



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Introduzione alla Crittografia (sulla sicurezza ed altre amenità informatiche)

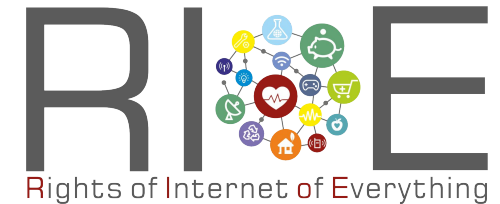
Mirko Zichichi mirko.zichichi@iota.org

slides in collaborazione con
prof. Stefano Ferretti

Mirko Zichichi <https://mirkozichichi.me>

PhD in **Law, Science, and Technology Joint Doctorate - Rights of Internet of Everything**

Ontology Engineering Group (OEG), Universidad Politécnica de Madrid
Dipartimento di Giurisprudenza, Università di Torino
Dipartimento di Scienze Giuridiche, **Università di Bologna**



IOTA
FOUNDATION

Ricercatore presso la **IOTA Foundation**

e-mail: mirko.zichichi@iota.org

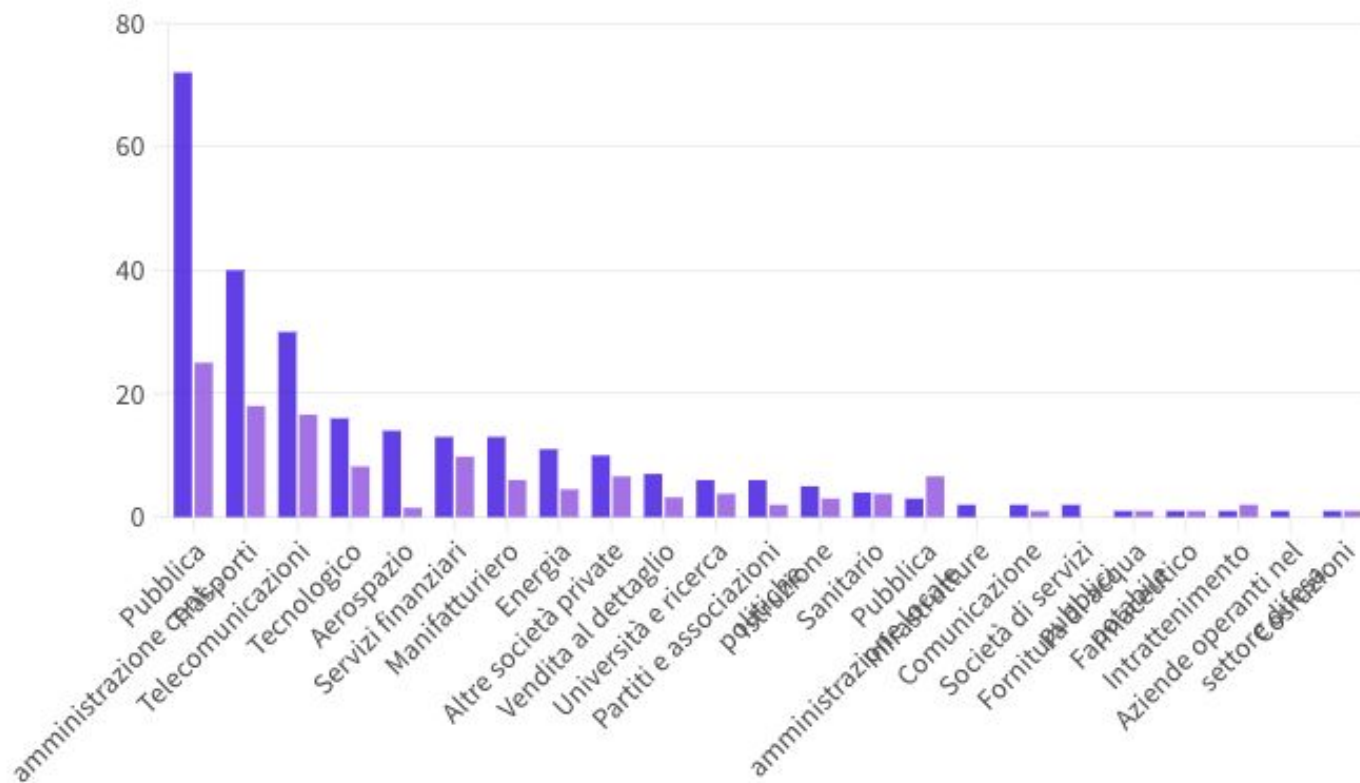


ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Numero di eventi cyber per settore

Confronto tra il 1° semestre del 2024 e il 2° semestre del 2023

■ Attacchi semestre 1 del 2024 ■ Attacchi semestre 2 del 2023



Fonte: Agenzia per la cybersicurezza nazionale





Comune di Palermo
@ComunePalermo · Follow



Attacco hacker. Amministrazione comunale: "Garantiti i servizi demografici"

In riferimento al grave attacco hacker dei giorni scorsi alle infrastrutture tecnologiche del Comune di Palermo sono state poste in essere una serie di attività volte a contenere l'attacco ransomware.



3:13 PM · Jun 9, 2022



21



Reply



Copy link

Read 3 replies



La Sicurezza Informatica

- La sicurezza informatica è la **protezione delle risorse dall'accesso, utilizzo, alterazione o distruzione non autorizzati**
- Due tipi di sicurezza
 - **Fisica**: protezione dei dispositivi fisici tramite allarmi, antifurto, porte blindate, casseforti...
 - **Logica**: protezione delle informazioni tramite risorse non fisiche (crittografia, firma elettronica...)



Sicurezza Informatica: Terminologia

- **Confidenzialità (Segretezza)**
 - Impedire la divulgazione non autorizzata di dati, garantire l'autenticità della fonte
- **Integrità**
 - Impedire le modifiche non autorizzate ai dati
- **Autenticazione**
 - Verificare l'identità della controparte (*con chi sto comunicando?*)
- **Disponibilità**
 - Impedire ritardi nella diffusione dei dati, o la loro rimozione
 - Es: Attacchi *Denial of Service* (DoS), Ransomware
- **Non ripudiabilità**
 - Impedire che la controparte possa negare una sua azione



Aspetti fondamentali

1. inesistenza di sistemi sicuri
2. sicurezza = conoscenza
3. entità componenti di un sistema (in)sicuro



1. Sistemi Sicuri

NON ESISTONO SISTEMI SICURI

- Il software può non essere perfetto
 - usualmente non lo è
 - ci possono esserci errori di progettazione nei protocolli che usiamo normalmente
- Il mito del sistema **inviolabile** deve essere assimilato a quello del caveau non svaligiabile o della nave inaffondabile (es. Titanic)
- Il **grado di sicurezza** è dato dal **tempo** necessario per violare il sistema, dall'**investimento** necessario e dalla **probabilità** di successo



1. Sistemi Sicuri: Complessità



UN SISTEMA PIÙ È COMPLESSO PIÙ È INSICURO

- **KISS** rule:

Keep It Simple and Stupid

- Eliminare la complessità non necessaria
 - “Quello che non c'è non si rompe”



2. Sicurezza = Conoscenza

- Nessun sistema sconosciuto può essere considerato sicuro
- Quali implicazioni ha questa affermazione sul tema del software **Open Source vs. Closed Source, Sistemi Aperti vs. Sistemi Chiusi?**
 - Atto di fiducia nei confronti di una comunità vs. di un'azienda
- L'educazione degli utenti è una forma di conoscenza fondamentale
 - (es. percezione dell'importanza del tema sicurezza, pratiche comunemente diffuse ma altamente insicure, resistenza ideologiche o per semplice abitudine ecc.)



2. Sicurezza = Conoscenza

Principio di Kerckhoffs

*“Un sistema crittografico deve essere sicuro anche se ogni suo aspetto, **tranne le chiavi**, è noto pubblicamente”*

- Il contrario di “Security by obscurity”
- “System security should not depend on the secrecy of the implementation or its components”

NIST special publication 800-123, Guide to General Server Security,

<http://dx.doi.org/10.6028/NIST.SP.800-123>



Auguste Kerckhoffs (1835 – 1903),
Source:

<https://commons.wikimedia.org/w/index.php?curid=39712823>



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

3. Entità Componenti di un Sistema

- Le entità che compongono un sistema sono:
 - 1) **HARDWARE**
 - 2) **SOFTWARE**
 - 3) **HUMANWARE**
- La sicurezza è un processo:
 - Serve un continuo apporto di lavoro e di educazione
 - Un sistema considerato sicuro oggi può non esserlo domani
 - I sistemi che non vengono aggiornati in modo continuo divengono fragili e insicuri
- La componente umana non deve **mai** essere sottovalutata



3. Entità Componenti di un Sistema:

3) HUMANWARE - Top Passwords 2018

- 123456
- password
- 123456789
- 12345678
- 12345
- 111111
- 1234567
- sunshine
- qwerty
- iloveyou

<https://www.teamsid.com/100-worst-passwords-top-50/>



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

3. Entità Componenti di un Sistema:

3) HUMANWARE - Top Passwords 2018

- 4.7% degli utenti ha come password *password*
- 8.5% ha come passwords *password* o *123456*
- 9.8% ha come passwords *password*, *123456* o *12345678*
- 14% ha una password tra le top 10 passwords
- 40% ha una password tra le top 100 passwords
- 79% ha una password tra le top 500 passwords
- 91% ha una password tra le top 1000 passwords



Tu che password usi?

- Quasi tutto il mondo usa le stesse pwd
- Quindi è banale entrare nella maggior parte degli account
- Ecco come si spiegano tanti dei "furti di identità"
- Inoltre, studi hanno verificato che è facile recuperare password da semplici informazioni pubbliche su facebook



Dictionary Attack

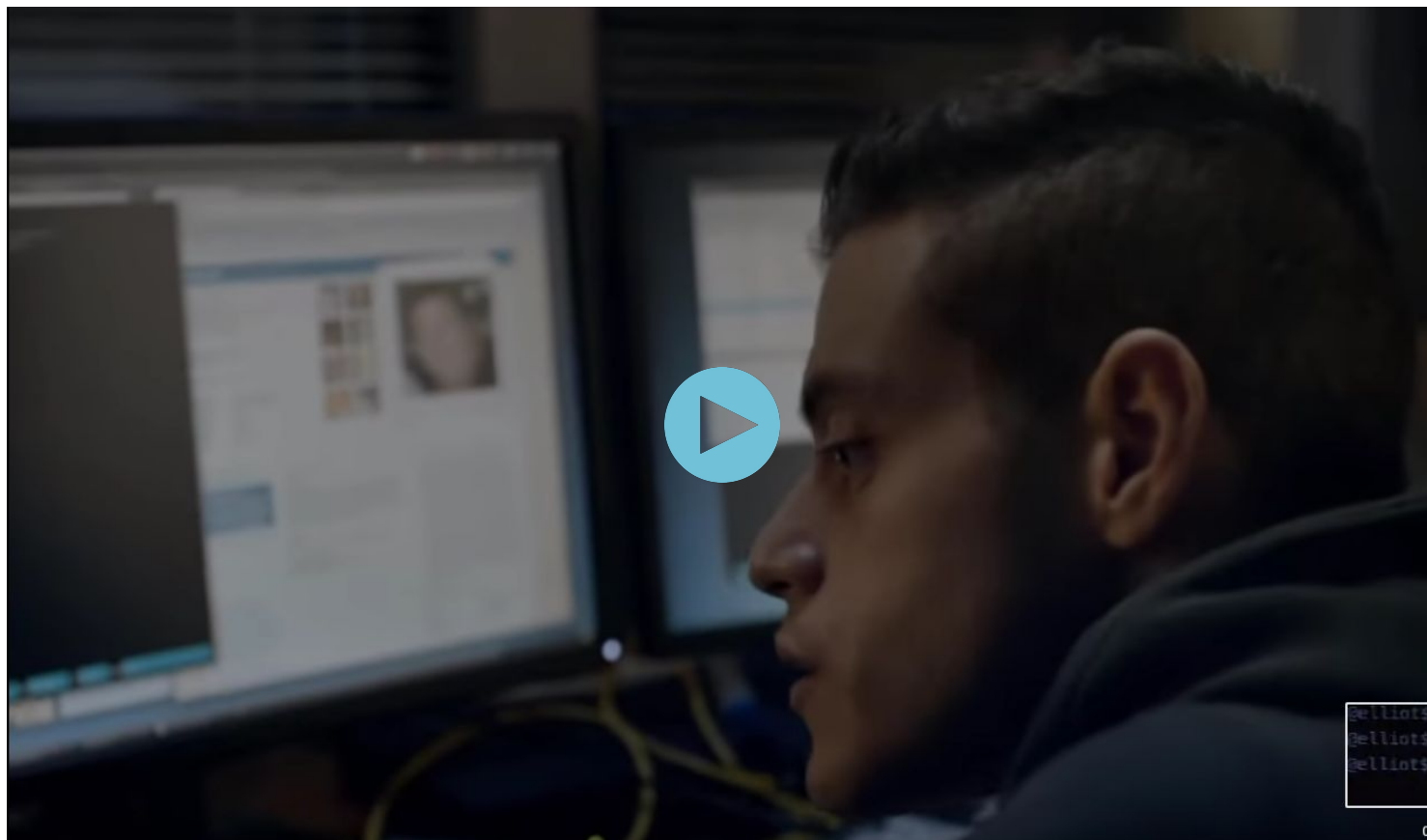
```
Activities Terminal Thu 22:14
chad@chad-Pc: ~/wpscan

File Edit View Search Terminal Help

[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "qwerty" - 137 of 168 [child 0] (98/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "111111" - 138 of 168 [child 1] (99/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "iloveu" - 139 of 168 [child 3] (100/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "000000" - 140 of 168 [child 2] (101/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "michelle" - 141 of 168 [child 0] (102/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "tigger" - 142 of 168 [child 1] (103/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "sunshine" - 143 of 168 [child 3] (104/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "chocolate" - 144 of 168 [child 0] (105/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "password1" - 145 of 168 [child 1] (106/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "soccer" - 146 of 168 [child 2] (107/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "Password123" - 147 of 168 [child 3] (108/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "123456789asd\" - 148 of 168 [child 0] (109/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "123456789asd" - 149 of 168 [child 1] (110/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "anthony" - 150 of 168 [child 2] (111/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "friends" - 151 of 168 [child 3] (112/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "butterfly" - 152 of 168 [child 0] (113/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "1234567890" - 153 of 168 [child 1] (114/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "purple" - 154 of 168 [child 2] (115/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "angel" - 155 of 168 [child 3] (116/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "123456789" - 156 of 168 [child 0] (117/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "jordan" - 157 of 168 [child 1] (118/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "12345" - 158 of 168 [child 2] (119/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "password" - 159 of 168 [child 3] (120/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "123456" - 160 of 168 [child 0] (121/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "princess" - 161 of 168 [child 1] (122/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "1234567" - 162 of 168 [child 2] (123/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "iloveyou" - 163 of 168 [child 3] (124/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "rockyou" - 164 of 168 [child 0] (125/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "nicole" - 165 of 168 [child 1] (126/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "12345678" - 166 of 168 [child 2] (127/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "abc123" - 167 of 168 [child 3] (128/129)
[REDO-ATTEMPT] target 192.168.1.10 - login "Administrator" - pass "daniel" - 168 of 168 [child 0] (129/129)
1 of 1 target completed, 0 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-09-28 22:14:06
chad@chad-Pc:~/wpscan
```



Phishing + Dictionary Attack



Video: <https://www.youtube.com/watch?v=JMYEr4Bgey4>



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

"People often represent the **weakest link in the security chain** and are chronically responsible for the failure of security systems."

Secrets and Lies: Digital Security in a Networked World, Bruce Schneier, 2000



“Are humans really the weakest link?”

Erik Hollnagel: “***all human activity*** --individually and/or collectively-- ***is variable*** in the sense that it is ***adjusted to the conditions***. The variability is therefore a strength, indeed a necessity, rather than a liability.” <https://erikhollnagel.com/ideas/no-view-of-human-error.html>

La variabilità dell’attività umana è un ->

Compromesso tra **efficienza** e **rigorosità**

Secondo questo principio, le richieste di produttività tendono a *ridurre la rigorosità*, mentre le richieste di *sicurezza riducono l'efficienza*



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Crittografia

Crittografia

- Disciplina che studia le tecniche per **cifrare un messaggio** in modo tale che solo il **legittimo destinatario** sia in grado di leggerlo
- Requisiti:
 - Cifrare/decifrare messaggi deve essere ragionevolmente efficiente
 - Deve essere “difficile” interpretare un messaggio cifrato da parte di chi non è autorizzato



Crittografia applicata



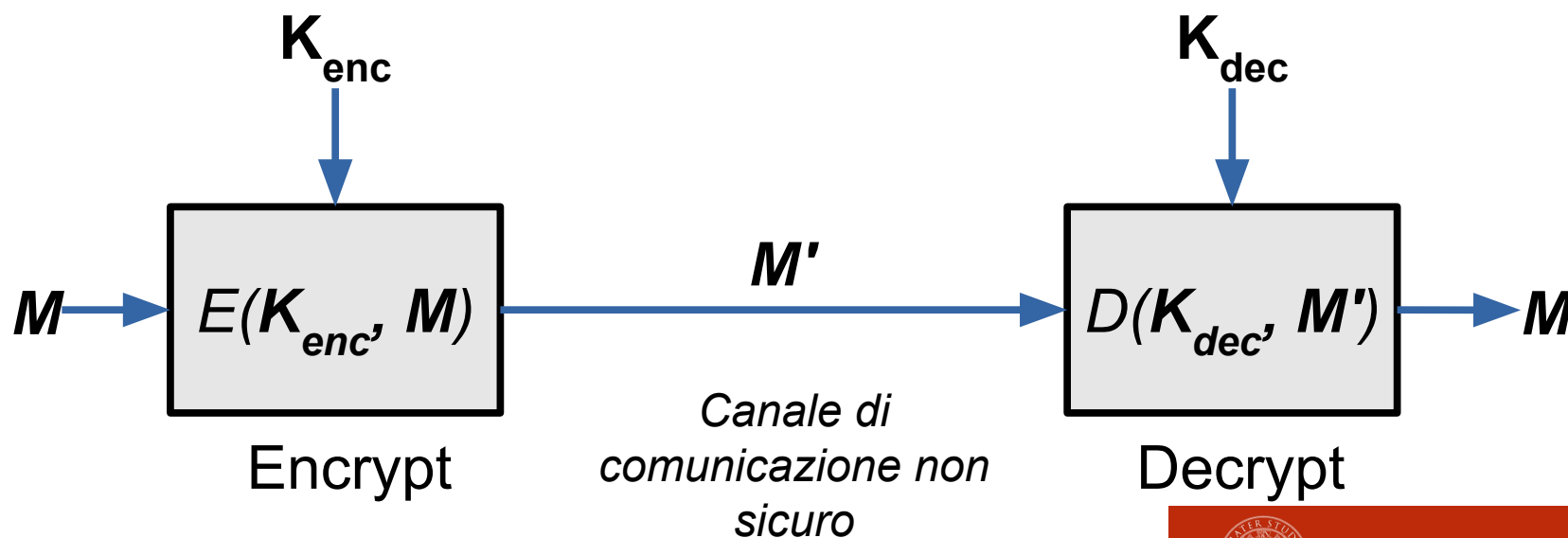
Il cifrario a cilindro di Thomas Jefferson
(1800, comunicazioni diplomatiche)

ENIGMA
(Seconda Guerra Mondiale, esercito tedesco)



Principio di base

- **Algoritmo di cifratura:** trasforma un messaggio “in chiaro” M in un messaggio cifrato M'
- **Chiavi:** necessarie per l'algoritmo di cifratura K_{enc} e decifratura K_{dec}
- **Algoritmo di decifratura:** ricavava dal messaggio cifrato M' il messaggio in chiaro M



Notazione

- $E(K_{enc}, M) = M'$ <- Funzione di **cifratura (encryption)**
 - K_{enc} <- chiave di cifratura (encryption key)
 - M <- messaggio in chiaro (plaintext)
 - M' <- messaggio cifrato (ciphertext)
-
- $D(K_{dec}, M') = M$ <- Funzione di **decifratura (decryption)**
 - K_{dec} <- chiave di decifratura (decryption key)



Esempio

Cifrario di Cesare

<https://cryptii.com/pipes/caesar-cipher>



Sistemi crittografici

- Si deve sempre avere che:

$$D(K_{\text{dec}}, E(K_{\text{enc}}, M)) = M$$

- Se cifro un messaggio M usando la chiave K_{enc} e decifro il risultato usando K_{dec} , devo ottenere nuovamente M



Sistemi crittografici

- A **chiave segreta** (crittografia **simmetrica**)
 - $K_{\text{enc}} = K_{\text{dec}}$ <- le chiavi di cifratura e decifratura sono **uguali**
- A **chiave pubblica** (crittografia **asimmetrica**)
 - $K_{\text{enc}} \neq K_{\text{dec}}$ <- le chiavi di cifratura e decifratura sono **diverse**

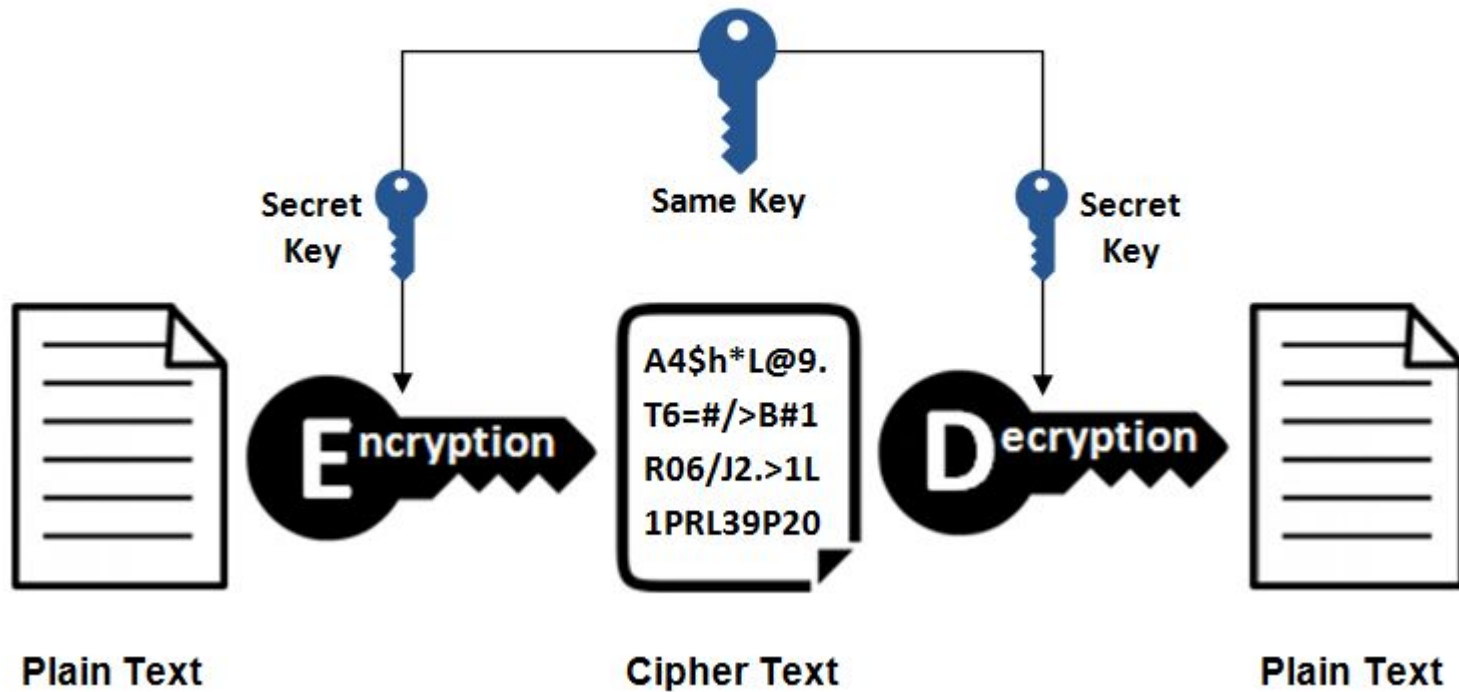




ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

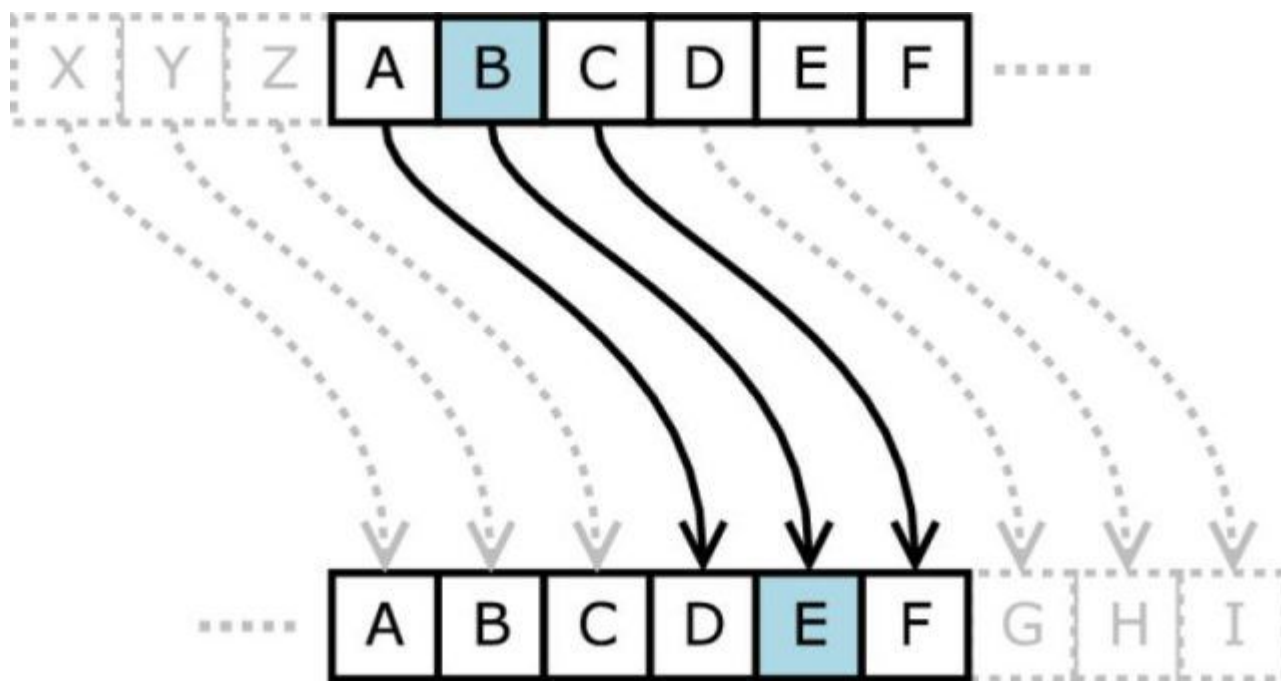
Crittografia Simmetrica

Symmetric Encryption



Sistemi crittografici a chiave simmetrica

- Uno dei primi esempi è il “cifrario di Cesare”
 - La chiave K è un numero intero
 - Ogni lettera dell'alfabeto viene sostituita da quella che la segue di K posizioni



Sistemi crittografici a chiave simmetrica

- Cifrario Polimorfico:
 - Un cifrario che **cambia ad ogni cifratura**
 - Ogni volta darà risultati diversi
 - Esempio:
 - Usiamo il cifrario di Cesare, ma cambiamo il valore di K secondo una sequenza pre-determinata
 - Ex: {3, -1, 2, 6, ...}
 - 1° cifratura di “Hello” → “Khoor” (shift di 3 lettere)
 - 2° cifratura di “Hello” → “Gdkkn” (shift di -1)
 - ...



DES

(Data Encryption Standard)

- Progettato da IBM e adottato come standard dal governo USA nel 1977
 - Chiave lunga 56 bit
 - Messaggio diviso in blocchi da 64 bit che vengono cifrati individualmente
- Esistono $2^{56} \approx 7.2 \times 10^{16}$ chiavi
 - Sembra un numero grande, ma un moderno calcolatore può esaminarle tutte in poche ore!
- Una variante (*Triplo DES*) usa chiavi più lunghe e fornisce un livello accettabile di sicurezza



AES

(Advanced Encryption Standard)

- Adottato come standard nel 2001, sostituisce DES
- Caratteristiche di AES
 - Il messaggio viene scomposto in **blocchi da 128 bit** che vengono cifrati individualmente
 - Si possono usare chiavi lunghe **128, 192 o 256 bit**
 - Esistono $2^{128} \approx 3.4 \times 10^{38}$ chiavi a 128 bit, per cui esaminarle tutte è **al momento** impraticabile



Quanto tempo serve per esplorare tutto lo spazio delle chiavi?

Tipo password	Lunghezza	Numero di password possibili	Tempo brute force a 10^9 pwd/s	Tempo brute force a 10^{13} pwd/s
alfa	8	$26^8 \approx 2.09 \times 10^{11}$	3.48 min	Istantaneo
alfa	10	$26^{10} \approx 1.41 \times 10^{14}$	1.64 giorni	14 sec
alfa	12	$26^{12} \approx 9.54 \times 10^{16}$	3.03 anni	2 ore
alfa+ALFA+num	12	$52^{12} \approx 3.23 \times 10^{21}$	1.0×10^5 anni	10 anni
alfa+ALFA+num	15	$62^{15} \approx 7.69 \times 10^{26}$	2.4×10^{10} anni	2.4×10^6 anni



Crittografia Simmetrica

- Punti a favore
 - Efficienza degli algoritmi di cifratura e decifratura
- Punti deboli
 - **Distribuzione delle chiavi** – Necessario un meccanismo sicuro per decidere la chiave da usare tra due parti
 - Scalabilità – Per ogni coppia di interlocutori, serve una chiave diversa → il numero di chiavi cresce in maniera esponenziale
 - Sicurezza limitata – Fornisce confidenzialità, ma limitata autenticità e non garantisce non-repudiabilità



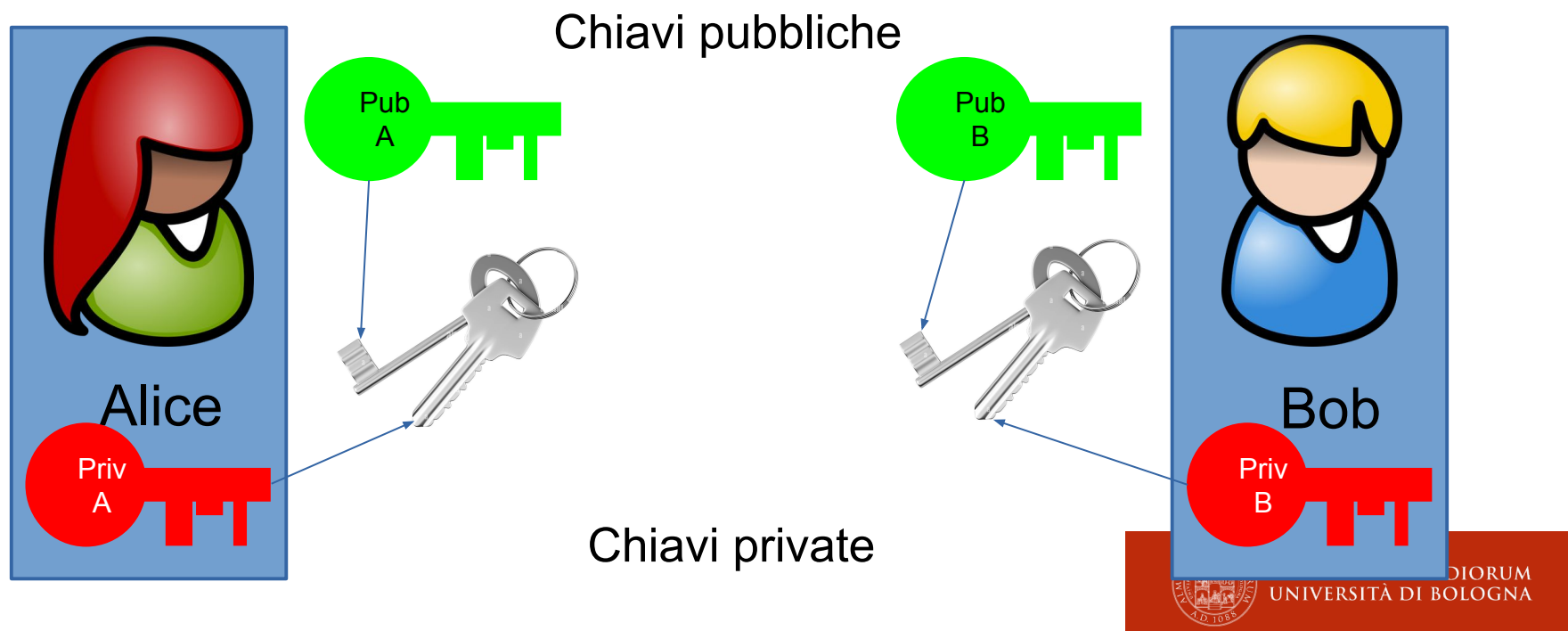


ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

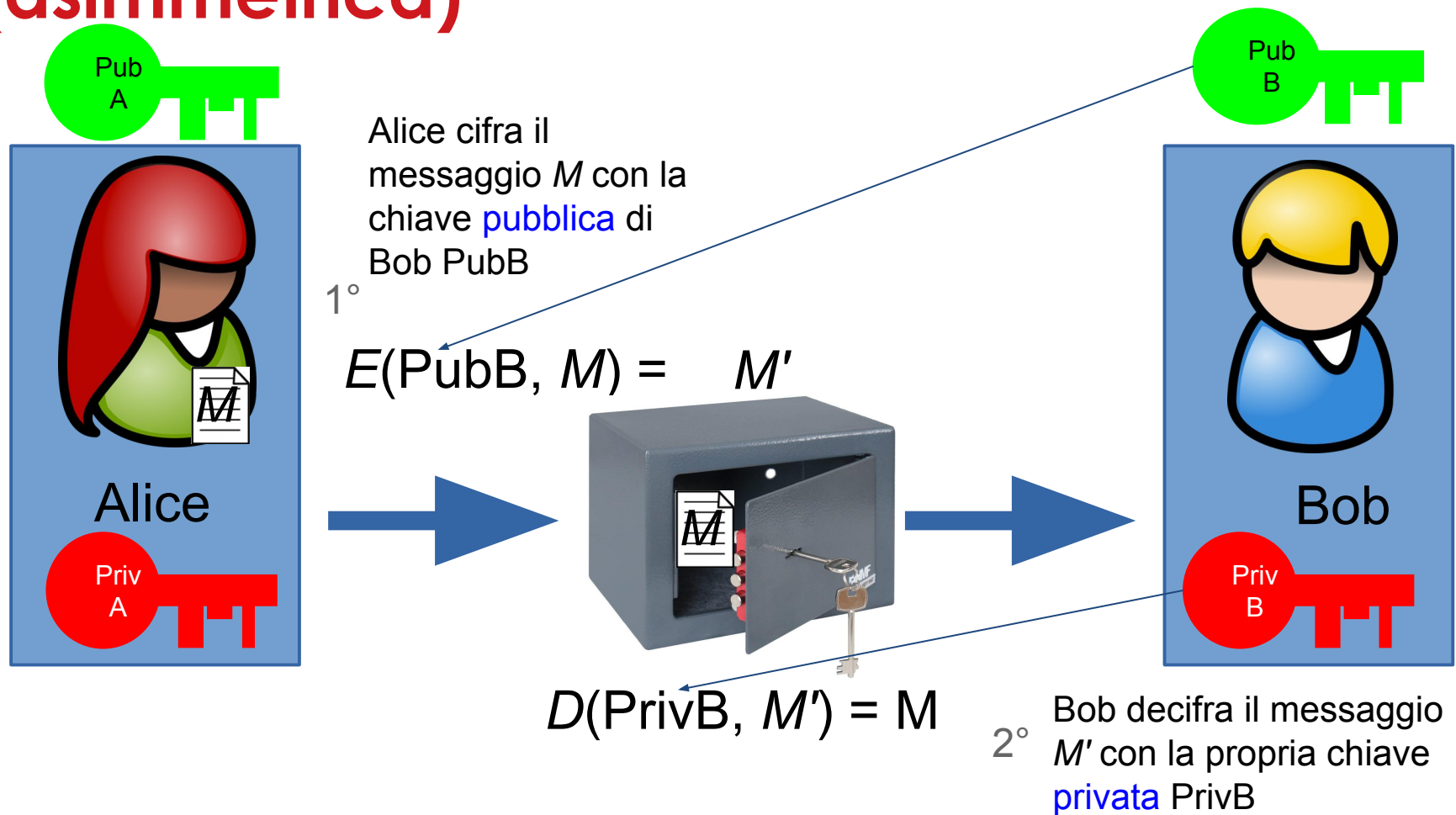
Crittografia Asimmetrica

Crittografia a chiave pubblica (asimmetrica)

- Chiave **pubblica** -> viene resa disponibile a chiunque
- Chiave **privata** -> l'utente deve custodirla gelosamente e non comunicarla a nessuno



Crittografia a chiave pubblica (asimmetrica)



Applicazioni

- **Autenticità**

- Validare la sorgente di un messaggio per garantire l'autenticità del mittente

- **Confidenzialità**

- Soggetti non autorizzati non possono accedere al dato

- **Integrità**

- Assicurarsi che il messaggio non sia stato alterato durante la trasmissione, accidentalmente o intenzionalmente

- **Non-repudiabilità**

- Un mittente non può inviare il messaggio e negare successivamente di averlo fatto



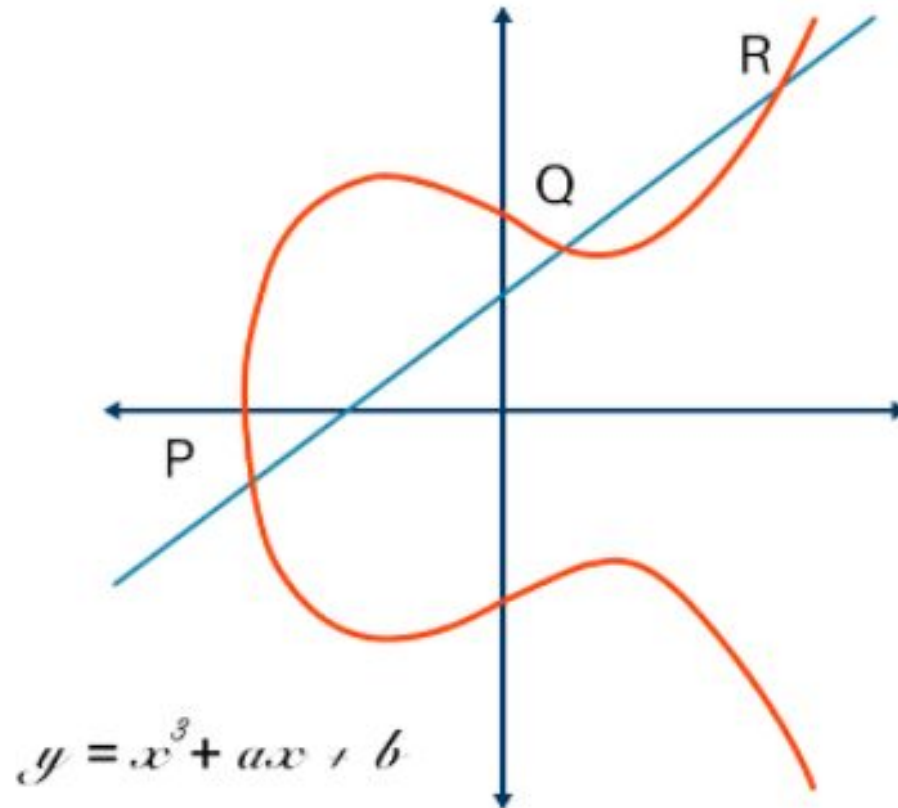
RSA

(Rivest–Shamir–Adleman)

- Uno degli algoritmi di crittografia asimmetrica più usati
- La sicurezza di RSA si basa sulla difficoltà pratica di fattorizzare il prodotto di due grandi numeri primi
 - il “problema della fattorizzazione”
 - fattorizzazione numeri interi ->
decomposizione di un numero intero positivo in un prodotto di numeri interi.
 - Quando i numeri sono sufficientemente grandi, non si conosce un algoritmo (*non quantistico*) efficiente di fattorizzazione di interi



ECDSA (Elliptic Curve Digital Signature Algorithm)

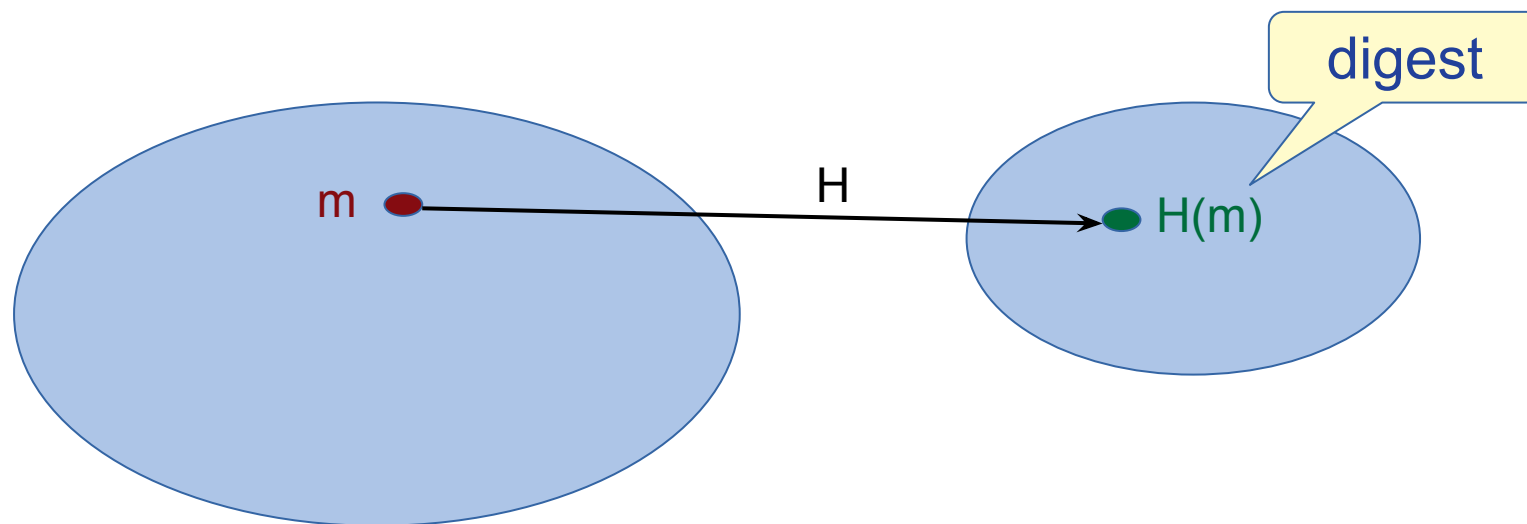




ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Hashing Functions

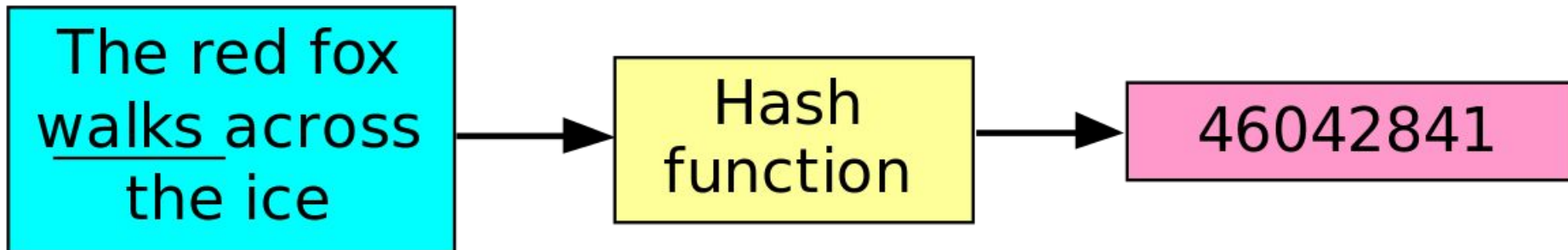
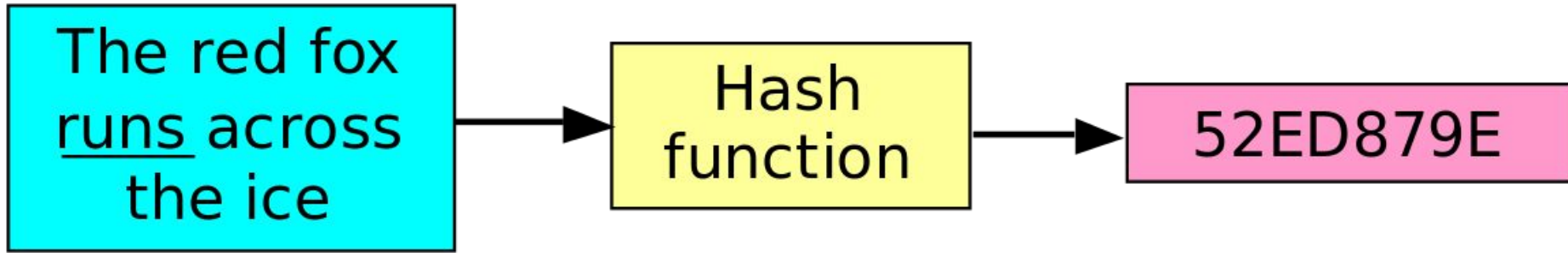
Funzione Hash



Messaggio M -----> $\text{hash}(M)$ -----> **digest**

Funzione hash -> utilizzata per *mappare*
dati di dimensioni arbitrarie in **valori di dimensioni fisse**

Hash Functions

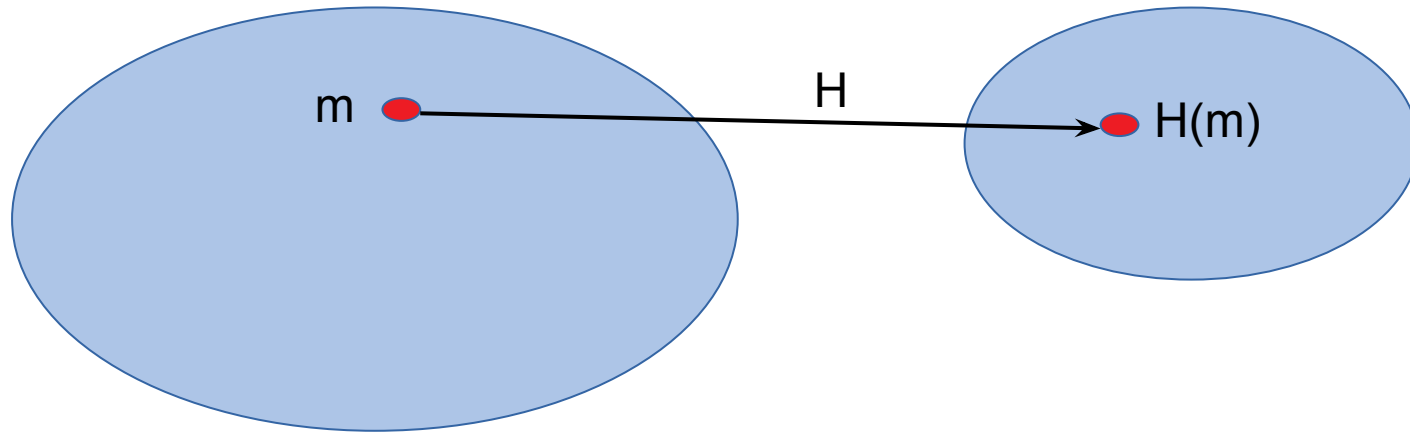


Cryptographic Hash Functions

- Classe speciale di funzioni hash
 - In queste diapositive quando ci riferiamo a "funzione hash", intendiamo in realtà una "funzione hash crittografica".
- Applicazioni
 - Verifica dell'integrità di messaggi e file
 - Firma digitale
 - Verifica della password
 - Proof-of-work



(Cryptographic) Hash Functions



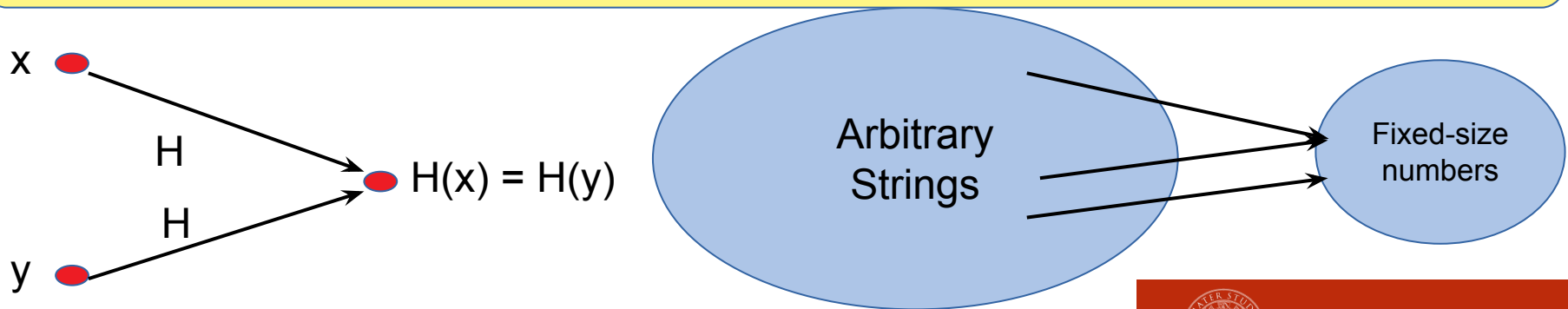
- Computationally efficient
- Security properties
 - Collision-free
 - Hiding
 - Crypto-puzzle friendly

1) Hash Property: Collision-free

~~find~~
Cannot have collisions

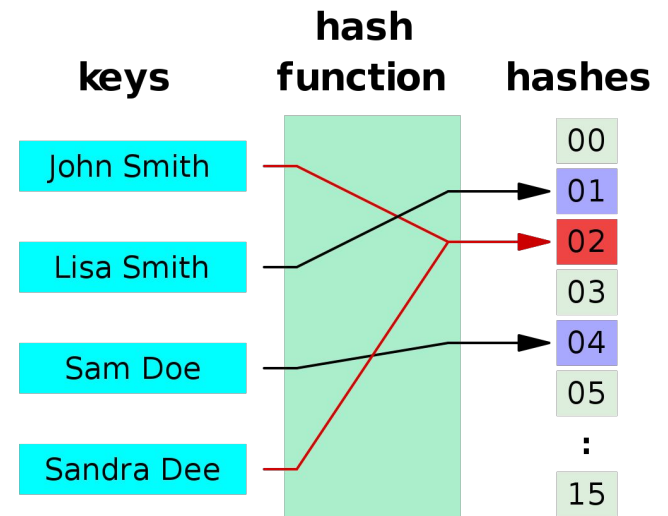
Pigeon principle: se n oggetti sono messi in m contenitori, con $n > m$, allora almeno un contenitore deve contenere più di un oggetto
Le collisioni esistono!

Non si possono trovare x e y tali che $x \neq y$ e $H(x) = H(y)$



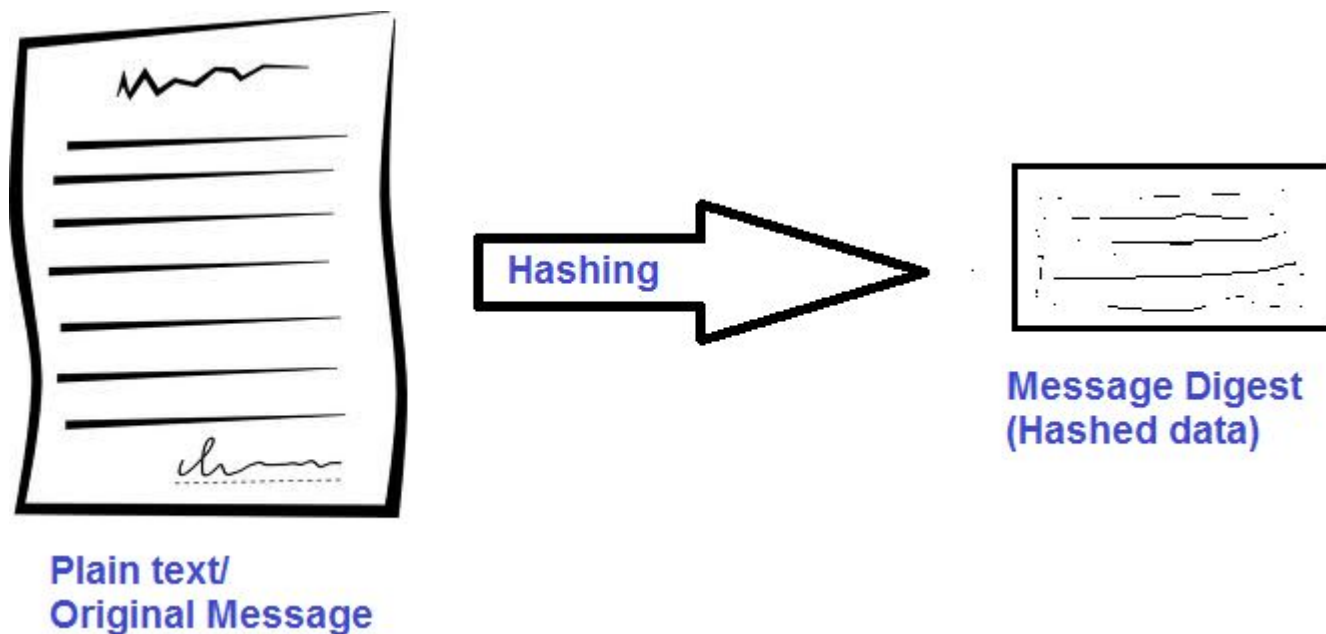
1) How to find a collision

- If output is 256 bits, then we have 2^{256} possible output values
- You need to **guess**
- If we pick 2^{130} randomly chosen input 99.8% chance that two of them will collide
 - This works no matter what H is ...
 - ... but it takes too long to matter

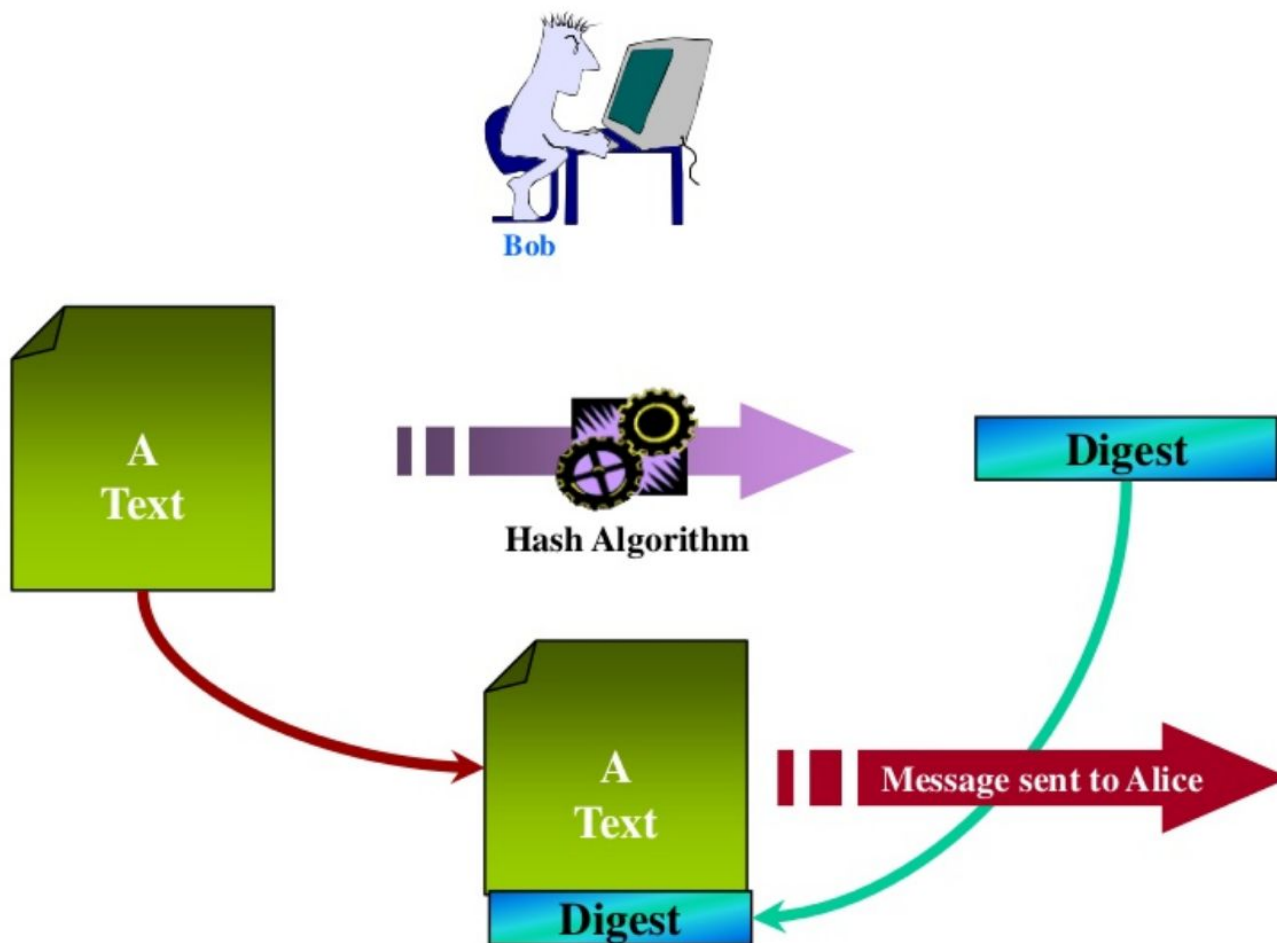


1) Applicazione: Hash Digest Identification

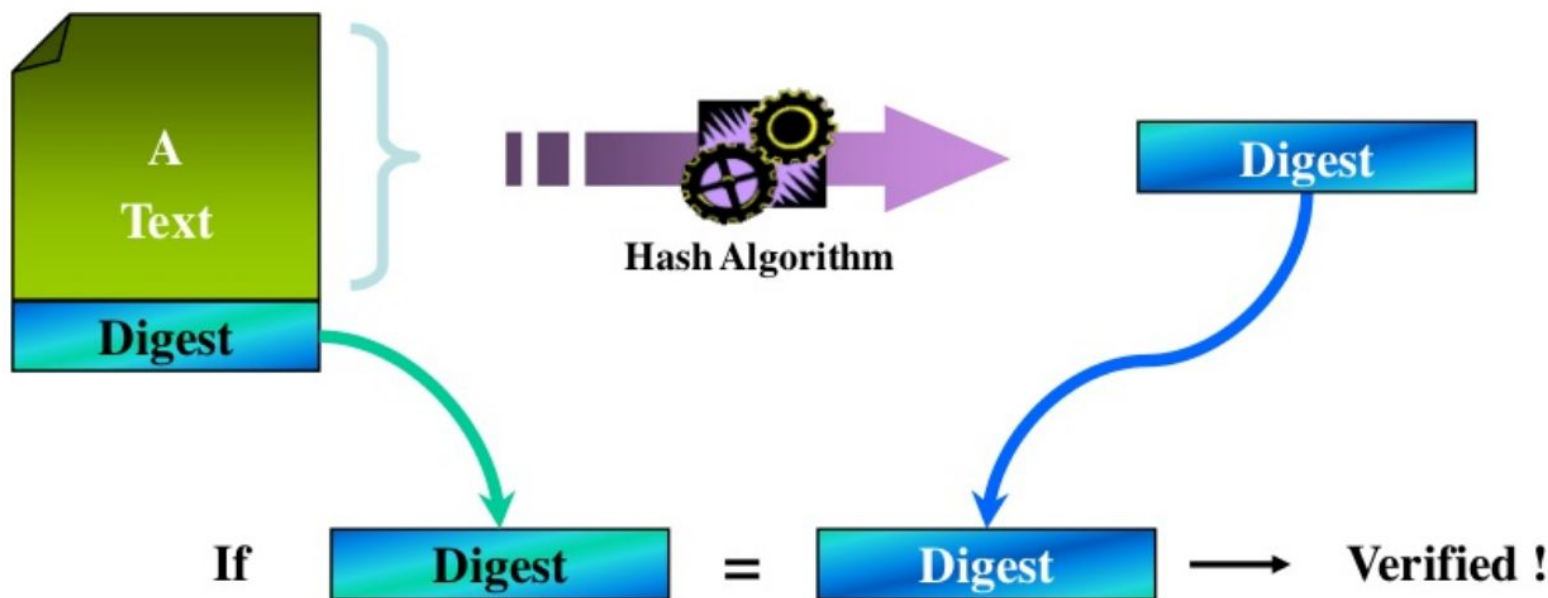
- Se conosciamo $H(x) = H(y) \rightarrow$ possiamo assumere che $x = y$
- Per riconoscere un file, basta ricordare il suo hash
 - Utile perché l'hash è piccolo come dimensione (256 bit)



1) Applicazione: Integrità dei dati



1) Applicazione: Integrità dei dati - Verifica



2) Hash Property: Hiding

- Dato $H(x)$ non deve essere facile trovare x



2) Hash Property: Hiding

- Dato $H(x)$ non deve essere facile trovare x

$H(\text{moat}) = 98\text{E}2\text{W}0\text{dja}8\text{A}...\text{aslOSa}216\text{F}3$

$H(\text{m}\text{aot}) = \text{E}6712\text{e}3\text{awa}4...\text{gz}3\text{wle}3\text{A}9\text{C}9$



2) Hash Property: Hiding

- Dato $H(x)$ non deve essere facile trovare x

$H(\text{moat}) = 98\text{E}2\text{W}0\text{dja}8\text{A}...\text{as}|\text{O}\text{Sa}216\text{F}3$

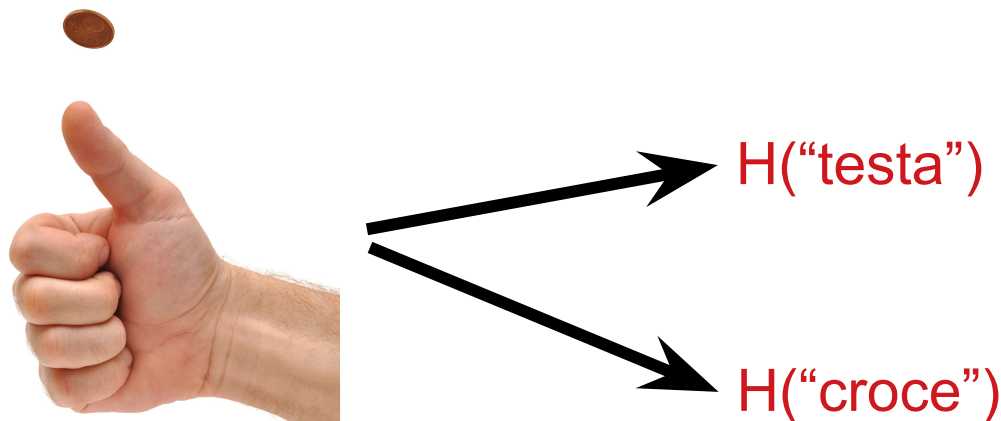
$H(\text{m}\text{a}\text{o}\text{t}) = \text{E}6712\text{e}3\text{awa}4...\text{gz}3\text{wle}3\text{A}9\text{C}9$



la lunghezza del digest è sempre uguale (256 bits per SHA256)



2) Esempio in cui Hiding fallisce



- Guardando i risultati dell'hash, è facile capire se x era una croce o una testa
 - Solo due input!
 - Basta fare l'hash dei due input e guardare il risultato dell'hash

2) Hash Property: Hiding

- Funziona solo quando abbiamo
 - Un **grande insieme** di possibili valori di x
 - Non ci sono particolari x che sono **più probabili**
 - Non ci sono particolari x che sono **più interessanti di altri**

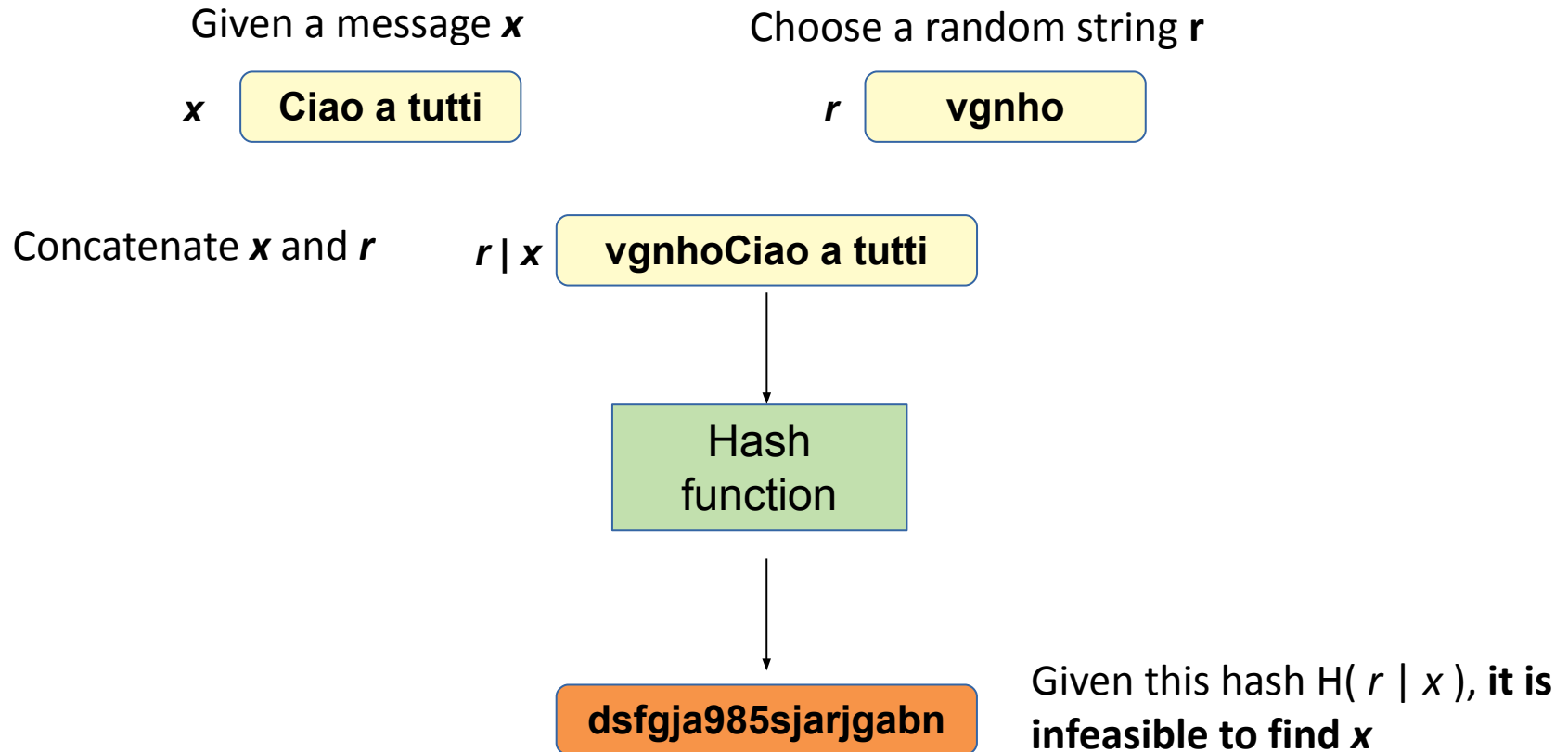


2) Applicazione: Commitment

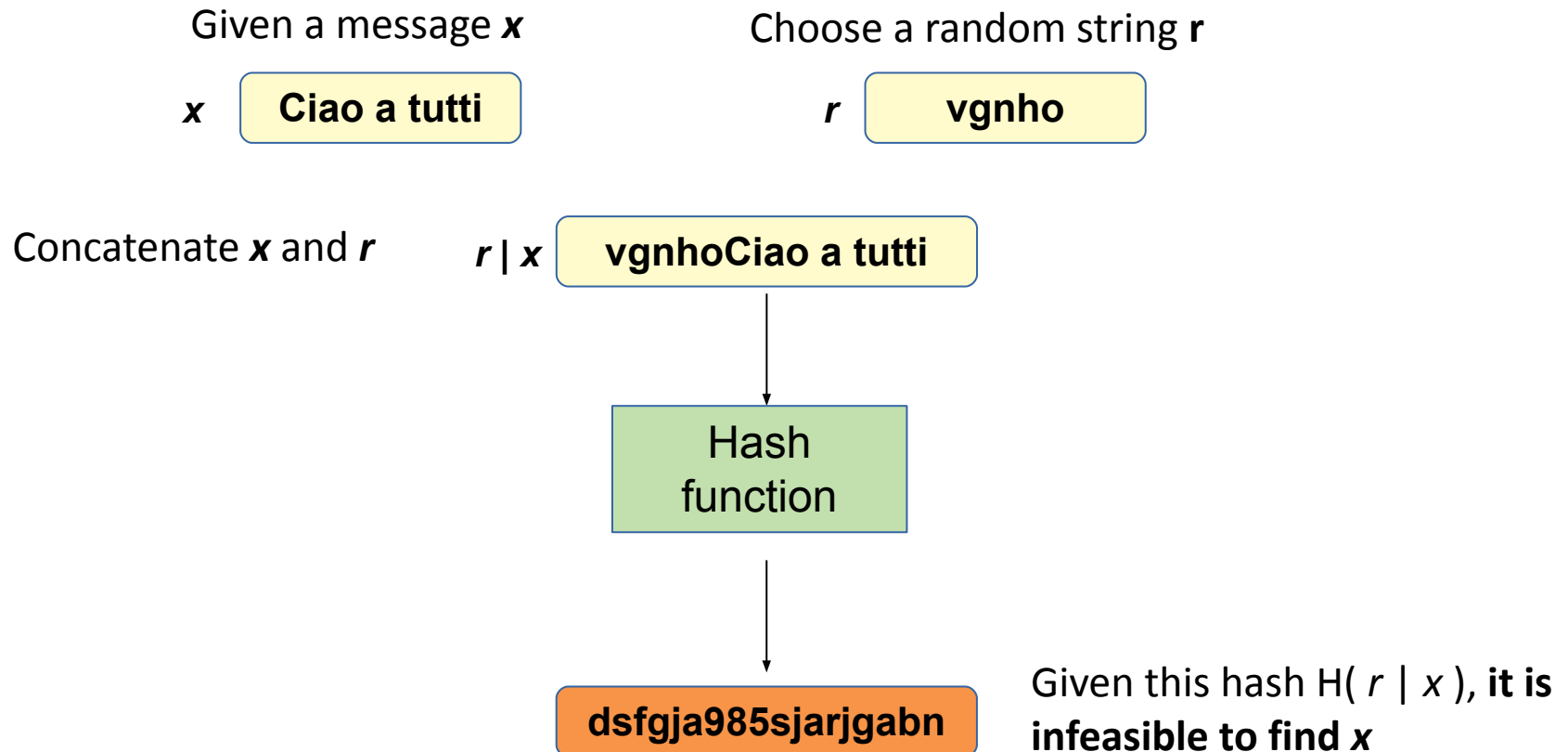
- Vuoi vincolarti a un valore, rivelarlo più tardi
 - "sigillare un valore in una busta" ora, e
 - "aprire la busta" più tardi



3) Hash Property: Crypto-puzzle friendly



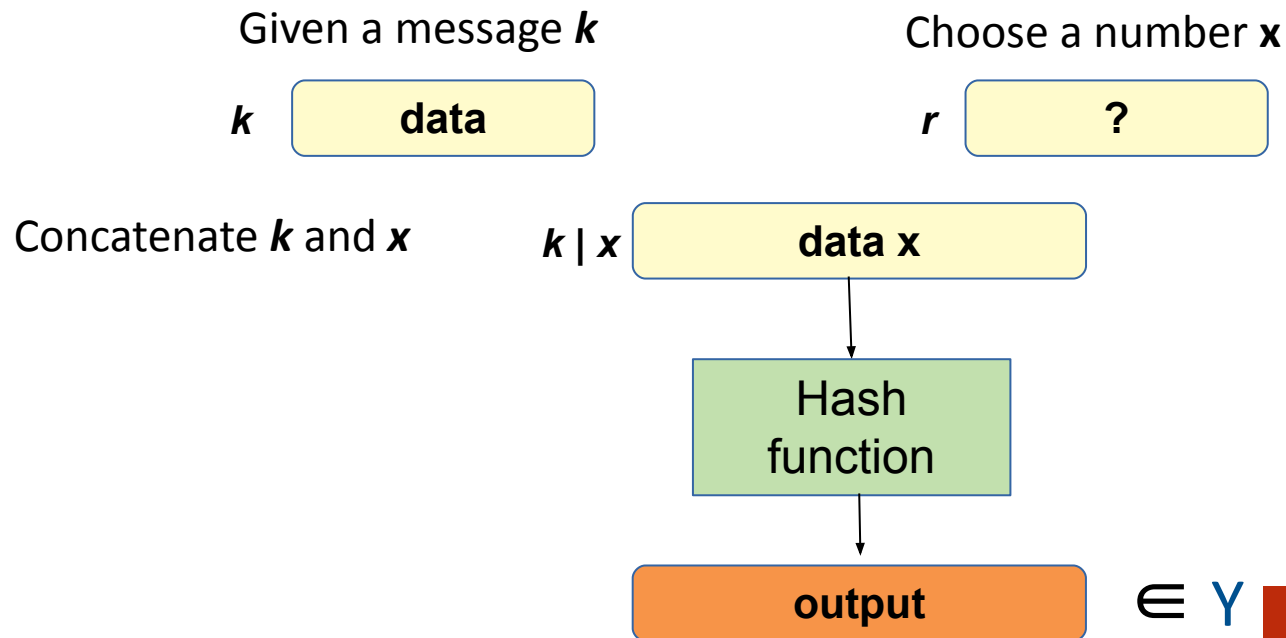
3) Hash Property: Crypto-puzzle friendly



This can be used to devise crypto-puzzles

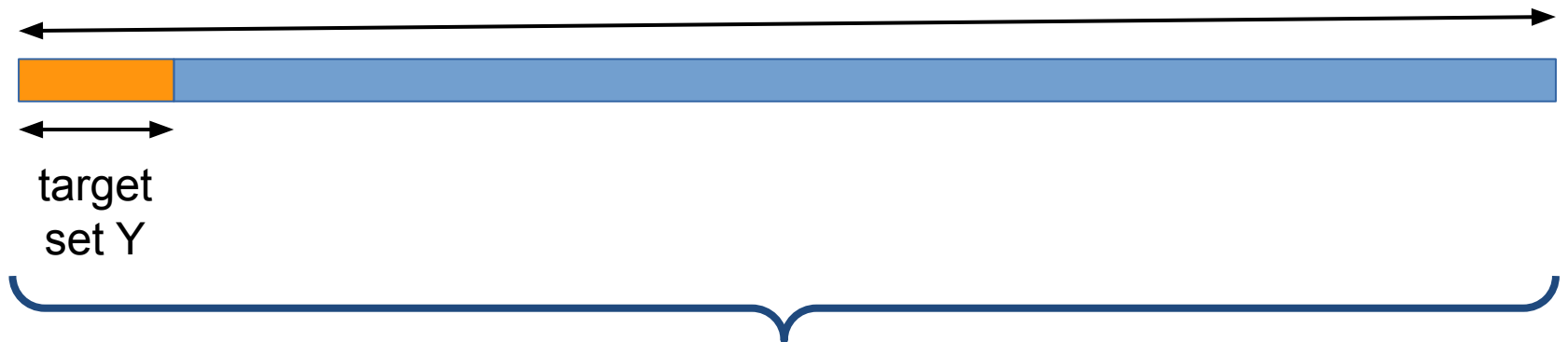
3) Applicazione: Search puzzle

- Given a puzzle **k** and a **target set Y**,
- try to find a solution **x** such that:
- $H(\text{**k**} \mid \text{**x**}) \in Y$



3) Applicazione: Search puzzle

- Given a puzzle **k** and a **target set Y**,
- try to find a solution **x** such that:
- $H(k \mid x) \in Y$

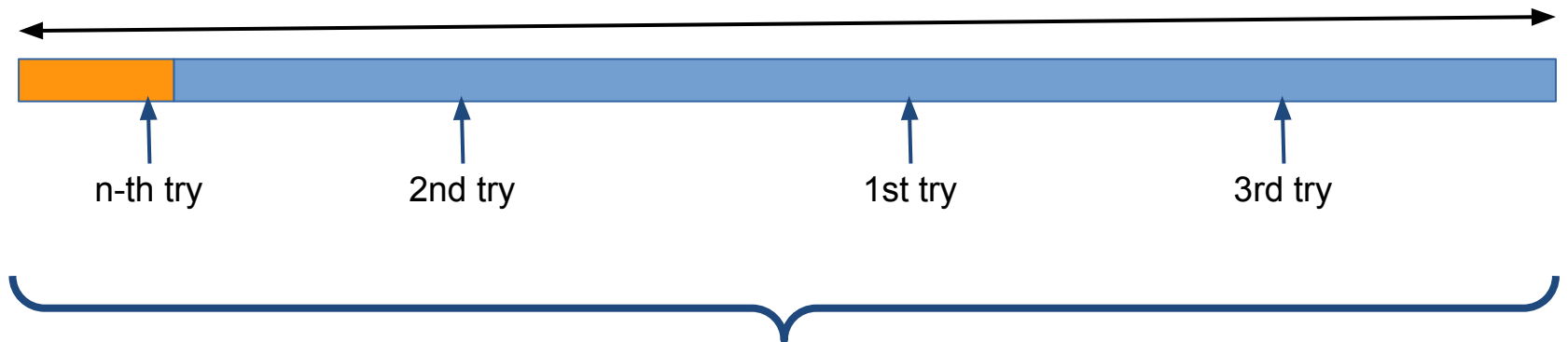


It is like a coin toss, every time you randomly fall in a point
within this set



3) Applicazione: Search puzzle

- Given a puzzle **k** and a **target set Y**,
- try to find a solution **x** such that:
- $H(k \mid x) \in Y$

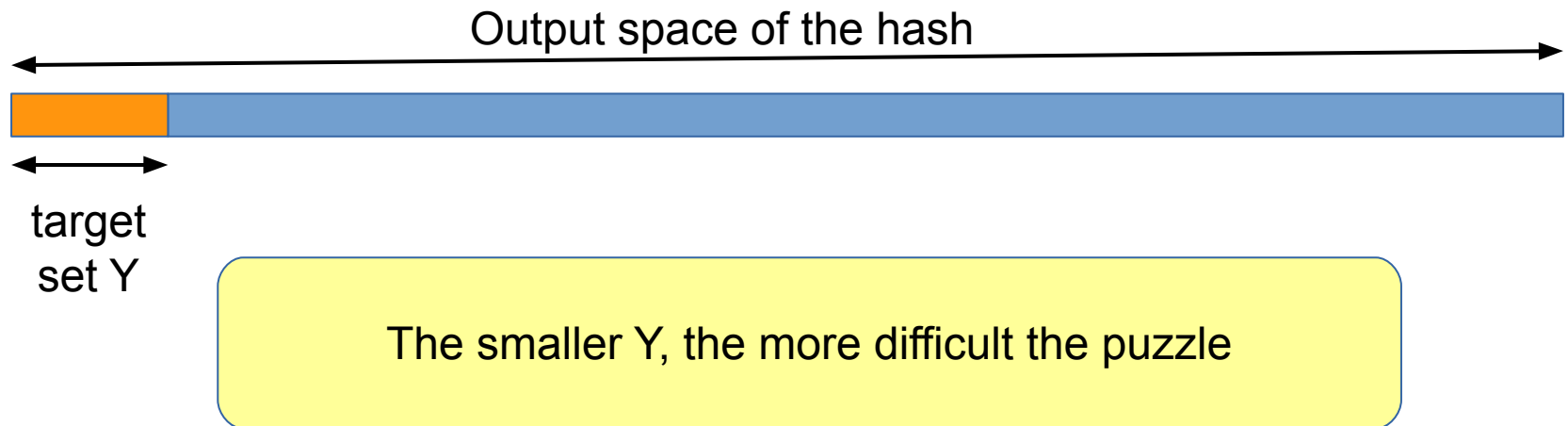


It is like a coin toss, every time you randomly fall in a point within this set



3) Applicazione: Search puzzle

- Given a puzzle k and a **target set** Y ,
- try to find a solution x such that:
- $H(k \mid x) \in Y$



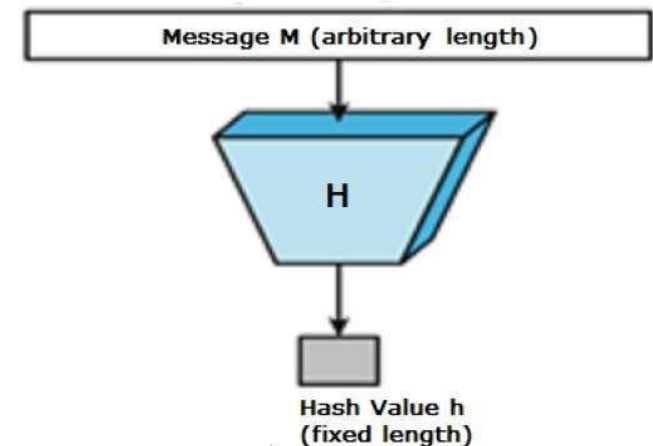
Hash Functions: Existing Schemes

Best-known cryptographic hash functions:

MD5 (128 bits)

SHA-1 (160 bits)

SHA-256/384/512 (256/384/512 bits)



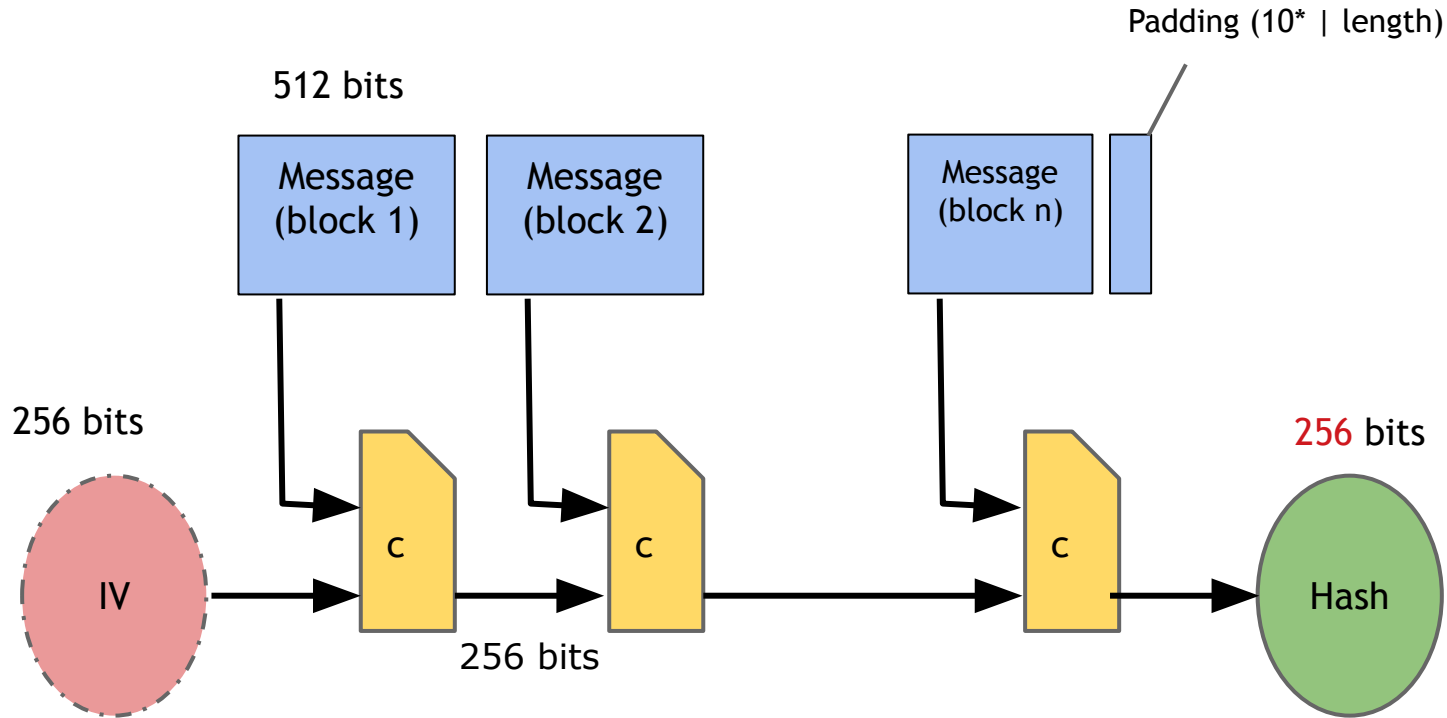
Security:

- **MD5:** A 2013 attack by Xie Tao, Fanbao Liu, and Dengguo Feng breaks MD5 collision resistance in 2^{18} time. This attack runs in less than a second on a desktop computer.
- **SHA-1:** Some theoretical attacks - not yet collisions

Suggested to use SHA-2 (256 or 512) or SHA-3/Keccak



SHA-256 Hash Function



- IV: Inizialization vector
(number used for every call to the function)
- c: compression function
- If c is collision free, then SHA-256 is collision free

Example (SHA-256)

msg1.txt

```
All work and no play makes Jack a dull boy  
All work and no play makes Jack a dull boy  
All work and no play makes Jack a dull boy
```

sha256

5f10e43e591ed245374fae017f8c11e429f6bc6ebf42f2d1d75fb4d6e39b8f3b

msg2.txt

```
All work and no play makes Jack a dull boy  
All work and no play makes jack a dull boy  
All work and no play makes Jack a dull boy
```

sha256

369c932a24add019689c3896657b4c625dc7864d4959aaccaffa2b75254e955b



Example (SHA-256)

Terminale

sferrett@steach: ~

```
sferrett@steach:~$ echo "ciao" | sha256sum
6f0378f21a495f5c13247317d158e9d51da45a5bf68fc2f366e450deafdc8302 -
sferrett@steach:~$ echo "miao" | sha256sum
549d63fbe569a4dcb200dec0d65786794c931d4a976e8644f1035695117b20e9 -
sferrett@steach:~$ echo "ciao mare" | sha256sum
079542b606786b70eb79bed391e8672331bdf6f09fa0c5e1349cdd0faa768687 -
sferrett@steach:~$ echo "ciao care" | sha256sum
a945dd5628c2864e297162adf2b400d8d04f3088582aae03ccea5a054c5e5a26 -
sferrett@steach:~$ █
```



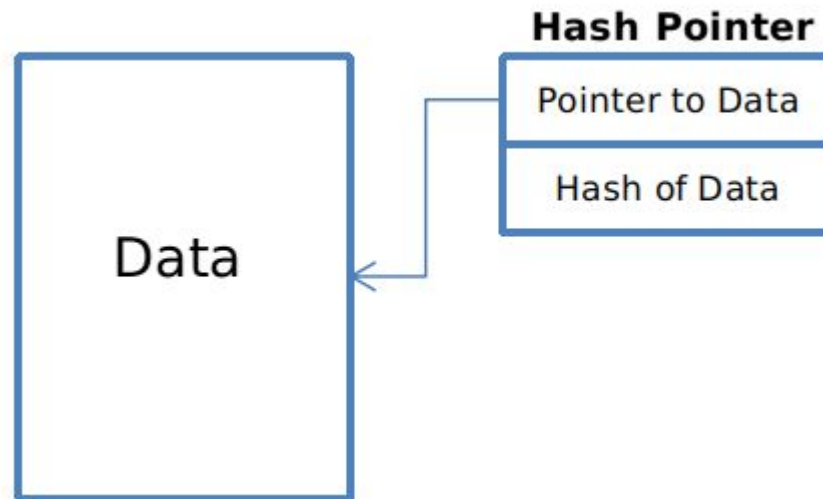


ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Hash Pointers and Data Structures

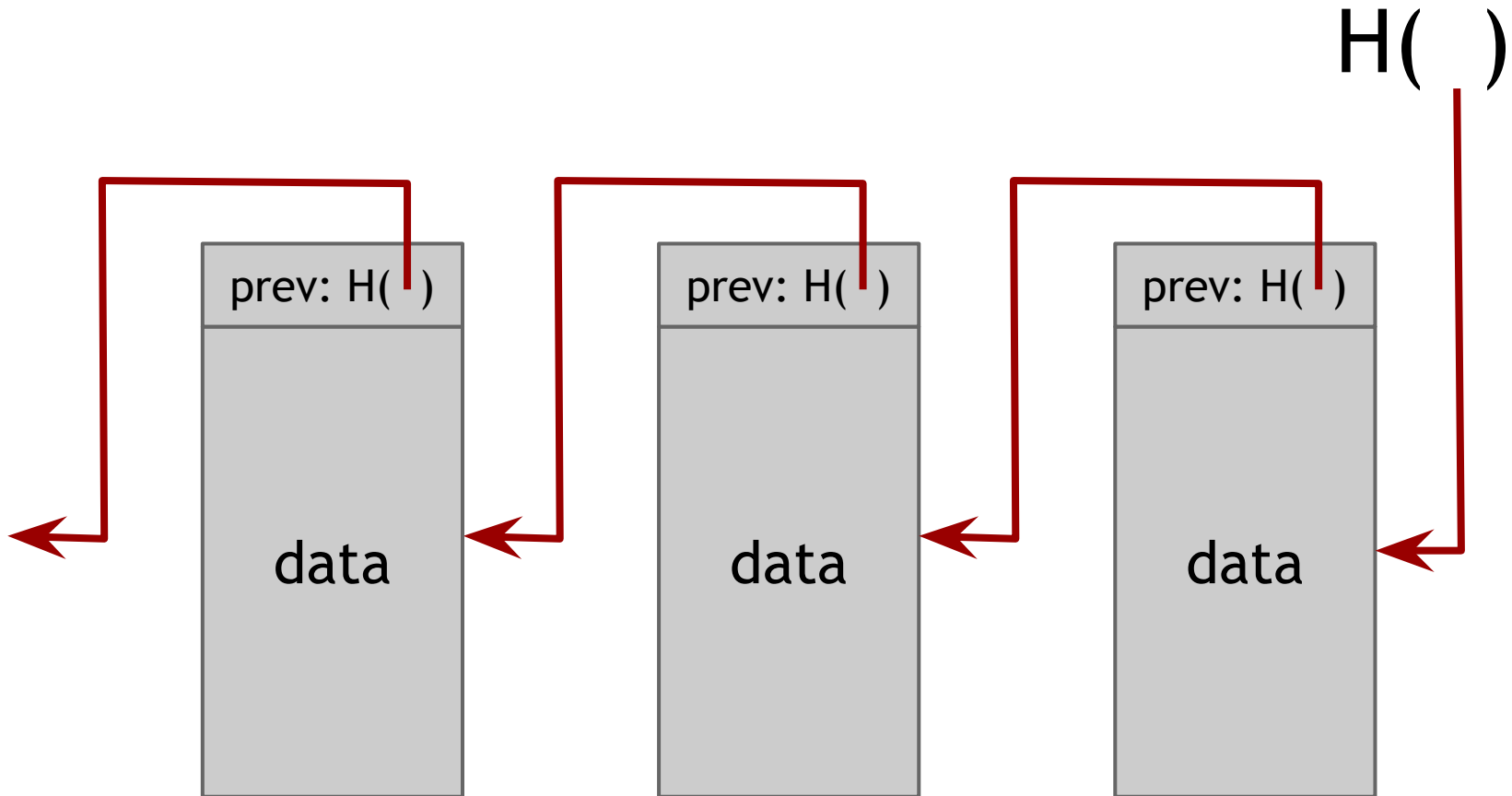
Hash Pointer

- **Hash Pointer** is comprised of two parts:
 - **Pointer** to where some information is stored
 - used to get the information
 - **Cryptographic hash** of that information
 - used to verify that information hasn't been changed



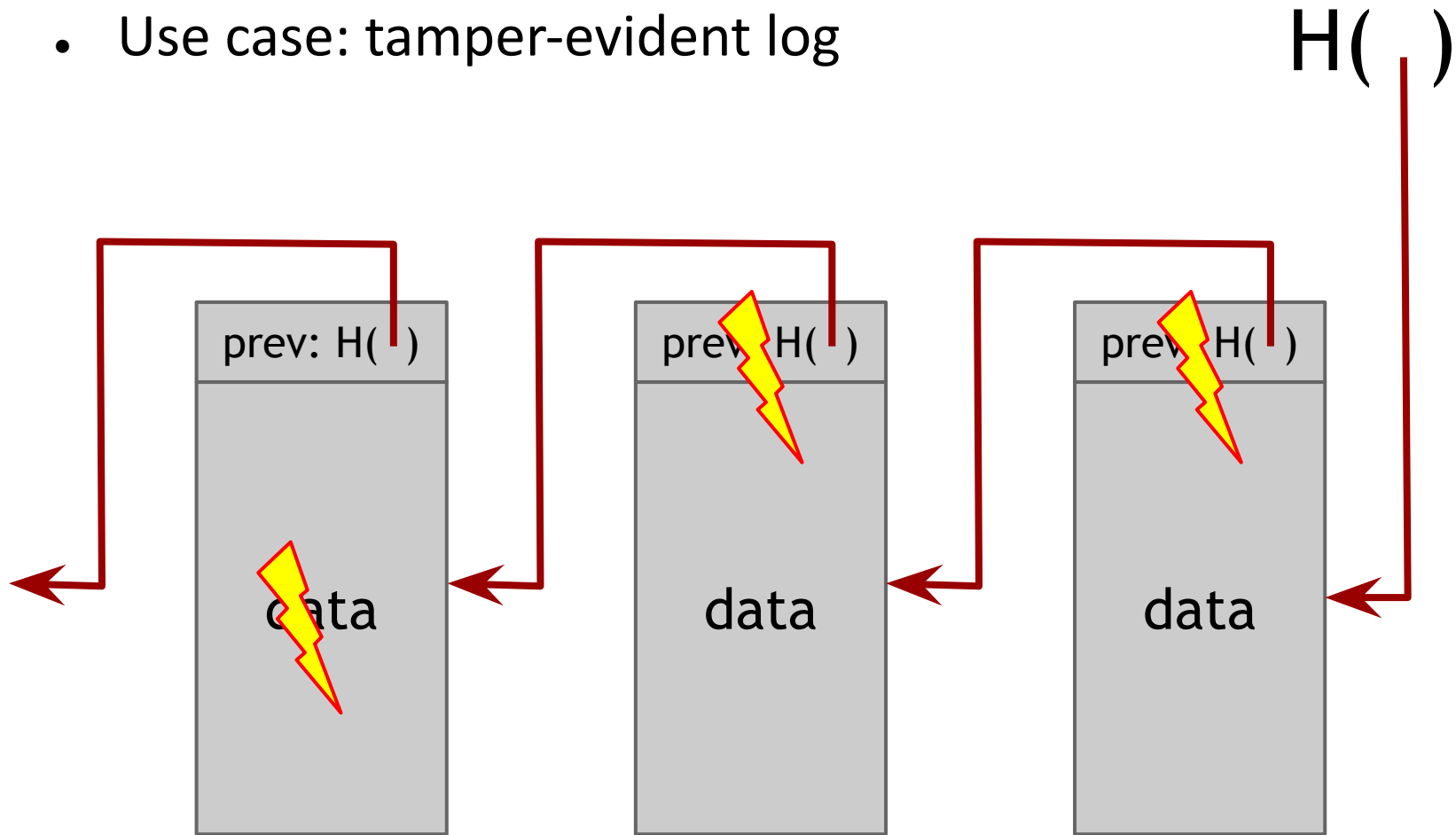
Key Idea

- Build data structures with hash pointers



Detecting Tampering

- Use case: tamper-evident log

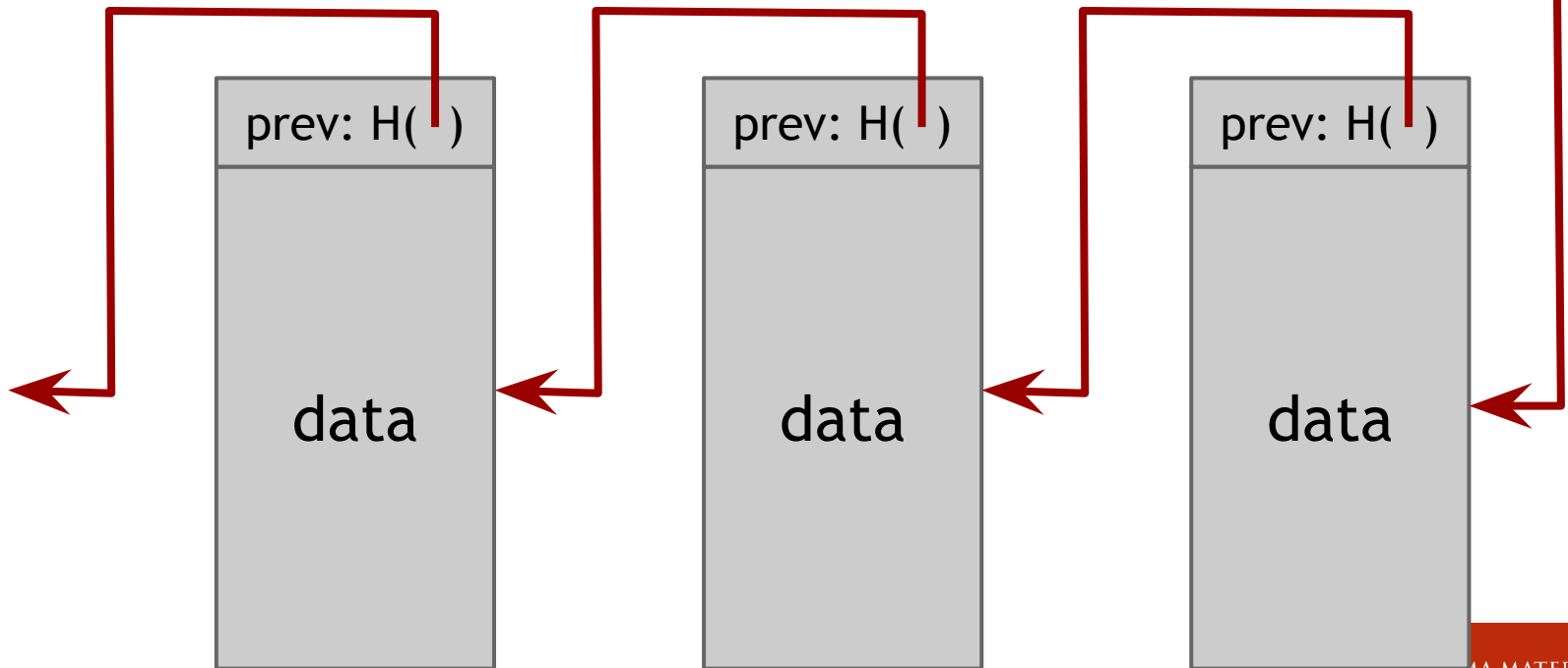


Proving membership in a chain of Hash Pointers

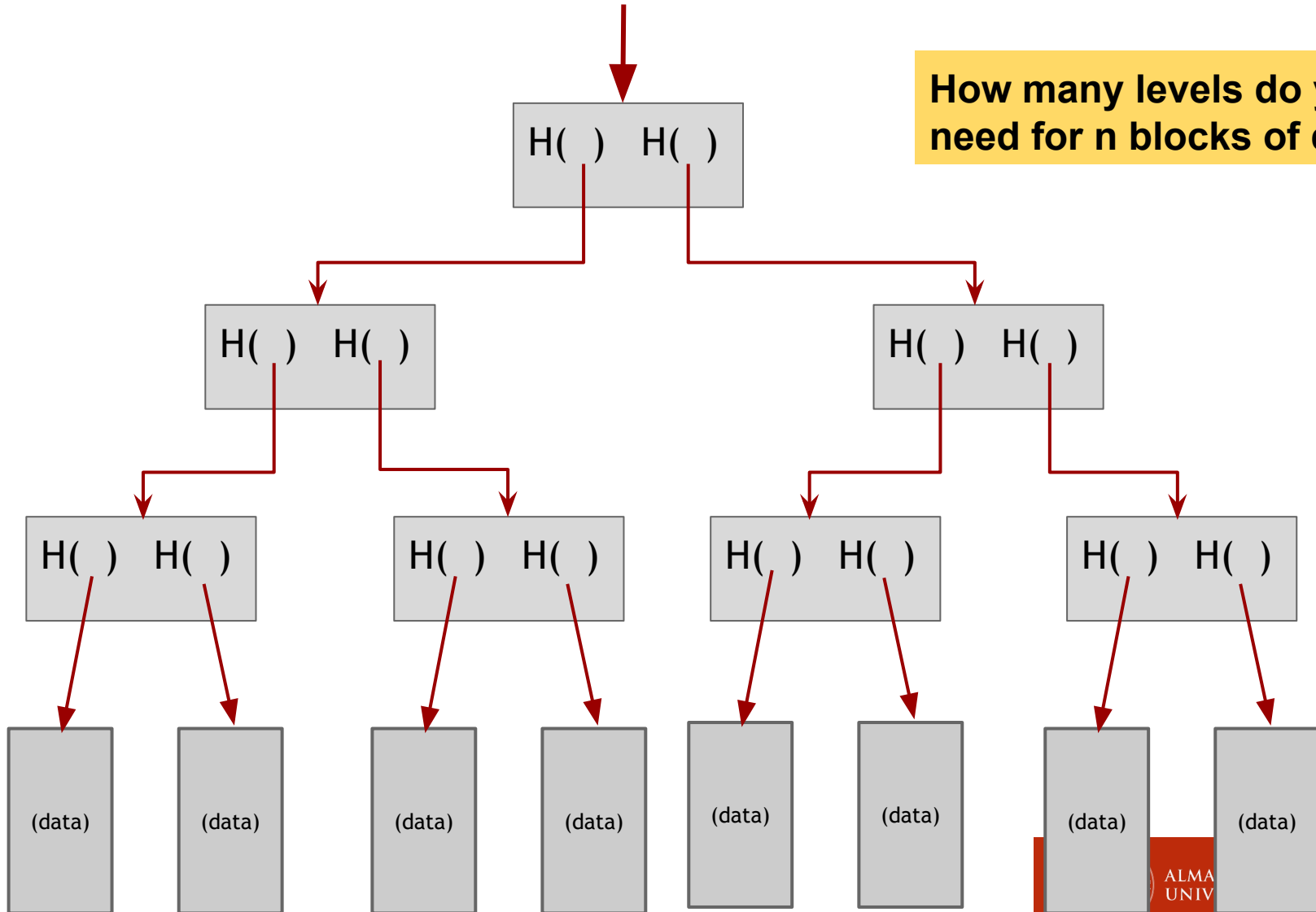
Given n items, how much does it cost to prove membership of an item in a hash pointer structure?

$O(n)$

$H()$

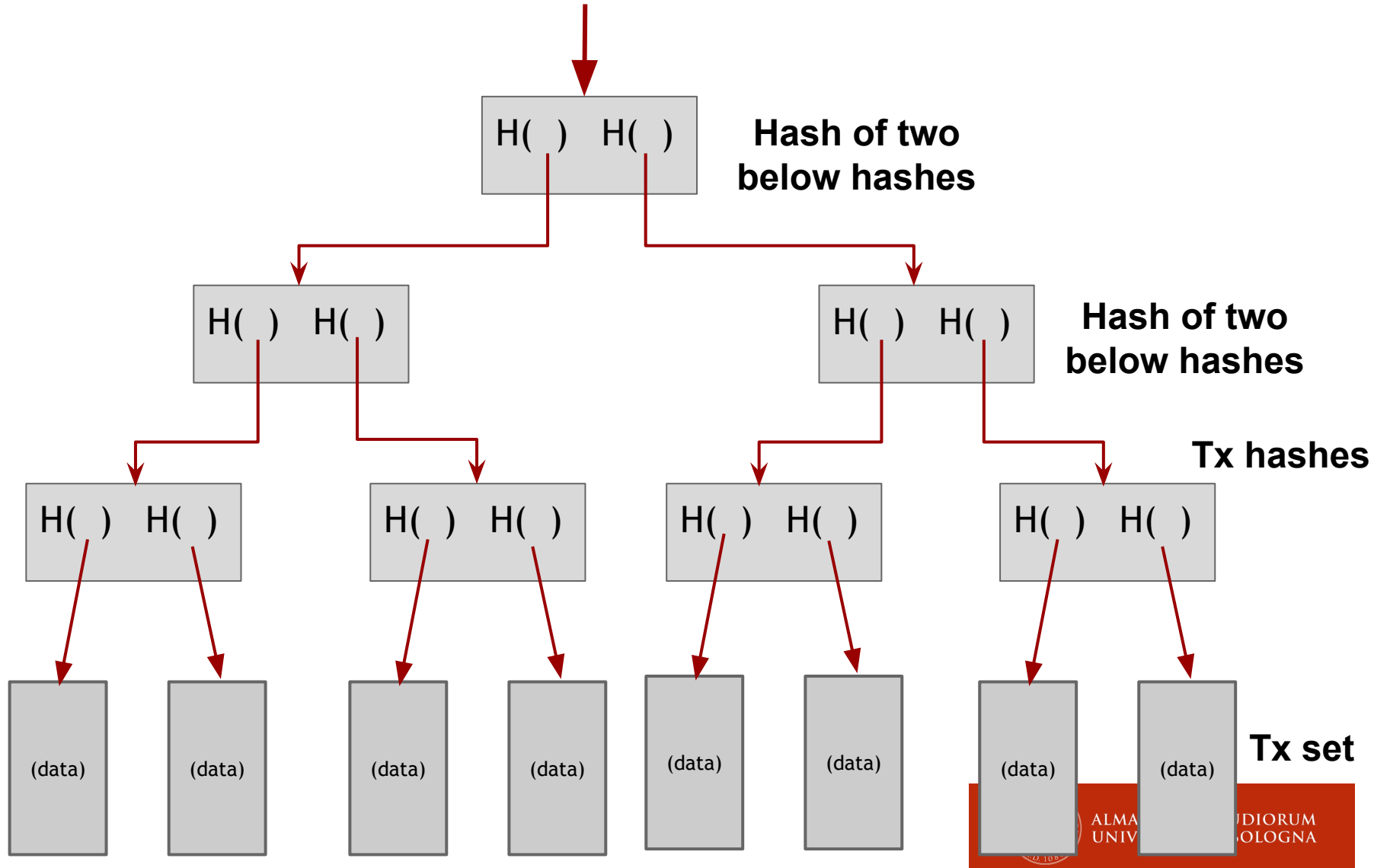


Binary tree with hash pointers = “Merkle tree”

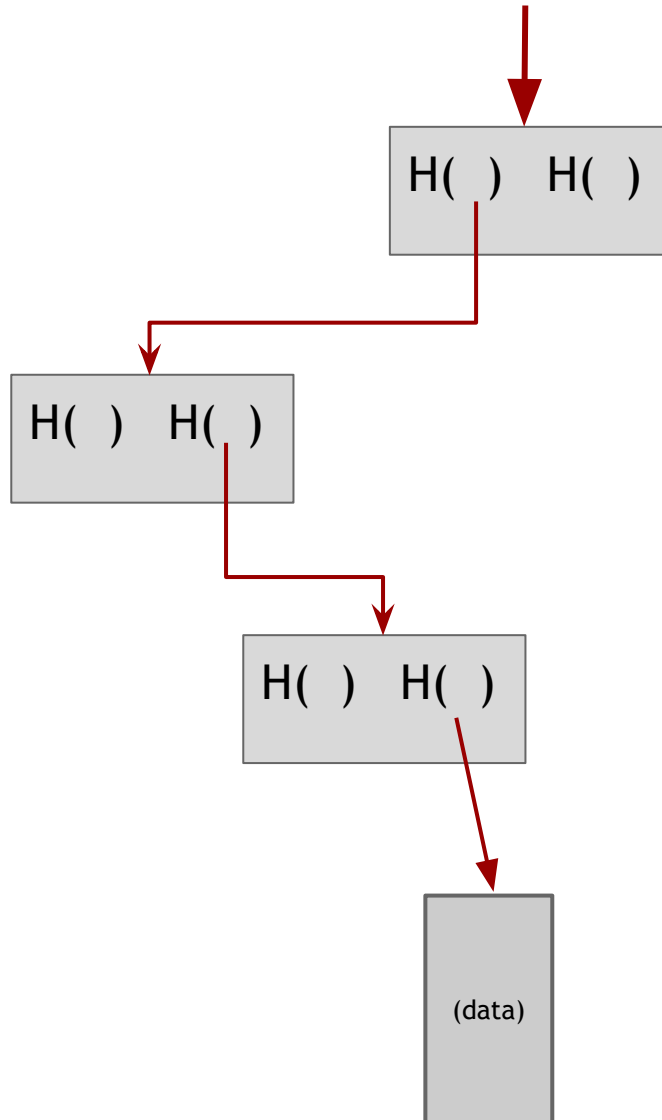


How many levels do you need for n blocks of data?

Binary tree with hash pointers = “Merkle tree”



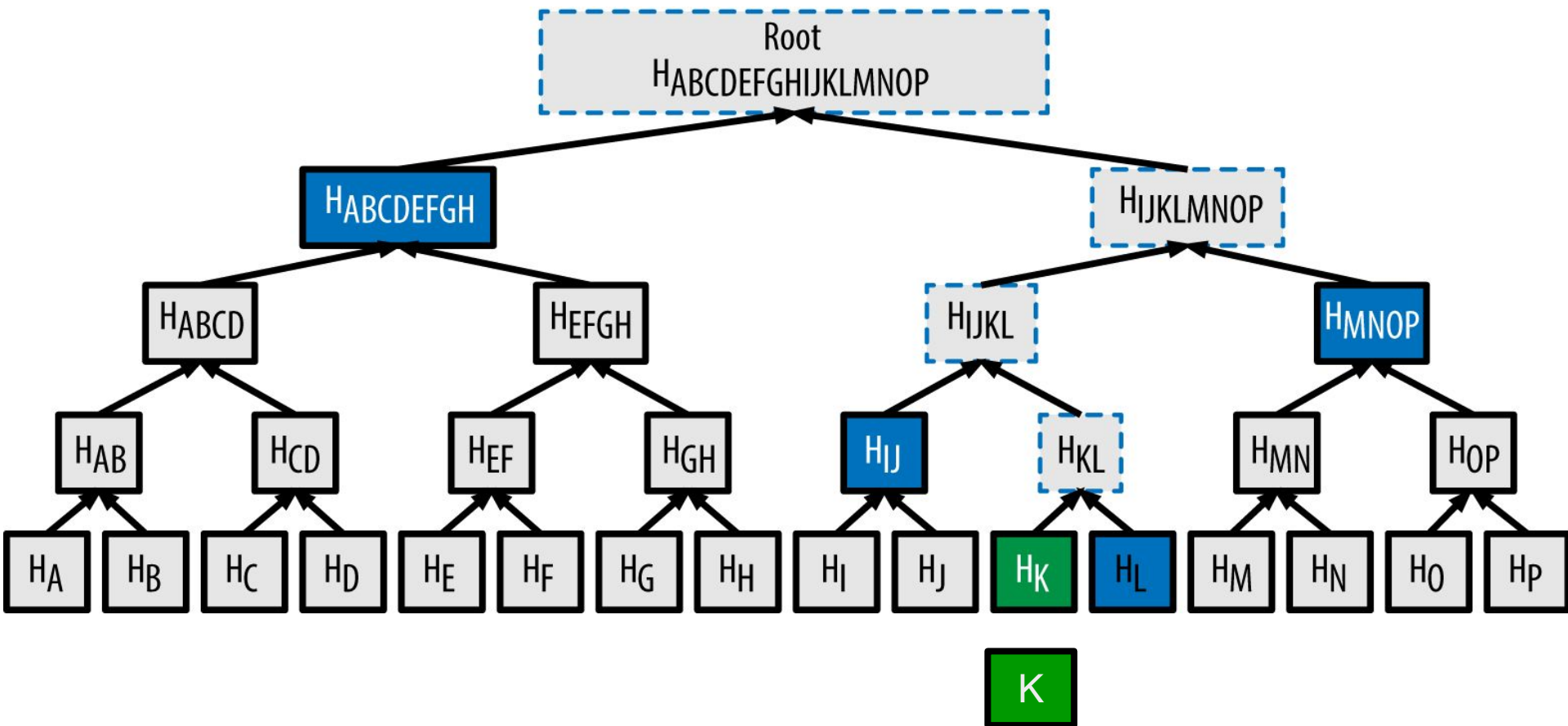
Proving membership in a Merkle tree



show $O(\log n)$ items

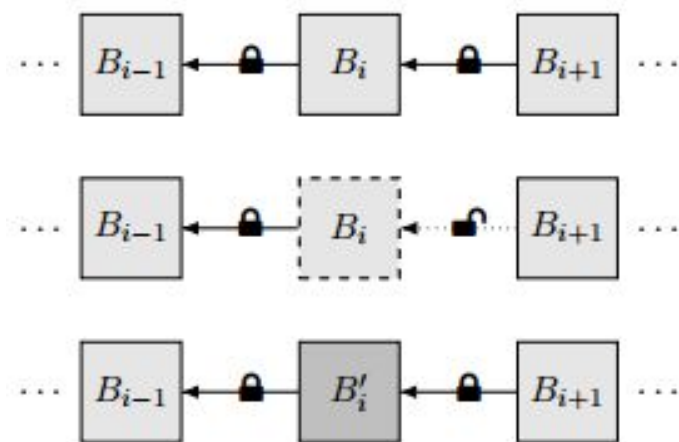


Proving membership in a Merkle tree



Chameleon hashing

- Chameleon hashing uses (trapdoor one-way) hash functions allied with a **public key** , **private key** pair.
- **Public key** -> can be used to measure hash function by anyone easily. The collision of two hashes in the hash function is impossible for the one who doesn't know the private key.
- **Private key** -> However, who knows the private key information can find the collision of the given inputs.
- This allows to change the original hash input with another “colliding” input





ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Non-Ripudio nella Sicurezza Digitale

Non ripudio nella Sicurezza Digitale

- Un servizio che fornisce la prova **dell'integrità e l'origine dei dati**.
- **Un'autenticazione**, a garanzia della genuinità dei dati stessi.



Non ripudio nella Sicurezza Digitale

- L'integrità dei dati -> hash dei dati garantisce una **bassissima probabilità che i dati vengano alterati.**
- L'integrità dei dati deve essere confermata dal destinatario



Verifica nella Sicurezza Digitale

- Il metodo più comune per la verifica dell'origine dei dati è l'utilizzo della **firma digitale** accompagnata da:
- **Certificati digitali** -> una forma di infrastruttura a chiave pubblica da cui dipende la firma digitale.

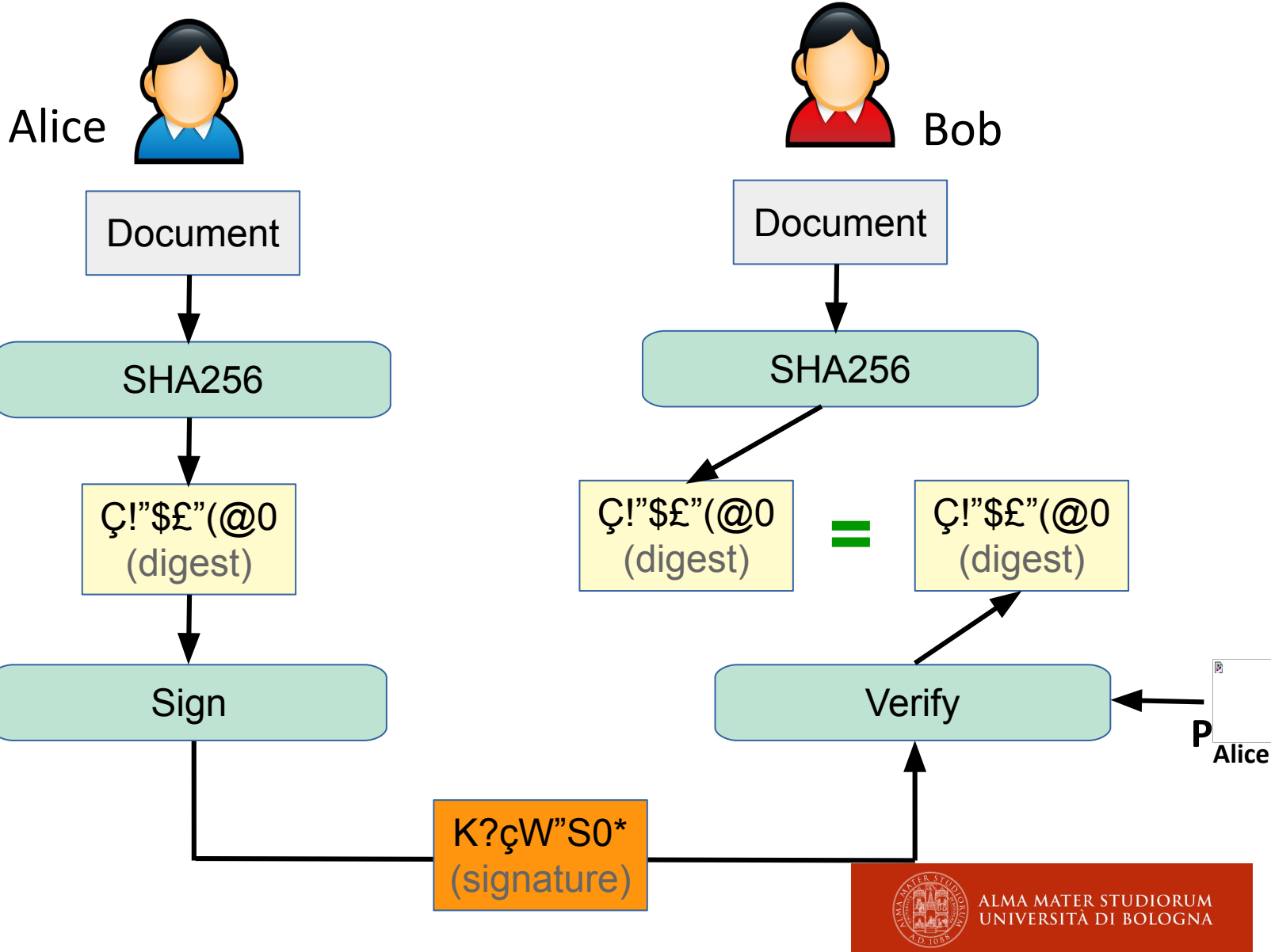


Firma Digitale

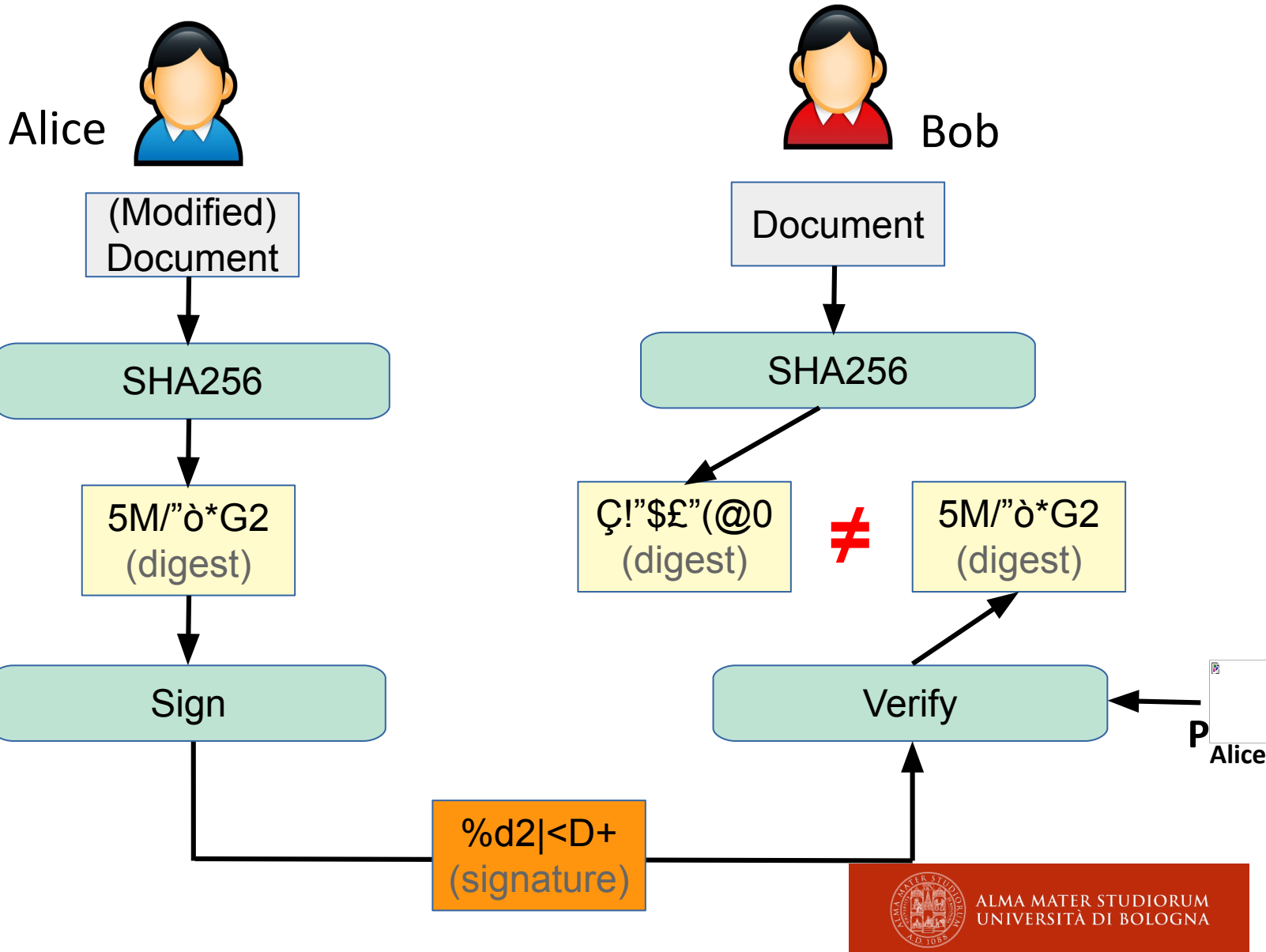
- Schema per la verifica dell'**autenticità** dei messaggi digitali (documenti).
- Impiega la crittografia **asimmetrica**
- **Integrità**: garantisce che il messaggio non sia stato alterato durante il trasporto (utilizzando il digest)
- **Autenticazione**: una firma digitale valida dà al destinatario un motivo molto forte per credere che il messaggio sia stato creato da un mittente conosciuto.



Alice firma un documento per Bob



Alice firma un documento (alterato) per Bob



Certificato Digitale

documento elettronico che attesta l'associazione univoca tra una chiave pubblica e l'identità di una persona



Digital Certificate Name: Alice Address: via Galliera, 3 email: alice@unibo.it
 P Alice



Infrastruttura a chiave pubblica (PKI)

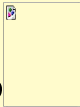
X.509

- **X.509**
 - formato più comune per i certificati digitali
- **Infrastruttura a chiave pubblica X.509 (RFC 5280)**
Public Key Infrastructure (PKI)
 - insieme di processi e mezzi che consentono a **terze parti fidate di verificare e/o farsi garanti** dell'identità di un utente, oltre che di associargli una chiave pubblica
- I certificati X.509 sono utilizzati in molti protocolli Internet, tra cui TLS/SSL, che è alla base di **HTTPS**, il protocollo sicuro per la navigazione in rete.



Certificato Digitale X.509

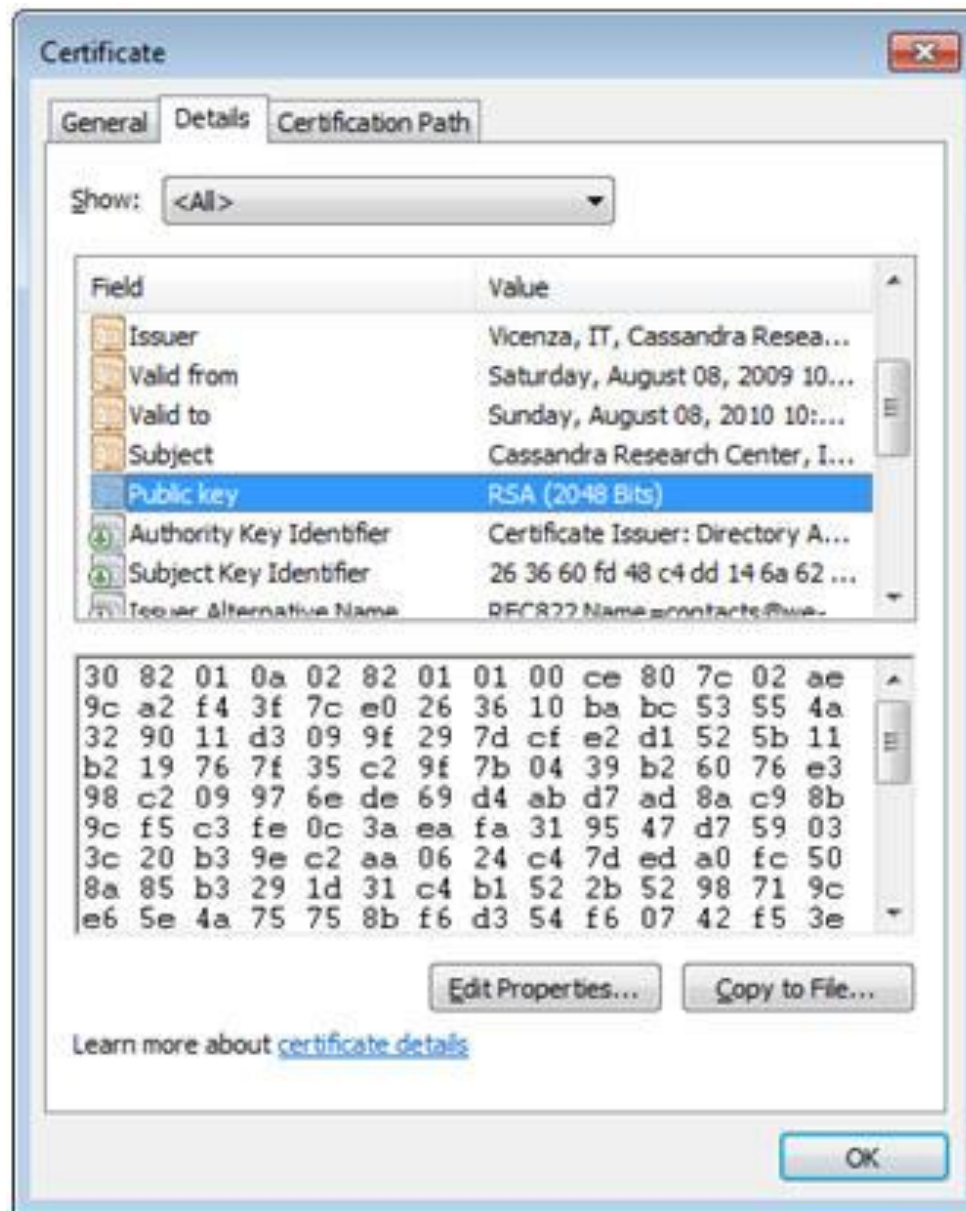


Digital Certificate Name: Alice Address: via Galliera, 3 email: alice@unibo.it
 P Alice
<div>X!çW"SO* (signature)</div>

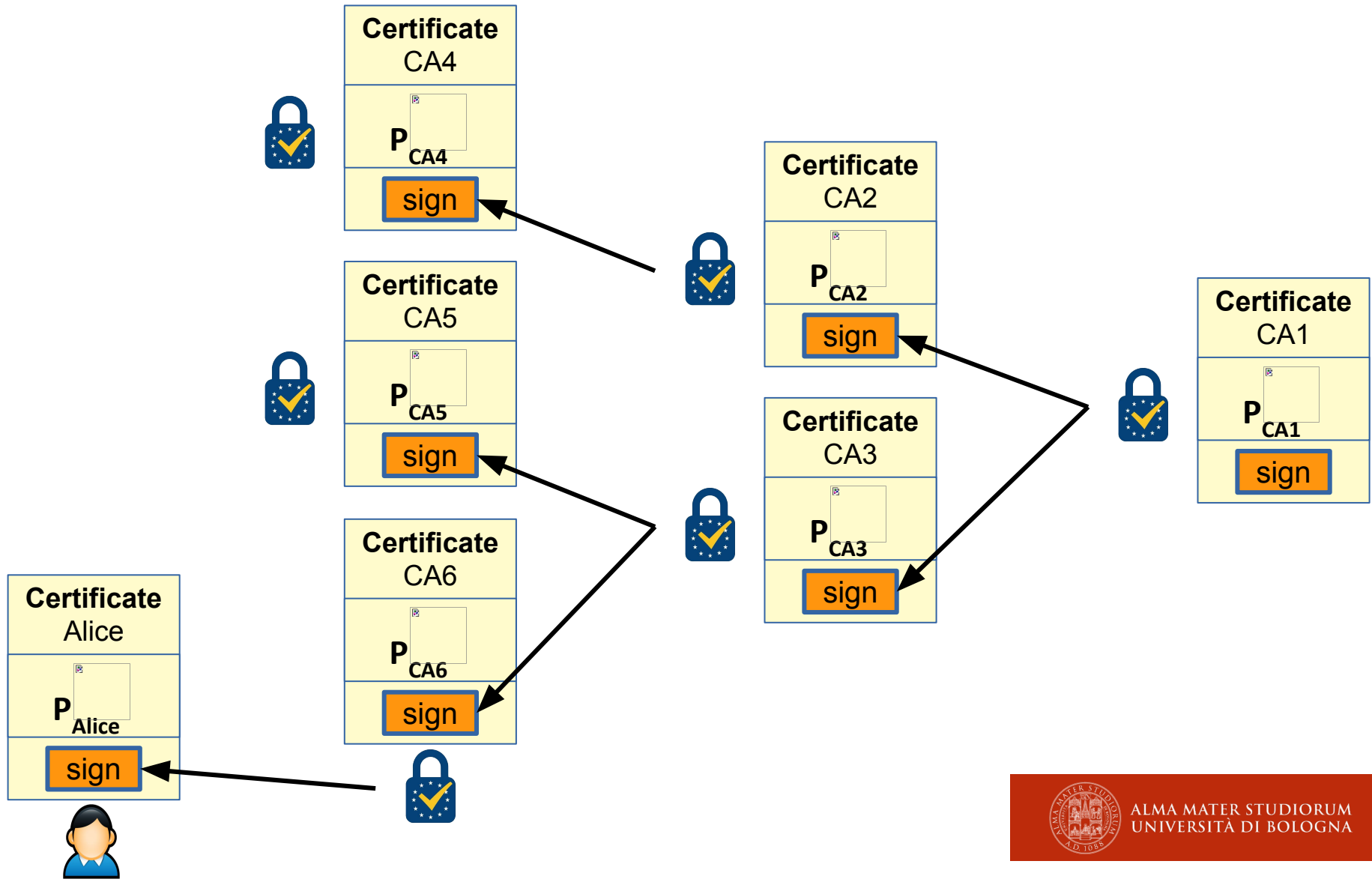
Autorità di Certificazione(CA)



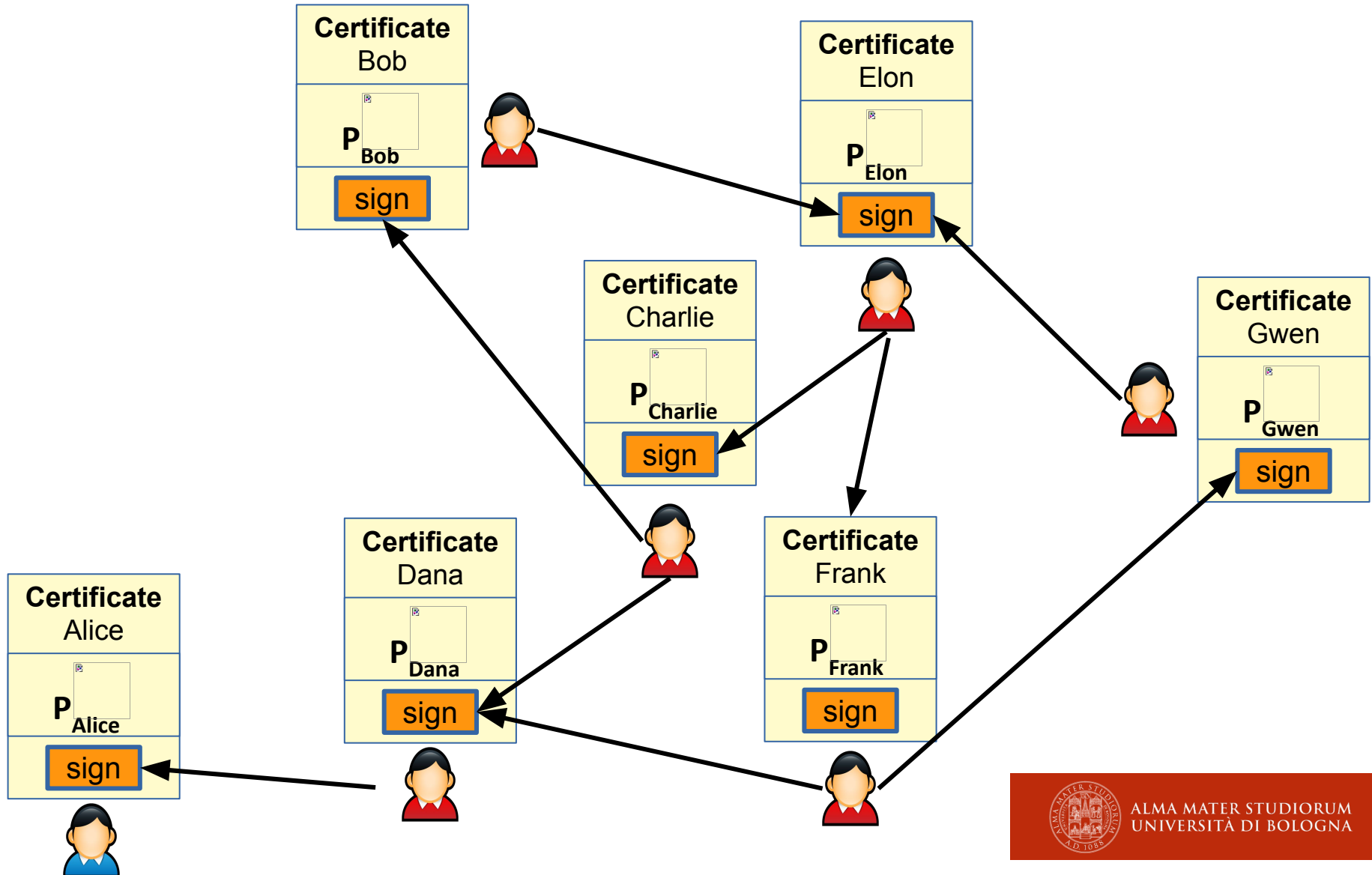
Certificato Digitale X.509



Infrastruttura a chiave pubblica X.509



Rete di fiducia (PGP)



Demo Firma Digitale

<https://www.phpdocx.com/demos/digital-signature-package>



electronic IDentification Authentication and Signature

- Il Regolamento UE n° 910/2014 - eIDAS
- Base normativa comune per **interazioni elettroniche sicure** fra cittadini, imprese e pubbliche amministrazioni
- **Interoperabilità a livello comunitario delle firme elettroniche e dei sistemi di validazione temporale**
"Una firma elettronica qualificata basata su un certificato qualificato rilasciato in uno Stato membro è riconosciuta quale firma elettronica qualificata in tutti gli altri Stati membri." (articolo 25, comma 3)



eIDAS riconosce 3 tipi di e-signature

1. Firme Elettroniche

Il regolamento eIDAS definisce un fondamento per tutte le firme elettroniche, affermando che **nessuna firma può essere negata legalmente soltanto per il fatto di essere in forma elettronica.**

Esempi

Firmare un'e-mail con il proprio nome o inserire un codice PIN



eIDAS riconosce 3 tipi di e-signature

2. Firme Elettroniche Avanzate (AdES)

Le firme AdES **devono corrispondere in modo univoco al firmatario e devono essere in grado di identificarlo.**

I firmatari generano la loro firma esclusivamente utilizzando dati posti sotto il loro controllo, mentre il documento finale deve essere a prova di manomissione.

Esempi

←Firme Digitali

XAdES, PAdES, CAdES, Associated Signature Container
Baseline Profile senza Certificato Qualificato, firma grafometrica, firma biometrica, ecc.



Firme Elettroniche Avanzate (AdES)



Document

SHA256

Ç!"\$£"(@0
(digest)

Sign

 S
Alice

K?çW"S0*
(signature)

Verify

 P
Alice

1. il documento finale deve essere a prova di manomissione
2. I firmatari generano la loro firma esclusivamente utilizzando dati posti sotto il loro controllo
3. devono corrispondere in modo univoco al firmatario e devono essere in grado di identificarlo



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

eIDAS AdES

- I formati che queste firme elettroniche avanzate devono possedere sono definiti nella **Decisione di esecuzione (UE) 2015/1506** (articolo 1):
 - “Gli Stati membri [...] riconoscono la firma elettronica avanzata **XML, CMS, PDF**”
- *“Le firme elettroniche avanzate di cui all'articolo 1 della decisione devono rispettare una delle seguenti specifiche tecniche ETSI”:*
 - **XAdES, CAdES, PAdES**



ETSI

- European Telecommunications Standards Institute
- Organizzazione non-profit responsabile della creazione e del mantenimento di questo insieme di **norme tecniche a sostegno del quadro giuridico eIDAS**.



XAdES: XML Advanced Electronic Signature

- Firme codificate in un formato testuale leggibile e conforme alle regole dell'XML (Extensible Markup Language).
- XAdES è leggibile **sia dall'uomo che dalla macchina**, il che lo rende adatto a una grande varietà di casi (immagini JPEG, file multimediali MP3, qualsiasi tipo di dati binari, documenti PDF, ecc.)
- XAdES consente **2 modalità** di firma:
 - **Detached**: produce un file XML **senza modificare il file iniziale**. I dati sono separati dalla firma, ma poi possono essere confezionati insieme.
 - **Encapsulated**: produce un file XML che include i dati. La firma poi impacchetta tutto insieme.
- **Vantaggio** -> facilita l'elaborazione automatica, supporta la firma multipla, due diversi firmatari possono firmare lo stesso documento o gruppi di documenti in parallelo o in sequenza.



CAdES: CMS Advanced Electronic Signature

- CMS -> Cryptographic Message Syntax, an IETF Standard per messaggi protetti da crittografia
- Le sue caratteristiche sono molto simili a quelle di XAdES, solo che CAdES **può essere applicato solo ai dati binari**.
- Inoltre, **manca** di alcuni concetti chiave di XAdES come la **la firma di più documenti**



PAdES: PDF Advanced Electronic Signature

- Questo formato è più limitato rispetto a XAdES -> solamente firma di file PDF
- Per impostazione predefinita, **la firma elettronica è sempre incorporata nel documento PDF firmato**, che è leggibile solo dall'uomo.
- Non è quindi adatto nel caso in cui i dati debbano essere letti anche da un computer.
- PAdES non supporta la firma parallela e **richiede un software PDF per firmare e verificare** la firma elettronica es. -> Adobe Reader.



eIDAS riconosce 3 tipi di e-signature

3. Firme Elettroniche Qualificate (QES)

QES è una forma più rigorosa di AdES. Ha lo stesso valore legale delle firme tradizionali.

Richiede ai firmatari di:

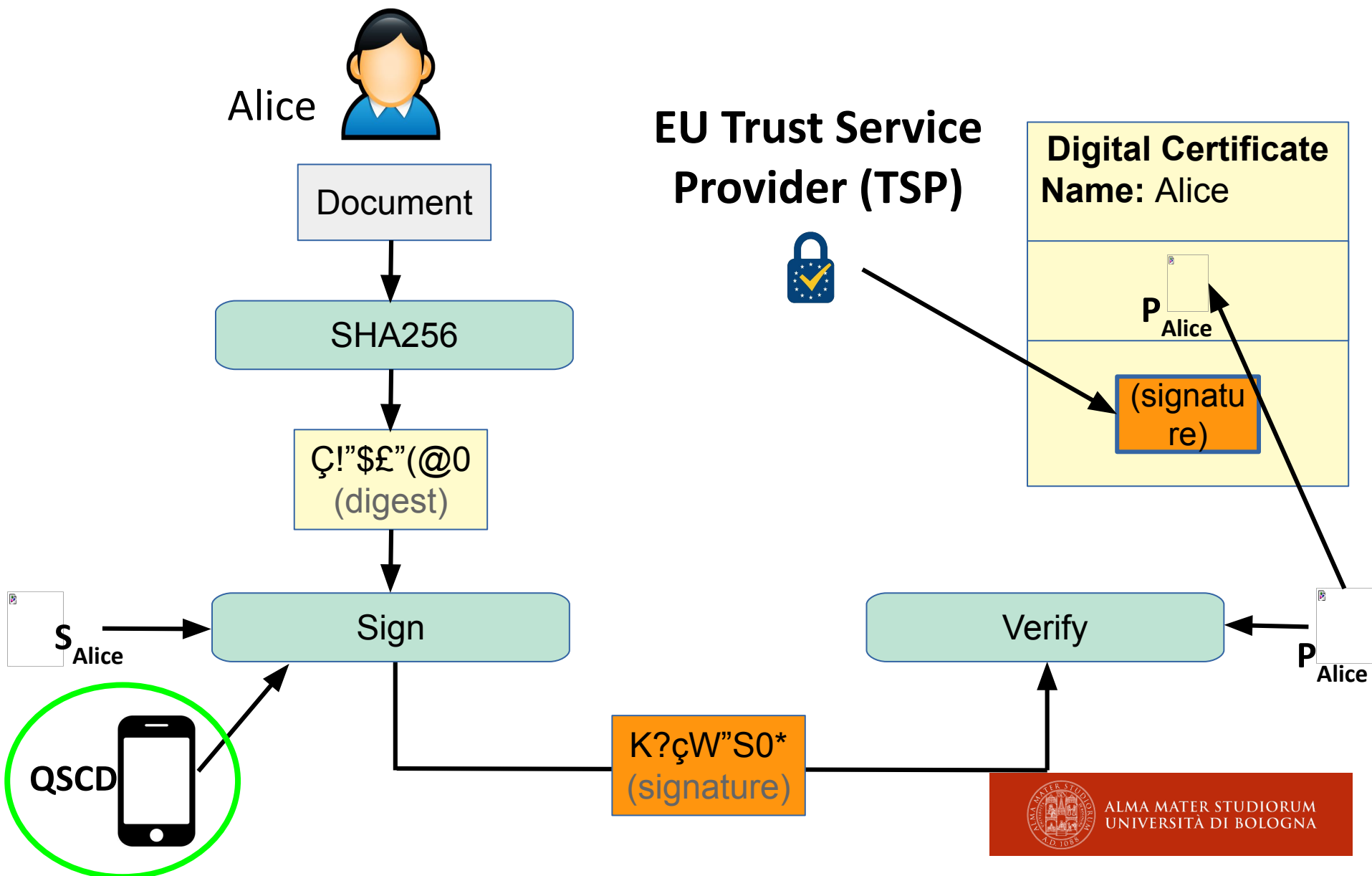
- a. utilizzare un **ID digitale basato su un Certificato Digitale**, rilasciato da un **EU Trust Service Provider (TSP)** qualificato
- b. utilizzare un **dispositivo per la creazione di una firma qualificata (QSCD)**.

Esempi

XAdES, PAdES, CAdES con Certificato Qualificato e dispositivo dicuro: smart card, USB token, o smartphone con una password one-time



Firme Elettroniche Qualificate (QES)

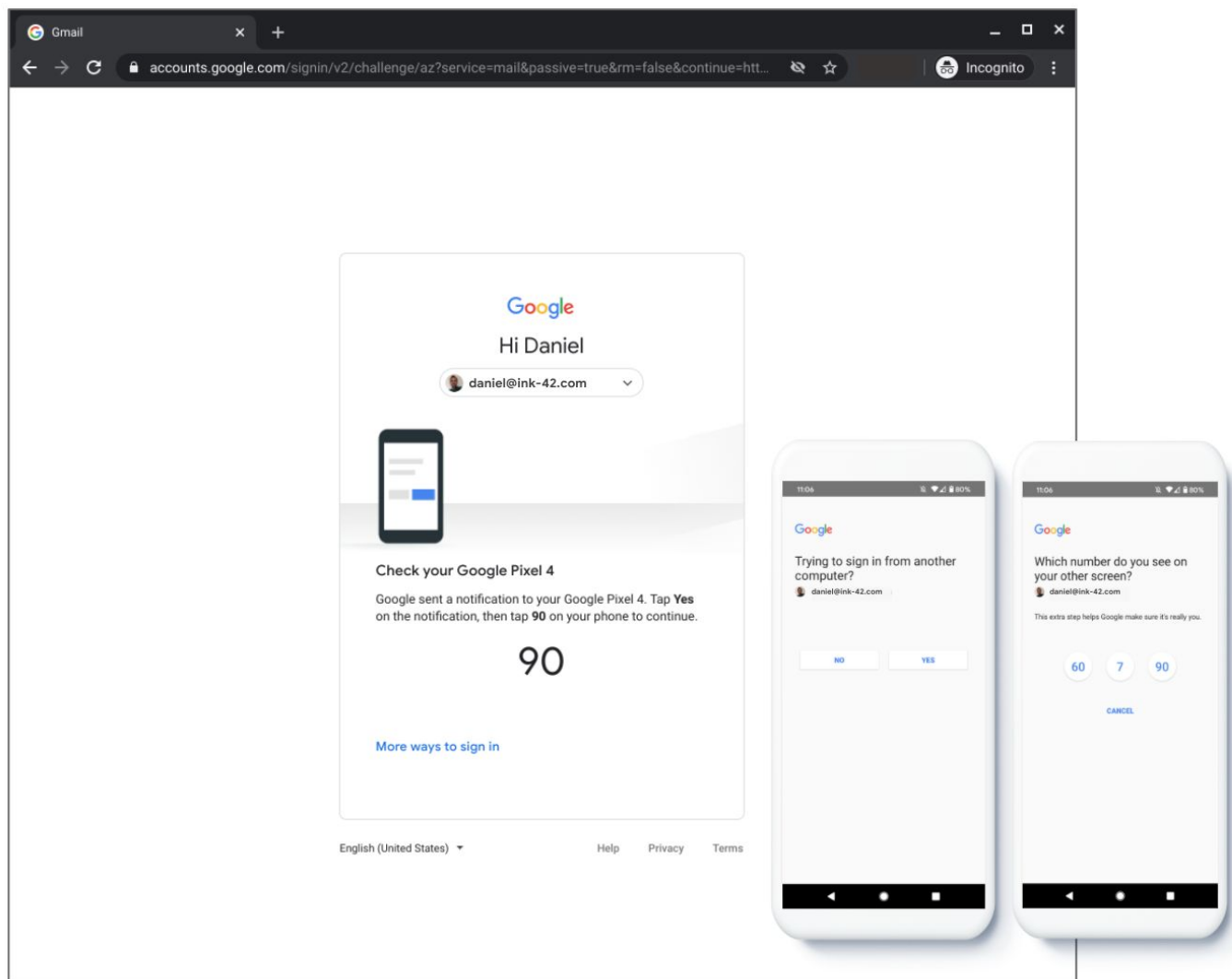


Password One-time e verifica in due passaggi

- Un'autenticazione forte combina due o più di:
 - Qualcosa che **conosci**
 - Qualcosa che **hai**
 - Qualcosa che **sei** (impronta digitale)
- Una combinazione delle prime due è la più comune ed è conosciuta come **verifica in due passaggi**:
 - Conosci: Una **password personale**
 - Hai: un oggetto fisico come un "security token" della banca o **uno smartphone con una Password One-Time**



Verifica in due passaggi



Password One-time: Challenge-Response

1. Alice dichiara la sua intenzione di accedere al Servizio
2. Il Servizio seleziona una "**sfida**" e la invia ad Alice
3. Alice calcola una "**risposta**" alla sfida e la rimanda indietro
4. Il Servizio confronta la risposta ricevuta da Alice con la **risposta "attesa"** per la sfida che ha inviato
 - Se corrispondono, accesso consentito, altrimenti no

One-Time -> la "**risposta**" è **unica** per la sfida e può essere utilizzata una sola volta (perché la "**sfida**" **cambia** ogni volta)



Password One-time: Challenge-Response

Crittografia Simmetrica

1. Alice dichiara la sua intenzione di accedere al Servizio con il quale condivide una chiave segreta **K**
2. La sfida del Servizio è: una **stringa random** “ciaosfida” inviata ad Alice
3. Alice calcola la risposta alla sfida cifrandola con la chiave **K**
risposta = C(“ciaosfida”, K)
4. Il Servizio decifra la risposta, **risultato = D(risposta, K)** e confronta il risultato con “ciaosfida”
 - Se **risultato == “ciaosfida”**, accesso consentito, altrimenti no



Password One-time: Challenge-Response

Crittografia Asimmetrica

1. Alice dichiara la sua intenzione di accedere al Servizio con il quale condivide una chiave pubblica
2. La sfida del Servizio è: una **stringa random** “ciaosfida” inviata ad Alice
3. Alice calcola la risposta alla sfida firmandola digitalmente
risposta = sign(“ciaosfida”)
4. Il Servizio verifica la risposta, **risultato = verify(risposta)**
 - Se la firma è valida, accesso consentito, altrimenti no



Demo Firma Digitale eIDAS

<https://ec.europa.eu/digital-building-blocks/DSS/webapp-demo/sign-a-document>





ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Marcatatura Temporale Fidata

Marca Temporale (timestamp)

- Sequenza di caratteri rappresentante **una data e/o un'ora** per accertare l'effettivo verificarsi di un determinato evento

2020-10-07T15:54:19+00:00

- Standard **ISO 8601** per la rappresentazione -> usato nei protocolli di rete per limitare la possibilità di errore
- Nella maggior parte dei calcolatori viene derivato tramite lo **Unix time** ->

il numero di secondi passati dal 1° Gennaio 1970

1602086059

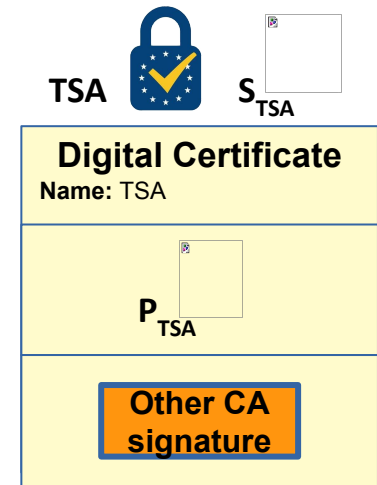
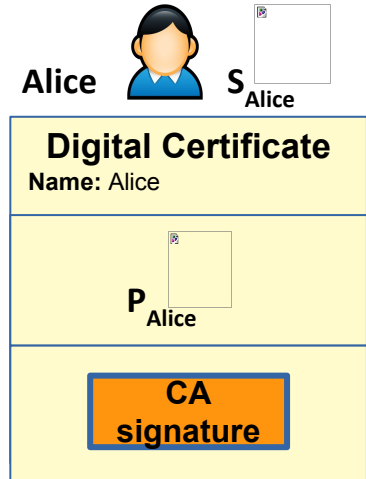


Marcatura Temporale basata su Infrastruttura a chiave pubblica

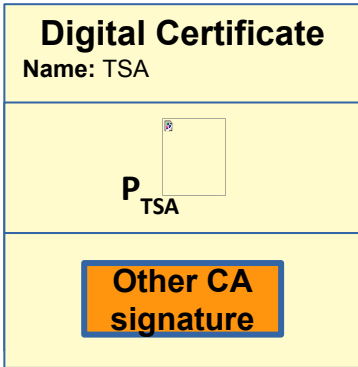
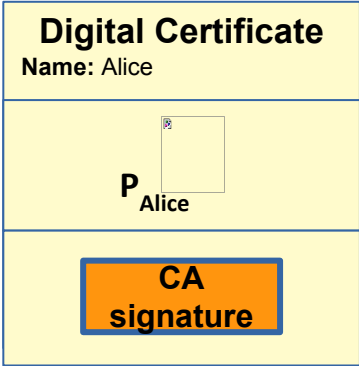
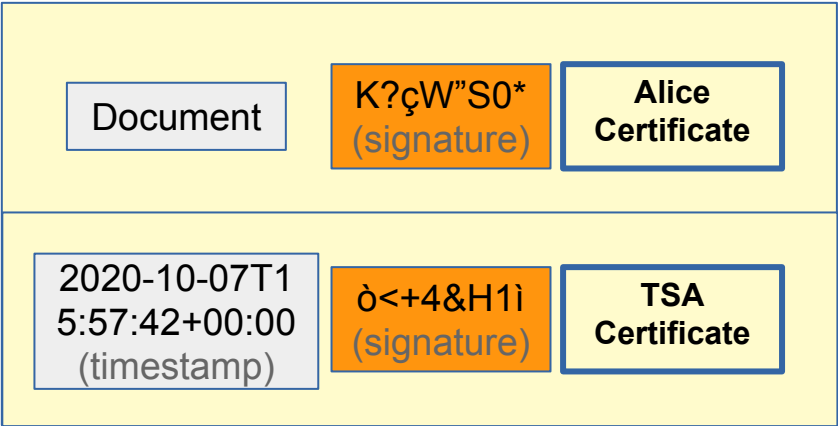
- L'apposizione della marca temporale permette di **stabilire l'esistenza ed il contenuto del documento a partire da un determinato momento.**
- Lo standard **RFC 3161** definisce il processo di marcatura temporale fidata basato su una **PKI X.509**
- Il processo di marcatura temporale consiste nell'apposizione di un timestamp su un documento digitale, da parte di un **Certificatore Accreditato** (Time Stamping Authority **TSA**), mediante firma digitale sul documento.



PKI



Marcatura Temporale



ANSI ASC X9.95 Standard

- **L'ANSI X9.95** è un'estensione del RFC 3161 per garantire una maggiore sicurezza sull'integrità dei dati
 - **Schemi basati sul collegamento** -> il timestamp viene generato in modo tale da essere collegato ad altri timestamp (merkle tree).
 - **Schema a chiave transitoria** -> variante PKI con chiavi di firma che hanno una "breve durata".
 - **MAC** -> schema semplice basato su una chiave segreta condivisa
 - **Database** -> gli hash dei documenti sono archiviati in un archivio fidato.
 - **Schemi ibridi**



Marcatatura temporale distribuita

- Invece di un unico TSA ci si può affidare ad un **algoritmo distribuito** che guida **diverse parti** che dialogano tra di loro a raggiungere un **consenso** ->
- Con l'avvento della **blockchain** e delle tecnologie relative (**Distributed Ledger Technologies**), l'hash dei documenti digitali può essere incorporato in una transazione che viene memorizzata nella blockchain.
- In questo caso l'immutabilità della DLT prova l'ora in cui quei dati esistevano.



Marcatatura temporale distribuita: Problemi

- La sicurezza di questo approccio deriva dal **meccanismo di consenso**. Es. in Bitcoin questo è il **Proof of Work** (PoW), un enorme lavoro di calcolo effettuato ogni volta che viene aggiunto un nuovo blocco alla blockchain.
- La manomissione del timestamp richiederebbe più risorse di calcolo rispetto al resto della rete combinata.
- Tuttavia, il protocollo di molte DLTs rende i suoi **timestamp vulnerabili ad un certo grado di manipolazione** -> un timestamp può essere spostato fino a due ore nel futuro e possono essere accettati prima dei dati con timestamps antecedenti.





ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Identità decentralizzate

Chiavi come Identità

Mirko Zichichi



associato a

X1457cb4jiuKlo98hgf

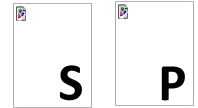
Indirizzo Pubblico

(alphanumeric)



Come creare una nuova Identità

- Creare una nuova coppia di chiavi asimmetriche (sk , pk)
 - La chiave pubblica pk è il “nome” dell’identità
 - Pseudonimo
 - pk spesso chiamata **indirizzo** (address)
 - Meglio usare **Hash**(pk) come indirizzo
 - La chiave privata sk permette di “**parlare a nome**” dell’identità
- Ognuno può controllare la propria identità perché solo lui conosce sk
- Se pk “**sembra random**”, nessuno sa associarla ad un individuo

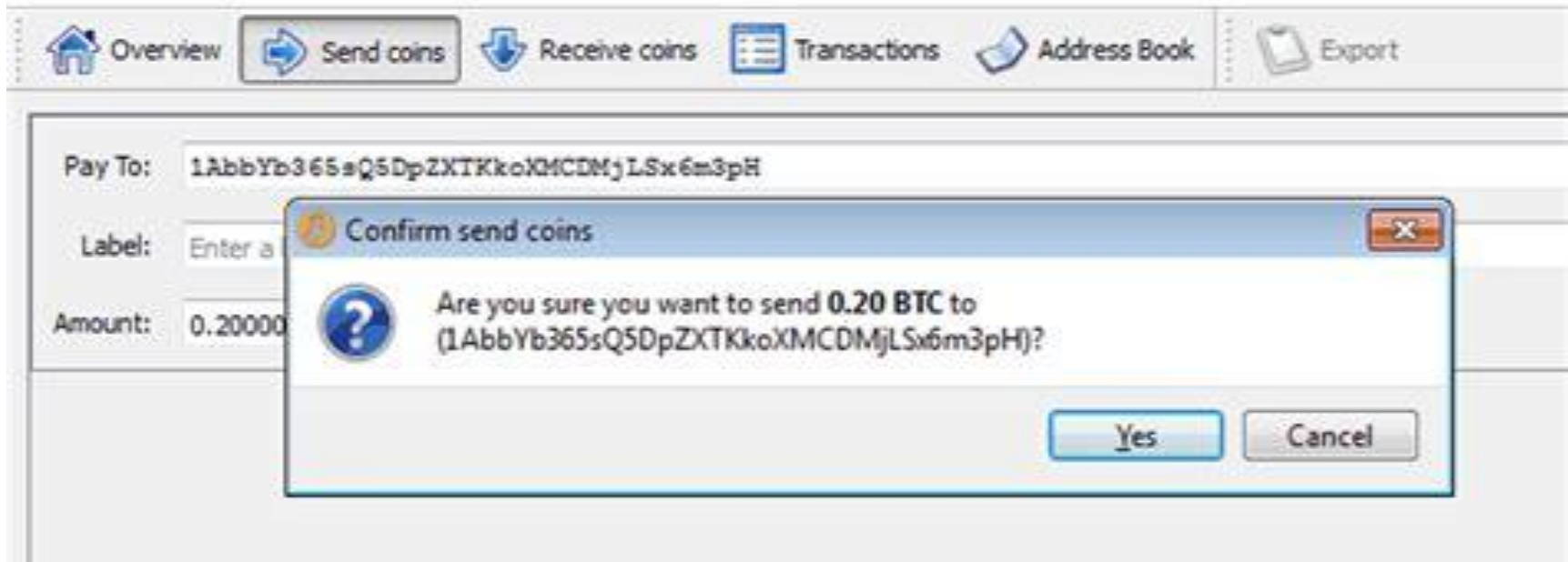


Gestione decentralizzata delle identità

- Chiunque può **creare una nuova identità in qualsiasi momento** e farne quante ne vuole!
- La probabilità di generare la stessa chiave di un altro utente è trascurabile
- **Nessun punto centrale di coordinamento**
- Nessuna autorità centrale che registri le identità nel sistema
- Queste "identità decentralizzate" sono chiamate **"indirizzi" in Bitcoin**



Gestione decentralizzata delle identità



Identificatori Decentralizzati (DID)

- Un tipo di identificativi che consentono un'**identità digitale verificabile e decentralizzata**.
- Essi si basano sul paradigma dell'**identità auto-sovrana** (self-sovereign identity).
- Un DID identifica **qualsiasi soggetto** (ad es. una persona, un'organizzazione, una cosa, un modello di dati, ecc.)
- Questi identificatori sono progettati per consentire al controllore di un DID di dimostrare il controllo su di esso e per essere implementati indipendentemente da qualsiasi registro centralizzato, fornitore di identità o autorità di certificazione.



Identificatori Decentralizzati (DID)

The standard elements of a DID doc

1. **DID** (for self-description)
2. **Set of public keys** (for verification)
3. **Set of auth methods** (for authentication)
4. **Set of service endpoints** (for interaction)
5. **Timestamp** (for audit history)
6. **Signature** (for integrity)



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Mirko Zichichi

mirko.zichichi2@unibo.it

<https://mirkozichichi.me>