
- ▶ Per controllare se il file è stato successivamente modificato:
  - al tempo  $T_1$  calcolo:  $H' = h(M)$ ;
  - verifico se  $H' = H$ ;

$h(M)$  è l'impronta digitale del file



# Le primitive crittografiche

## *funzione Message Authentication Code (MAC)*



### ► Impiego:

- Integrità dei dati
- Autenticità dei dati: verificare chi è stato il mittente dei dati



# Le primitive crittografiche

## *proprietà di sicurezza*

- ▶ **Confidenzialità:** protezione del dato da uno soggetto estraneo
- ▶ **Autenticazione:** certezza di indentificare l'interlocutore
- ▶ **Integrità:** verificare che il messaggio non è stato modificato durante la trasmissione.
- ▶ **Non-ripudio:** negare il disconoscimento del messaggio al mittente o al destinatario;
- ▶ **Anonimia:** nascondere l'identità di chi a compiuto una determinata azione nel contesto crittografico.

# Le primitive crittografiche

*proprietà di sicurezza: confidenzialità*

Privacy



Segretezza

Protezione dell'informazione trasmessa dall'accesso da soggetti non autorizzati



Sistema di cifratura simmetrici/asimmetrici

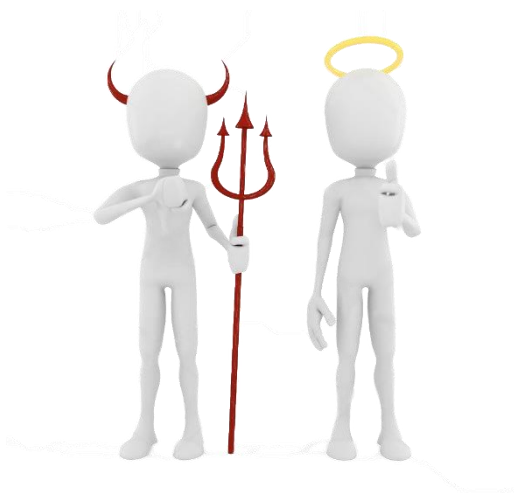
# Le primitive crittografiche

## *proprietà di sicurezza: autenticazione*

Origine del  
messaggio



Soggetto \ Entità  
con il quale si sta  
interloquendo




Temporale



# Le primitive crittografiche

## *proprietà di sicurezza: integrità*

- ▶ Solo chi è autorizzato può modificare l'attività di un sistema o le informazioni trasmesse



scrittura, cambiamenti,  
cancellazione, creazione,  
ritardi, replay e  
riordino di messaggi, etc.

# Le primitive crittografiche

## *proprietà di sicurezza: non ripudiabilità*

- ▶ è impossibile negare l'occorrenza di una determinata azione



# Le primitive crittografiche

*proprietà di sicurezza: anonimia*



- ▶ proteggere l'identità di chi sta utilizzando un servizio o proteggere l'accesso al servizio stesso: **grado di anonimia**

# Hash

» MD4/5 e SHA1



# Funzione Hash

- ▶ il valore hash  $h(M)$  è una rappresentazione non ambigua e non falsificabile del «*messaggio  $M$* »



Integrità dei dati

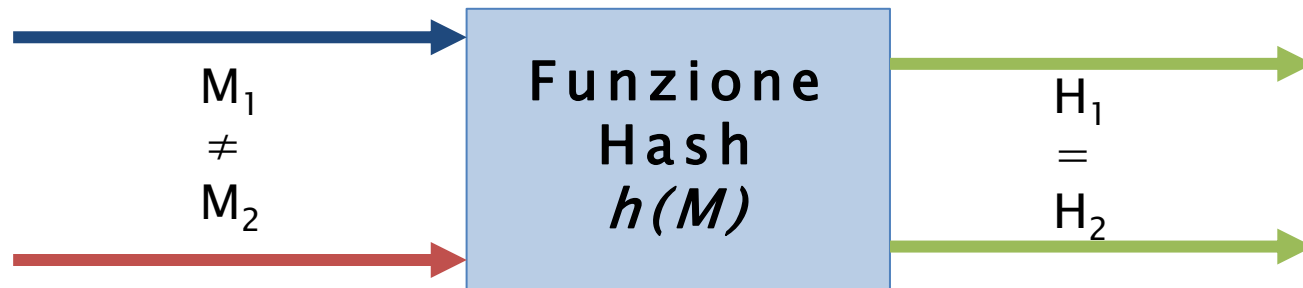


# Funzione Hash

## *collisione*

$$h: \Sigma^* \rightarrow \Sigma^n$$

$$h(m_1) = h(m_2)$$



Esistono infinite collisioni

# Funzione Hash

## *proprietà*

- ▶ **One-way (*pre-image resistant*):**
  - dato un *hash*  $y$ , è computazionalmente difficile trovare  $M \mid y=h(M)$
- ▶ **Sicurezza debole (*2nd pre-image resistance*):**
  - dato  $M$ , è computazionalmente difficile trovare una variazione di  $M$ ,  $M' \mid h(M)=h(M')$
- ▶ **Sicurezza forte (*collision resistance*):**
  - computazionalmente difficile trovare 2 diversi messaggi con lo stesso valore hash

# Funzione Hash

## *definizioni*

- ▶ **Una One-Way Hash Function (OWHF):**
  - verifica le proprietà pre-image e 2nd pre-image resistant;
  - viene detta *weak one-way hash function*
- ▶ **Una Collision Resistant Hash Function (CRHF):**
  - verifica la proprietà di collision resistance
  - viene detta *strong one-way hash function*

# Funzione Hash

## *costruzione*

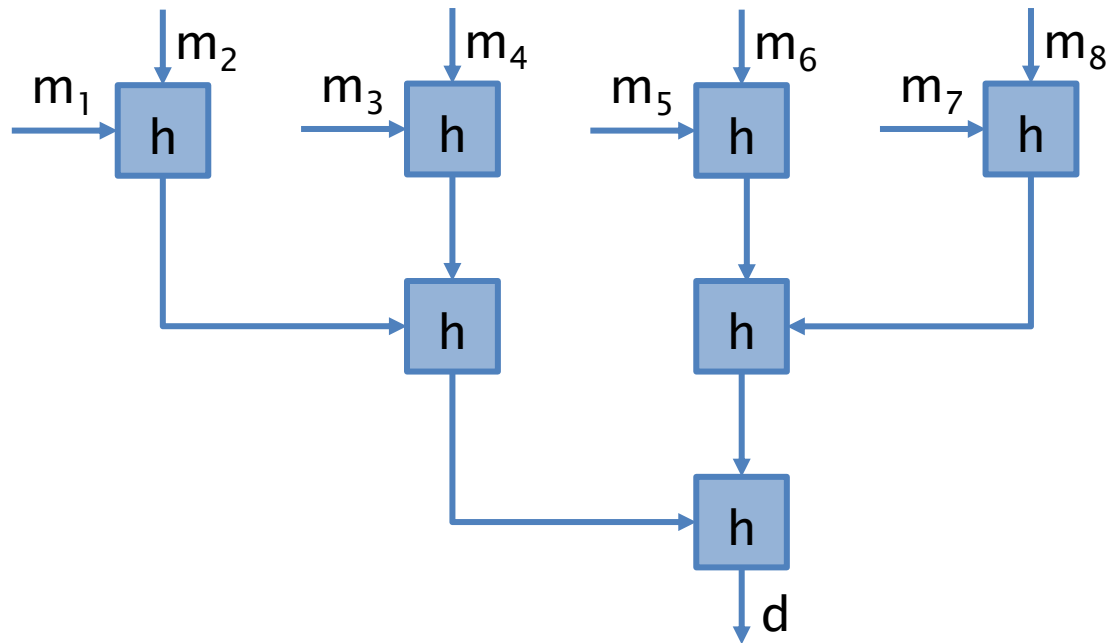
Messaggi di lunghezza arbitraria sono trattati utilizzando hash con input fisso:

- ▶ il messaggio input **M** viene diviso in **k blocchi** di lunghezza fissa:  $m_1, m_2, \dots, m_k$
- ▶ i blocchi vengono trattati in modo:
  - seriale/iterato
  - parallelo

# Funzione Hash

## *costruzione*

### Modello hash parallelo

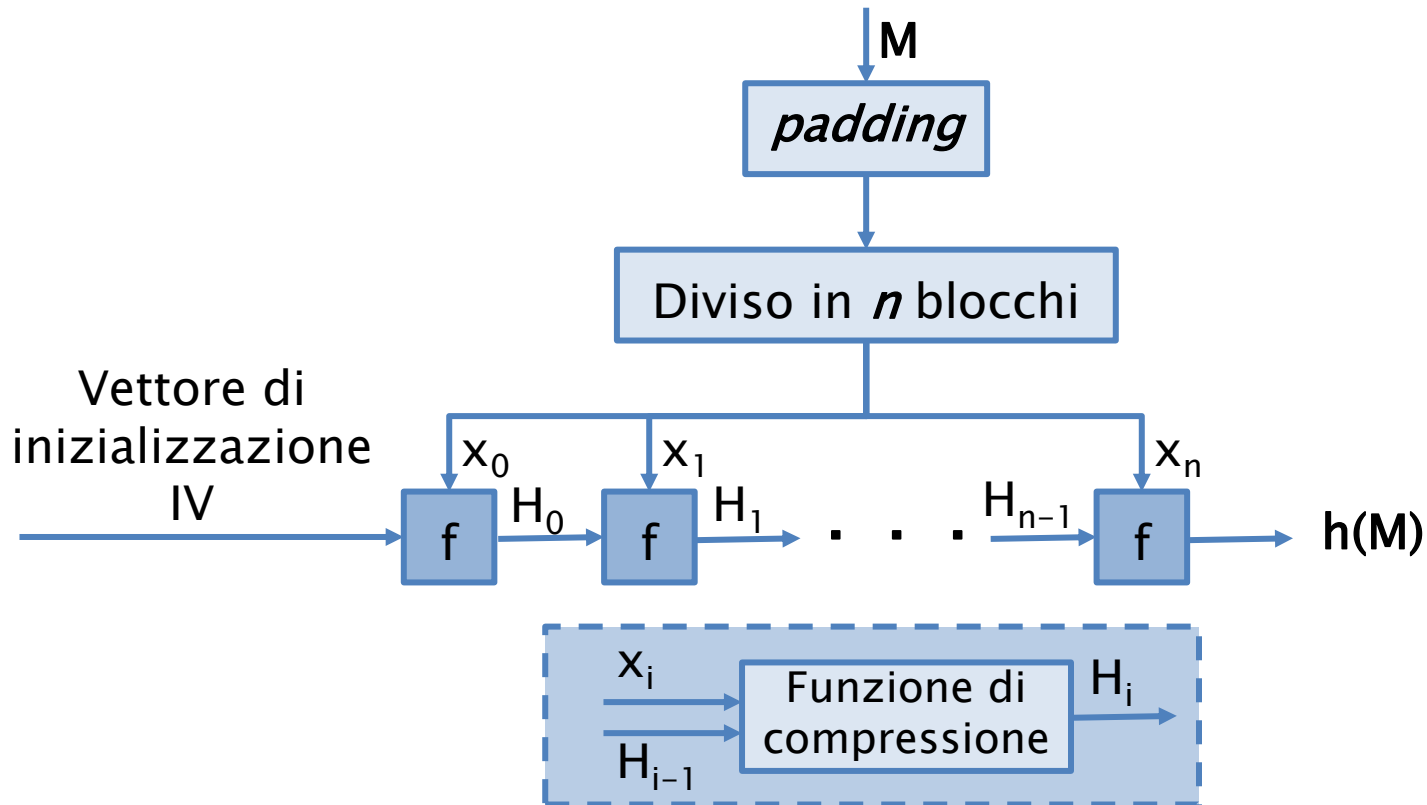


*È resistente alle collisioni se lo è la funzione  $h$*

# Funzione Hash

## *costruzione*

### Modello hash iterato

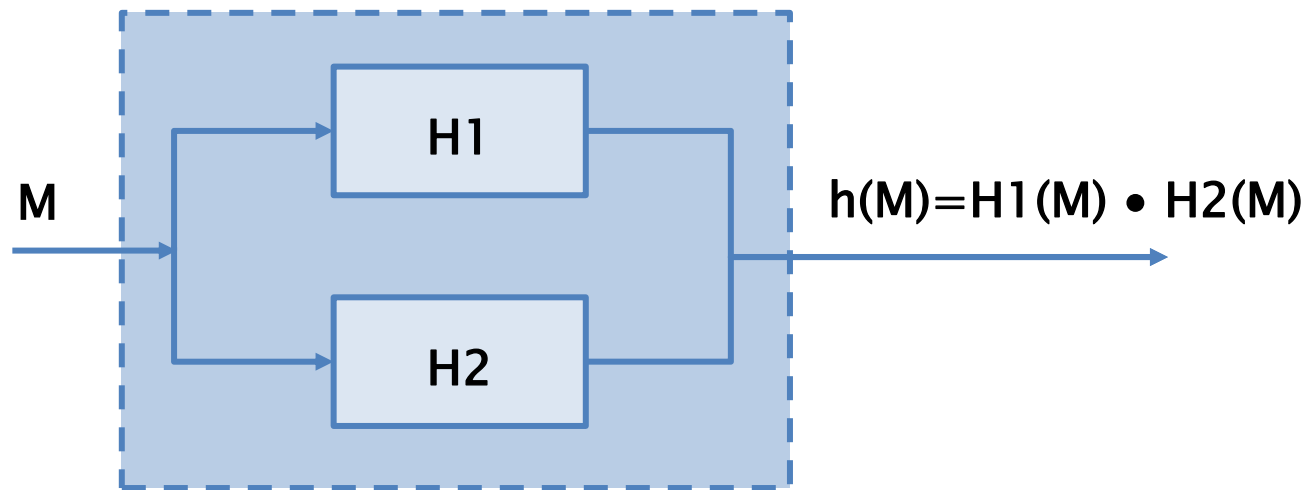


*una collisione per  $h(M)$  implica una collisione di  $f$*

# Funzione Hash

## *costruzione*

### Modello hash cascata



una collisione per  $h(M)$  vuol dire trovare  
una collisione sia per H1 che per H2

# Funzione Hash

## *costruzione*

### Cifrari a blocchi

- ▶ Cifrario a blocchi  $E_k(\bullet)$  per input ad  $n$  bit
- ▶ Funzione  $g$  che da  $n$  bit produce una chiave
  - $M'_1 \dots M'_t$  : è il messaggio  $M$  con eventuale *padding*
  - $H_0$  : è una costante predefinita
  - $H_t$  : è il valore hash
- $H_i = E_{g(H_{i-1})}(M'_i) \oplus M'_i$  [Matyas–Meyer–Oseas]
- $H_i = E_{g(H_{i-1})}(M'_i) \oplus M'_i \oplus H_{i-1}$  [Miyaguchi–Preneel]
- $H_i = E_{M'_i}(H_{i-1}) \oplus H_{i-1}$  [Davies–Meyer]



Fine prima parte...



## SSRI Lorenzo Laurato s.r.l.



Via Coroglio nr. 57/D (BIC- Città della Scienza)  
80124 Napoli



Tel. 081.19804755

Fax 081.19576037



lorenzo.laurato@unina.it

lorenzo.laurato@ssrilab.com



[www.docenti.unina.it/lorenzo.laurato](http://www.docenti.unina.it/lorenzo.laurato)

[www.computerforensicsunina.forumcommunity.net](http://www.computerforensicsunina.forumcommunity.net)