Crittografia

Crittografia simmetrica e asimmetrica, funzioni di hashing e firma digitale

Crittografia Simmetrica

Crittografia Simmetrica

Panoramica 1/5

La **crittografia simmetrica** è il tipo di cifratura più largamente utilizzata, anche da personale non addetto ai lavori come diplomatici, militari e aziende.

Questo tipo di crittografia richiede 5 elementi: **plaintext**, **encrypt algorithm**, **secret key**, **ciphertext** e **decrypt algorithm**.

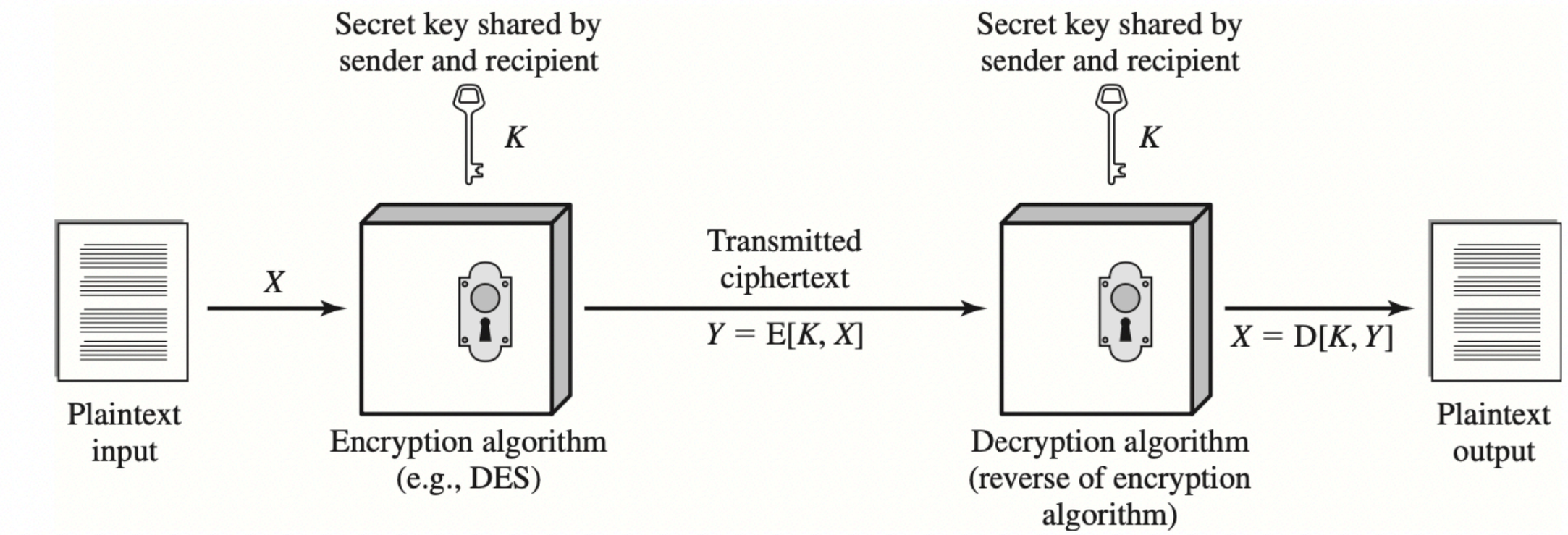
Crittografia Simmetrica

Panoramica 2/5

- **Plaintext** - Testo originale, in chiaro
- **Encrypt algorithm** - Algoritmo di cifratura (es. DSA, AES)
- **Secret key** - Chiave per la cifratura (password o passphrase)
- **Ciphertext** - Testo in output dopo aver applicato l’algoritmo di cifratura. È cifrato
- **Decrypt algorithm** - Algoritmo di decifratura

Crittografia Simmetrica

Panoramica 3/5



Crittografia Simmetrica

Panoramica 4/5

I requisiti per l'utilizzo sicuro della crittografia simmetrica sono due:

- Utilizzare un algoritmo forte: un attaccante, pur avendo più **ciphertext** e conoscendo l'algoritmo, non deve essere in grado di risalire al **plaintext**.

- Secret key condivisa: chi invia il messaggio e chi lo riceve devono essere in possesso della **secret key** per poter decifrare il messaggio.

Crittografia Simmetrica

Panoramica 5/5

Esistono due tipologie di attacco per la crittografia simmetrica: il primo è la **crittoanalisi**, questo studia coppie *plaintext-ciphertext* per capire il funzionamento dell'algoritmo; il secondo è l'attacco **brute-froce**, esso consiste nel testare tutte le combinazioni di chiavi possibili fino a trovare quella esatta.

Crittografia Simmetrica

Algoritmi di cifratura a blocchi 1/3

Nella cifratura simmetrica troviamo due categorie di algoritmi: **a blocchi** o **a flusso**.

Gli algoritmi a blocchi sono i più diffusi, essi dividono il plaintext in blocchi di lunghezza fissa (es. 64 bit) e cifrano l'intero blocco.

Tra gli algoritmi più conosciuti troviamo il **DES** (Data Encryption Standard) e l'**AES** (Advanced Encryption Standard).

Crittografia Simmetrica

Algoritmi di cifratura a blocchi 2/3

Nell'immagine sotto si mettono a confronto DES, 3DES e AES.

Table 2.1 Comparison of Three Popular Symmetric Encryption Algorithms

	DES	Triple DES	AES
Plaintext block size (bits)	64	64	128
Ciphertext block size (bits)	64	64	128
Key size (bits)	56	112 or 168	128, 192, or 256

Crittografia Simmetrica

Algoritmi di cifratura a blocchi 3/3

Recenti studi hanno dimostrato come l'algoritmo DES (con chiave a 56 bit) non sia più sicuro per via dell'aumento della potenza computazionale dei computer.

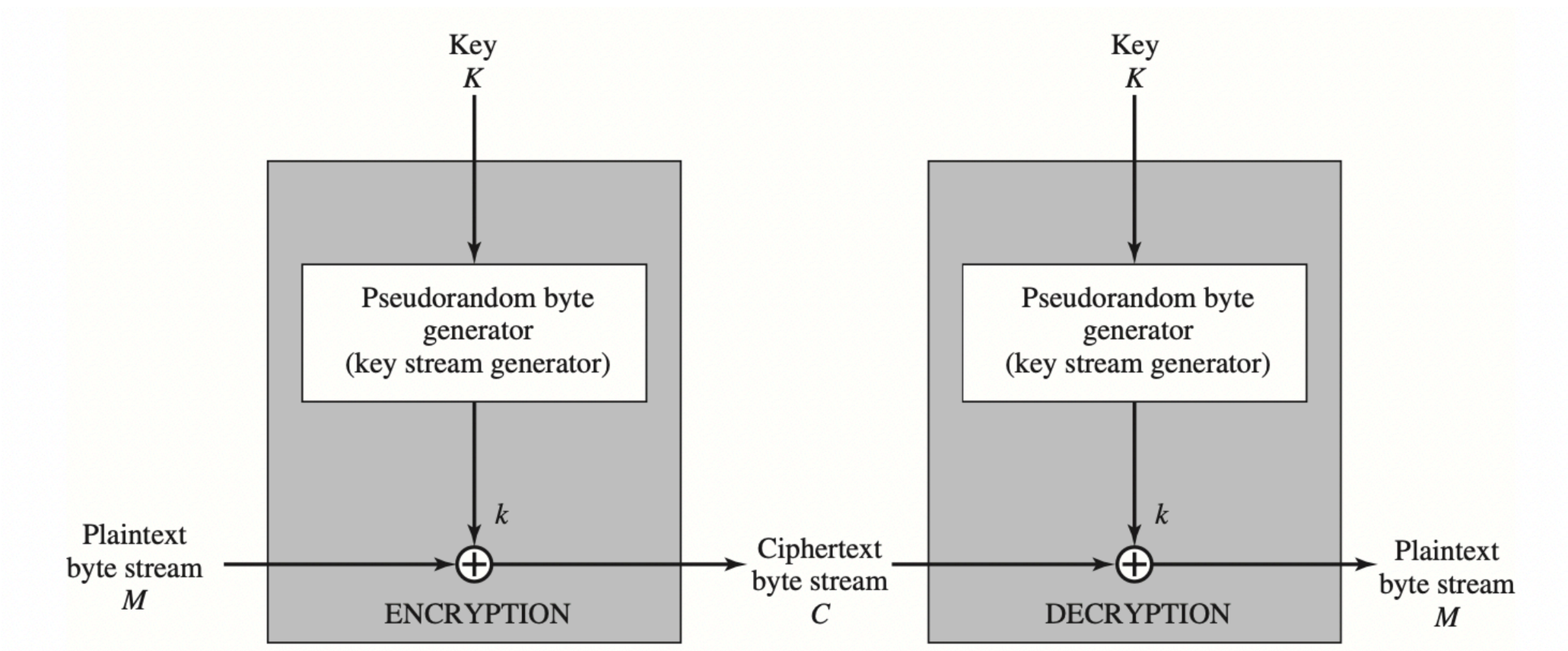
L'immagina mostra il tempo necessario per trovare una chiave:

Key Size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 decryptions/μs	Time Required at 10^{13} decryptions/μs
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	$2^{55}\ \mu s = 1.125$ years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	$2^{127}\ \mu s = 5.3 \times 10^{21}$ years	5.3×10^{17} years
168	Triple DES	$2^{168} \approx 3.7 \times 10^{50}$	$2^{167}\ \mu s = 5.8 \times 10^{33}$ years	5.8×10^{29} years
192	AES	$2^{192} \approx 6.3 \times 10^{57}$	$2^{191}\ \mu s = 9.8 \times 10^{40}$ years	9.8×10^{36} years
256	AES	$2^{256} \approx 1.2 \times 10^{77}$	$2^{255}\ \mu s = 1.8 \times 10^{60}$ years	1.8×10^{56} years

Crittografia Simmetrica

Algoritmi di cifratura a flusso 1/2

Nella cifratura simmetrica a flusso l'algoritmo cifra gli elementi in input continuamente.



(b) Stream encryption

Crittografia Simmetrica

Algoritmi di cifratura a flusso 1/2

Tipicamente viene cifrato un byte (8 bit) alla volta, nonostante esistano implementazioni di cifratura bit per bit o a più di 1 byte.

Nella figura della slide precedente la chiave è in input ad un generatore di byte pseudocasuali, il prodotto, chiamato **keystream**, viene combinato in XOR con il **plaintext** per produrre il **ciphertext**.

Crittografia Asimmetrica

Crittografia Asimmetrica

Panoramica e struttura 1/2

La **crittografia asimmetrica**, anche conosciuta come crittografia a chiave pubblica, è stata introdotta da Diffie e Hellman nel 1976 e ha rivoluzionato il mondo della crittografia.

Essa consiste nell'utilizzo di **due chiavi**, una chiave **pubblica** e una **privata**.

Rispetto alla crittografia simmetrica abbiamo un elemento in più in quanto la secret-key non esiste, ma vengono introdotte **public-key** e **private-key**.

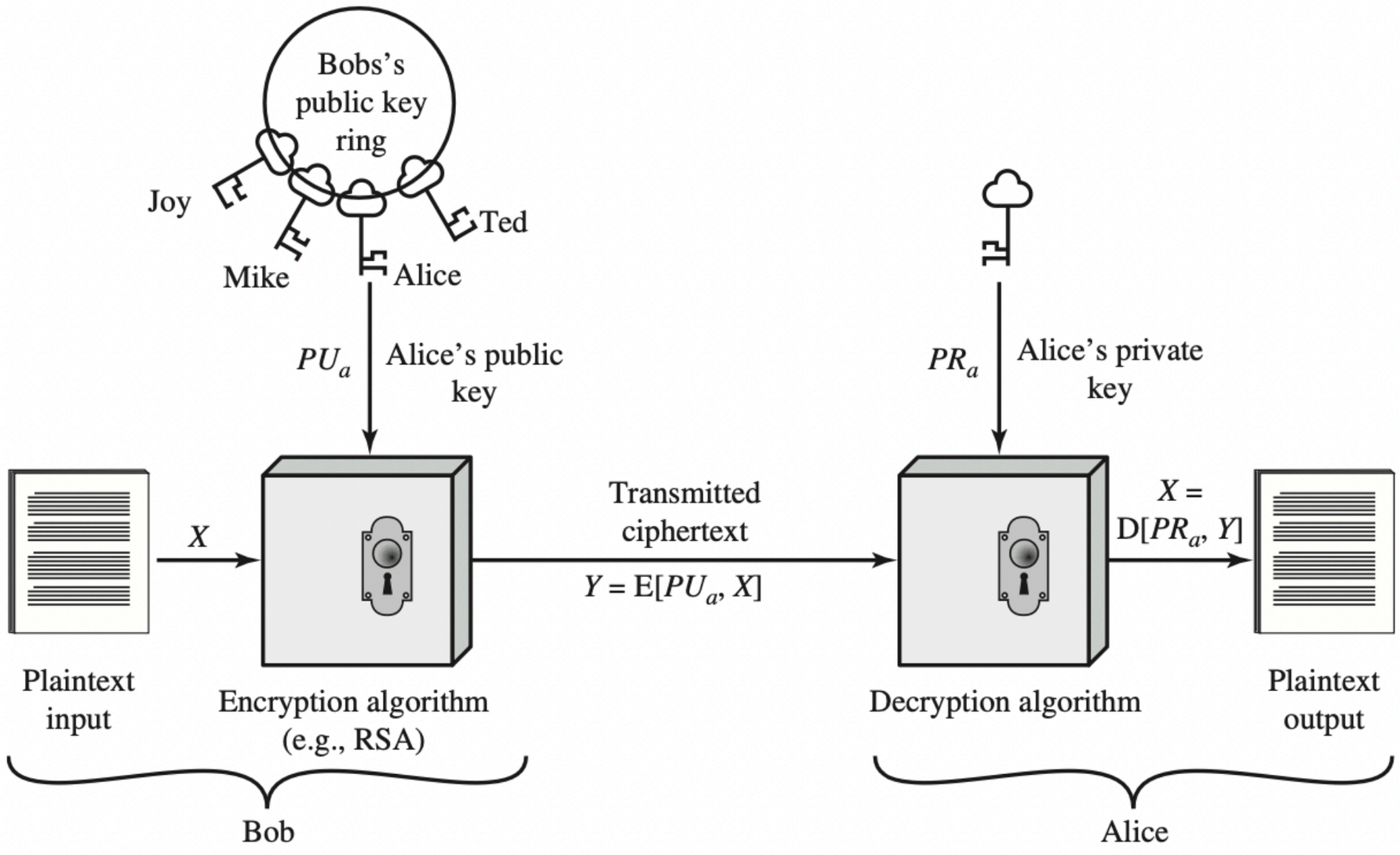
Crittografia Asimmetrica

Panoramica e struttura 2/2

1. Ogni utente genera una coppia di chiavi
2. Una delle chiavi viene caricata in un repository pubblico, questa è la **chiave pubblica**. La chiave complementare viene tenuta dal mittente, questa sarà la **chiave privata**.
3. Se Bob vuole mandare un messaggio ad Alice, Bob cifra il messaggio con la chiave pubblica di Alice.
4. Quando Alice riceve il messaggio lo decripta con la sua chiave privata. Nessun altro può leggere il messaggio.

Crittografia Asimmetrica

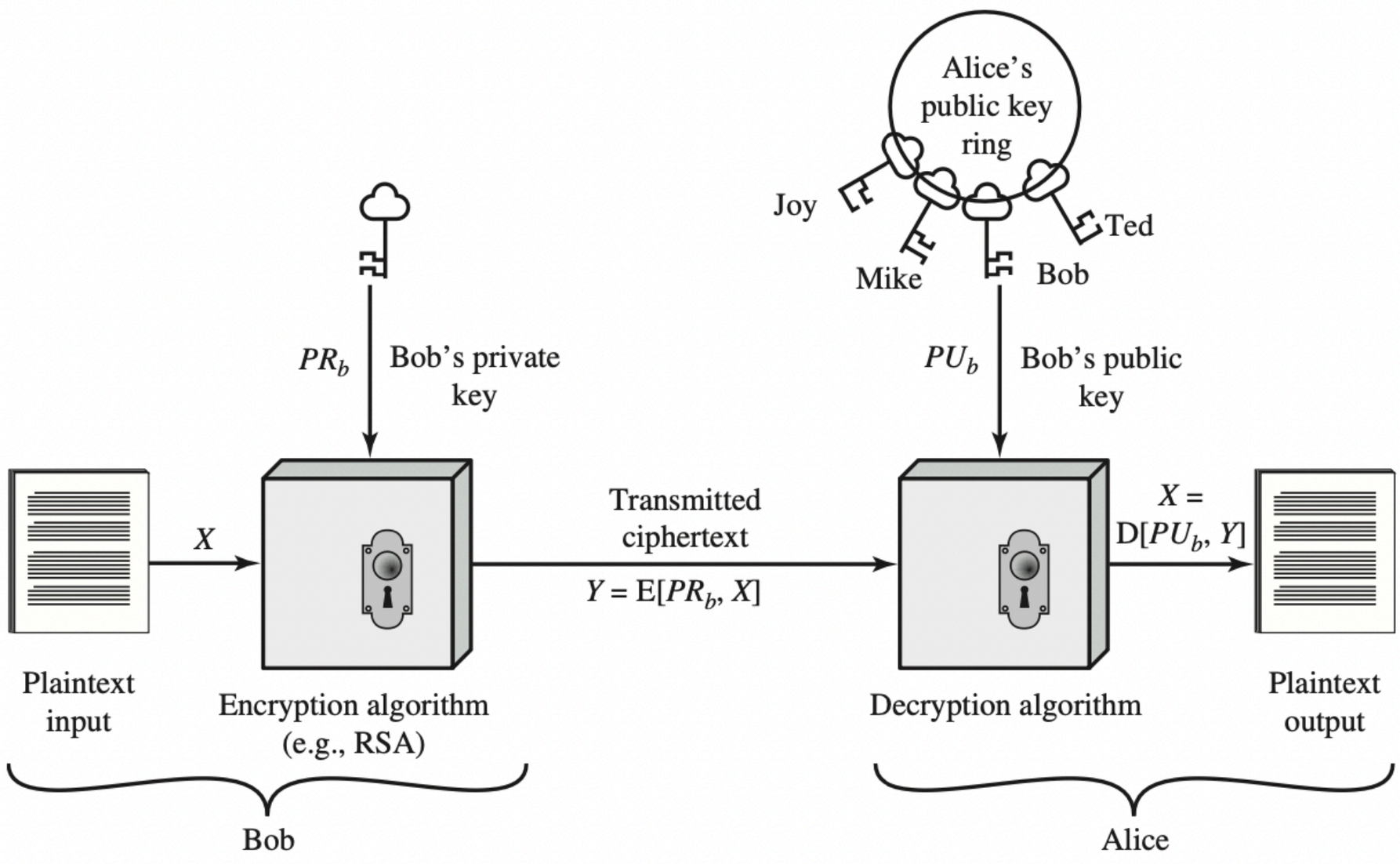
Cifratura con Chiave Pubblica



(a) Encryption with public key

Crittografia Asimmetrica

Cifratura con Chiave Privata



(b) Encryption with private key

Message Authentication e funzioni di hashing

Message Authentication

Autenticazione tramite crittografia simmetrica

L’autenticazione dei messaggi (o dati, nota come Message Authentication) ci aiuta contro la falsificazione dei dati, garantisce quindi l’autenticità di un messaggio e che la sorgente sia autentica.

La **crittografia simmetrica**, considerata singolarmente, non può garantire l’autenticità dei dati.

Message Authentication

Autenticazione senza cifratura dei messaggi

L’autenticazione senza la cifratura, quindi non garantendo la confidenzialità, è conveniente per i sistemi in quanto le funzioni per criptare e decriptare i messaggi sono CPU intensive.

Message Authentication

Message Authentication Code (MAC) 1/3

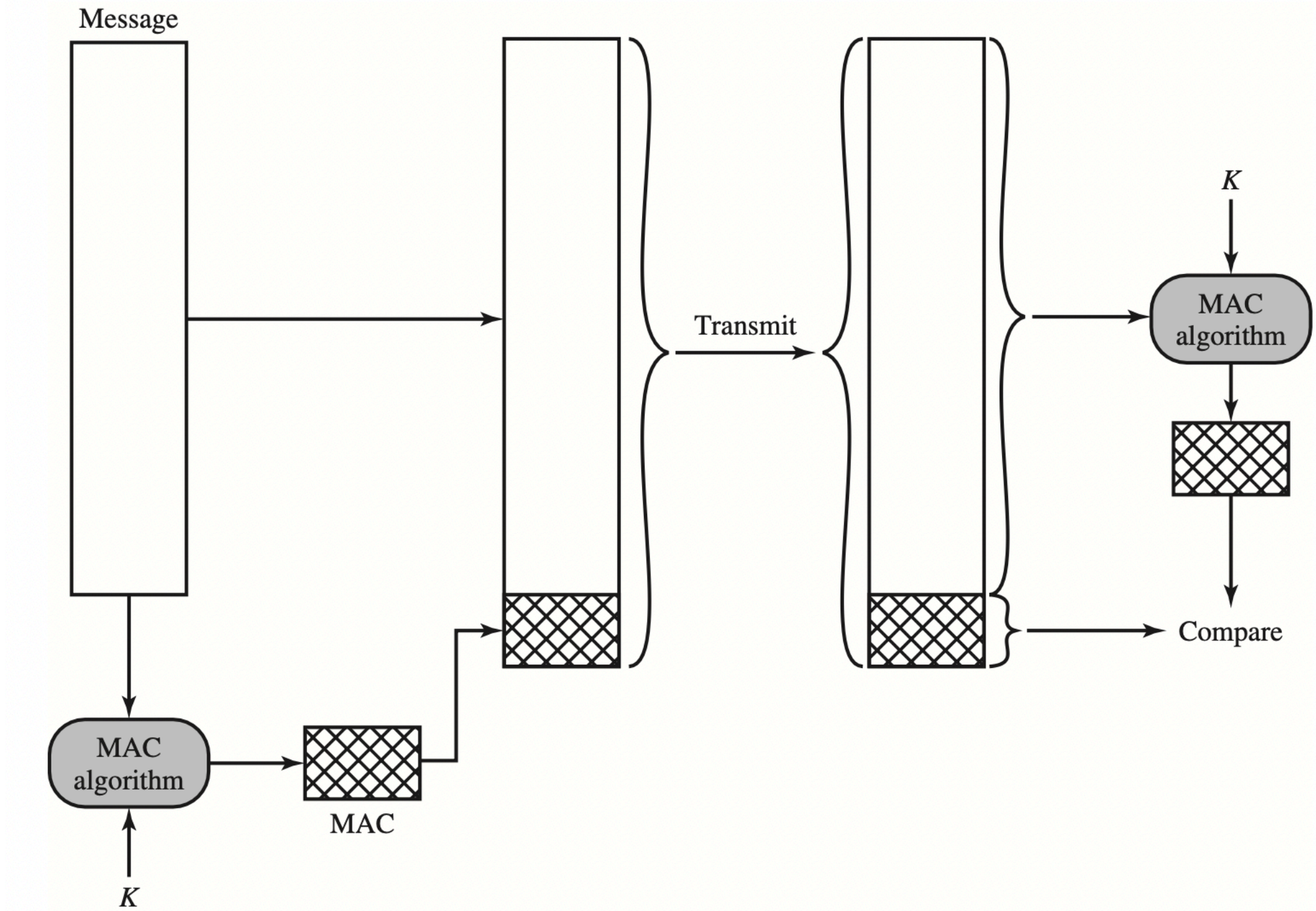
Una tecnica per la Message Authentication è il MAC, consiste nell'utilizzare una secret key per generare un blocco dati.

Fornisce **autenticazione** e **integrità**.

A deve inviare un messaggio a B, calcola il MAC come funzione le cui variabili sono il messaggio e la secret key: *MAC = F(K + M)*, il MAC viene accodato al messaggio e trasmesso. B ricalcola il MAC con la stessa secret key e lo compara con quello ricevuto.

Message Authentication

Message Authentication Code (MAC) 2/3



Message Authentication

Message Authentication Code (MAC) 3/3

Il processo è simile a quello visto per la cifratura, la differenza è nell'algoritmo di autenticazione che non deve essere reversibile, requisito fondamentale per le funzioni di decrittazione.

HMAC sono implementazioni del MAC con funzioni di hashing (SHA, MD5, ecc..)

Message Authentication

Funzioni di hashing one-way 1/6

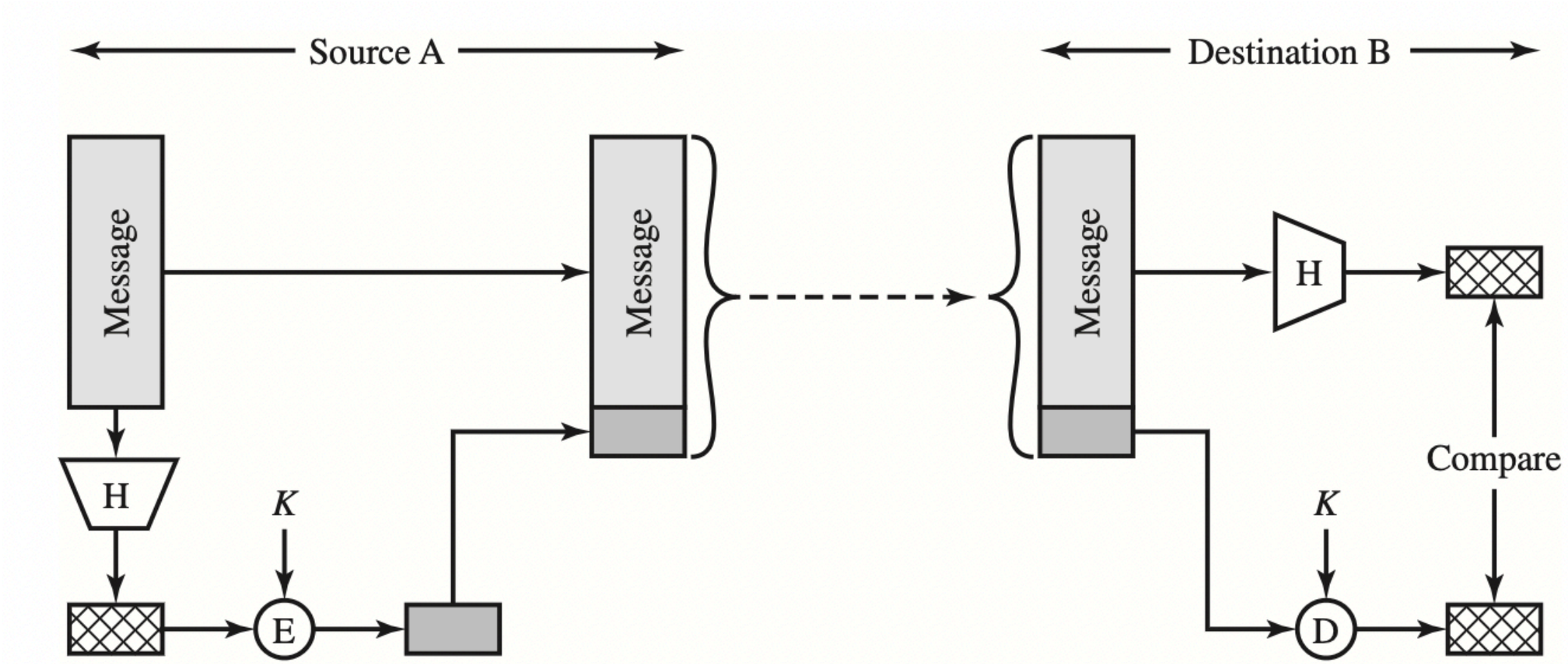
Le funzioni di hashing hanno un funzionamento simile: ricevono in input messaggi di lunghezza variabile e restituiscono una stringa, chiamata **digest**, di lunghezza fissa (128, 192, 256bit).

A differenza del MAC, con le funzioni di hashing non abbiamo bisogno di una secret key.

Message Authentication

Funzioni di hashing one-way 2/6

Autenticazione di messaggi tramite digest cifrati con **cifratura simmetrica**.

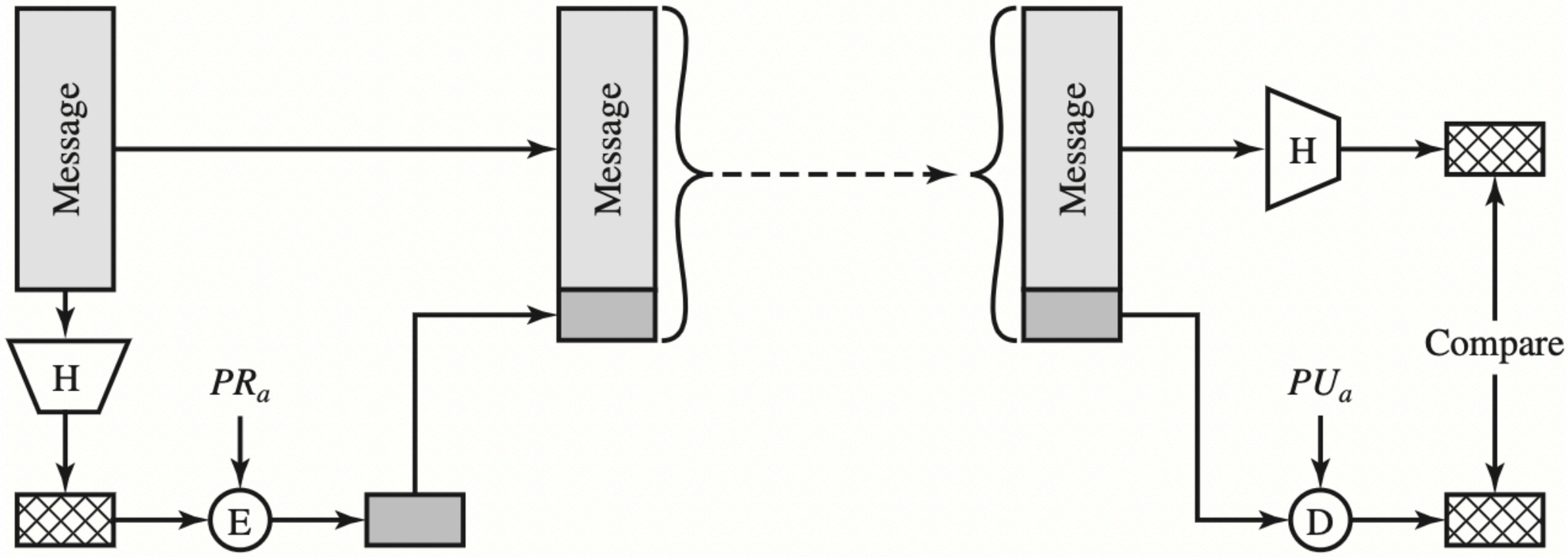


(a) Using symmetric encryption

Message Authentication

Funzioni di hashing one-way 3/6

Autenticazione di messaggi tramite digest cifrati con **cifratura asimmetrica**.

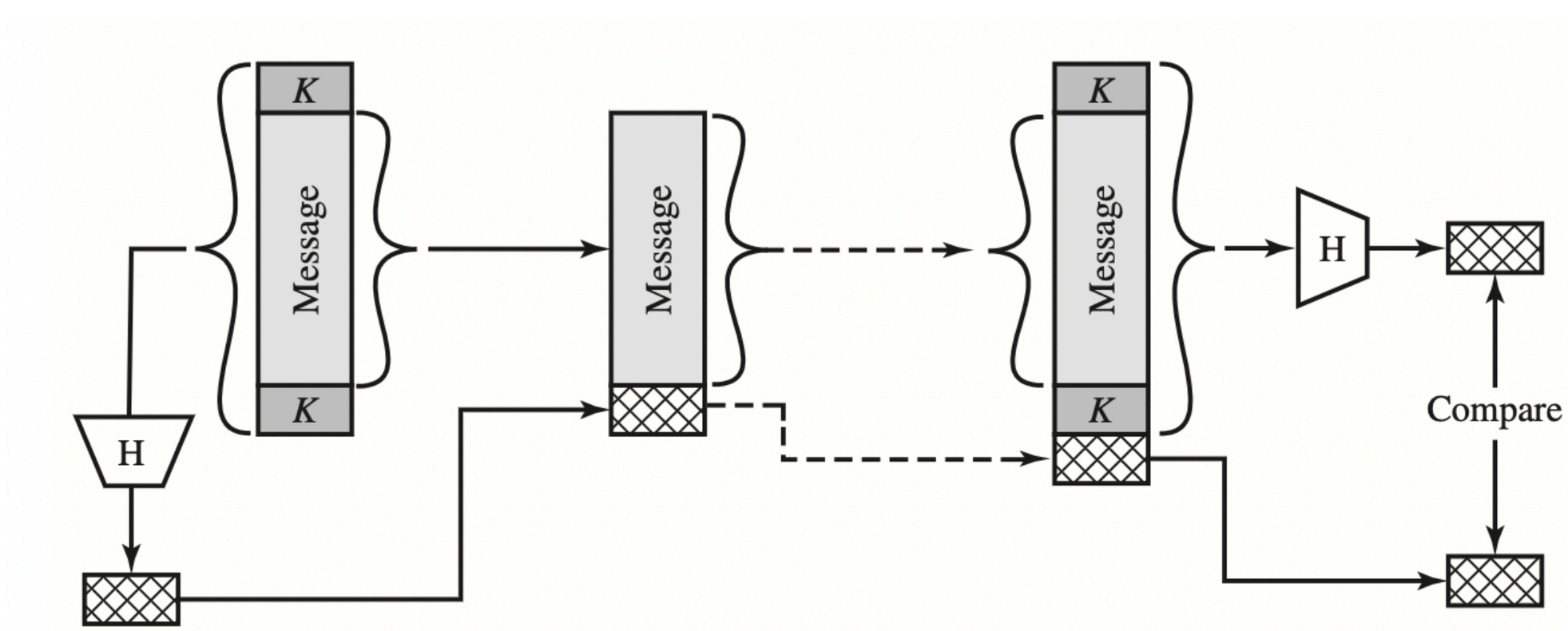


(b) Using public-key encryption

Message Authentication

Funzioni di hashing one-way 4/6

Autenticazione di messaggi tramite digest del messaggio concatenato ad una secret key.



(c) Using secret value

Message Authentication

Funzioni di hashing one-way 5/6

- Le funzioni di cifratura sono lente, anche se i dati da cifrare sono pochi.
- L'hardware per la cifratura è ottimizzato per grandi quantità di dati. Per piccoli blocchi si spende più tempo per l'inizializzazione e l'invocazione.
- Gli algoritmi di cifratura potrebbero essere coperti da un brevetto.

Message Authentication

Funzioni di hashing one-way 6/6

Nell’implementazione con concatenazione è impossibile per un attaccante risalire alla secret key, pertanto non si può alterare il messaggio se non rovinandolo.

Da notare che la secret key è concatenata sia prima del messaggio che dopo, questo aumenta la sicurezza.

Funzioni di hashing

Funzioni di hashing one-way 1/

Sono funzioni matematiche che non permettono di tornare al messaggio originale una volta calcolato il risultato (digest).

Una funzione di hash produce una “fingerprint” di un file, messaggio o qualsiasi altro blocco di dati.

Funzioni di hashing

Funzioni di hashing one-way 2/

Una funzione di hash (H) per garantire autenticazione del messaggio deve avere le seguenti proprietà:

1. H può essere applicata a blocchi di dati di qualsiasi dimensione
2. H produce una stringa di dimensioni fisse
3. H(x) deve essere di semplice computazione per qualsiasi x
4. Deve essere infattibile trovare x dato h, avendo H(x) = h
5. Per qualsiasi blocco x, è infattibile trovare y ≠ x con H(y) = H(x). Conosciuta come **resistenza alle collisioni debole**.
6. È infattibile trovare due messaggi x e y, tali per cui H(x) = H(y). Conosciuta come **resistenza alle collisioni forte**.

Funzioni di hashing

Funzioni di hashing one-way 3/

Le **prime 3 proprietà** sono requisiti per le applicazioni pratiche delle funzioni di cash per garantire la Message Authentication.

La **quarta proprietà** è fondamentale se la tecnica di autenticazione richiede l'utilizzo di una secret key, in modo da evitare che la chiave venga scoperta.

La **quinta proprietà** garantisce che sia impossibile trovare un altro messaggio con lo stesso **digest** dell'originale.

Funzioni di hashing

Altre applicazioni degli hash

- **Password** - Non permettere che le password siano decifrabili una volta salvate, così si permette l'accesso alle sole persone autorizzate (anche gli amministratori di sistema non possono recuperare la password originale)

- **Intrusion Detection** - Usare i digest dei file per verificare eventuali modifiche, confrontarli con i digest dei malware, riconoscere **signature** anomale nel traffico di rete

Algoritmi di Cifratura Asimmetrica

Algoritmi di Cifratura Asimmetrica

RSA

È l’algoritmo più conosciuto. Sviluppato nel 1977 da Rivest, Shamir e Adleman (da cui R.S.A.). Cifratura a blocchi.

Nel 1994 è stato ricavato il plaintext partendo da un ciphertext, la chiave era di 129 cifre, 428 bit.

Oggigiorno vengono usate chiavi di almeno 1024 bit (circa 300 cifre) che assicurano un buon livello di sicurezza.

Usare chiavi più lunghe 2048, 4096 aumenta la sicurezza perché aumenta il tempo di “cracking”.

Algoritmi di Cifratura Asimmetrica

Diffie-Hellman Key Agreement

Noto anche come Diffie-Hellman Key Exchange.

Lo scopo dell'algoritmo è permettere lo scambio di una secret key tra due parti in maniera sicura.

L'algoritmo è limitato al solo scambio di chiavi.

Firma Digitale

Firma Digitale

La crittografia asimmetrica può essere utilizzata per fornire autenticazione con una tecnica chiamata **Digital Signature** (Firma Digitale), la cui definizione è:

Il risultato di una trasformazione crittografica di dati che, se correttamente implementata, fornisce un meccanismo per verificare **autenticazione**, **integrità** e **non-repudiation**.

Questo metodo garantisce che il blocco dati è stato firmato dal firmatario indicato e che non è stato alterato dal momento della firma.

Firma Digitale

Algoritmi di Digital Signature

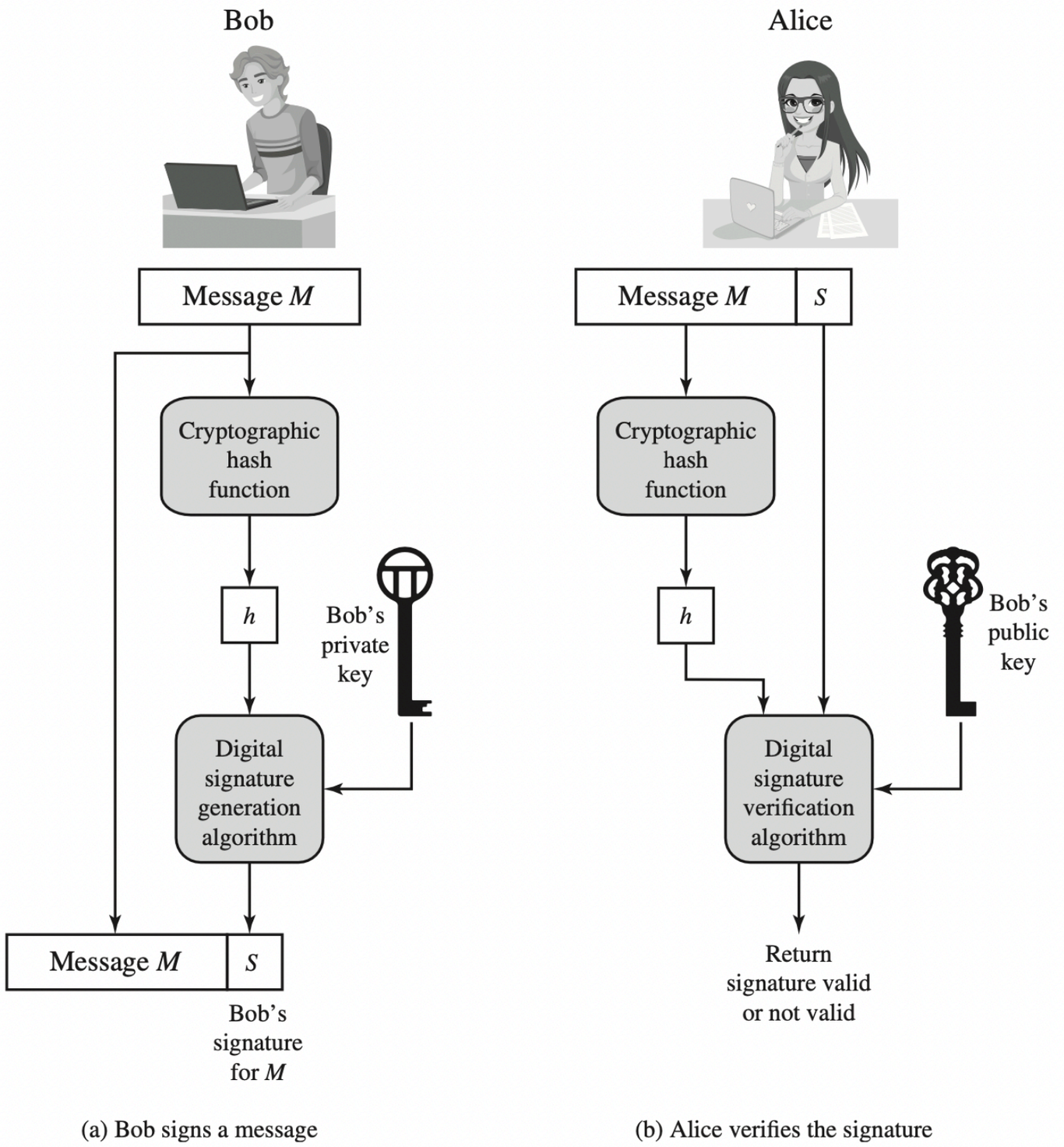
- **Digital Signature Algorithm** (DSA)
- **RSA Digital Signature Algorithm**: Basato sull'algoritmo di cifratura asimmetrica RSA
- **Elliptic Curve Digital Signature Algorithm** (ECDSA)

Firma Digitale

Struttura di un algoritmo di Digital Signature

Bob invia un messaggio ad Alice (ma non gli interessa della confidenzialità) e vuole assicurarsi che Alice sia sicura che il messaggio provenga da lui.

Bob usa una funzione di hash per generare il digest del messaggio, questo digest insieme alla **chiave privata** di Bob sono gli input dell'algoritmo di firma. Il risultato, che è la **firma**, viene concatenato alla fine del messaggio.



(a) Bob signs a message

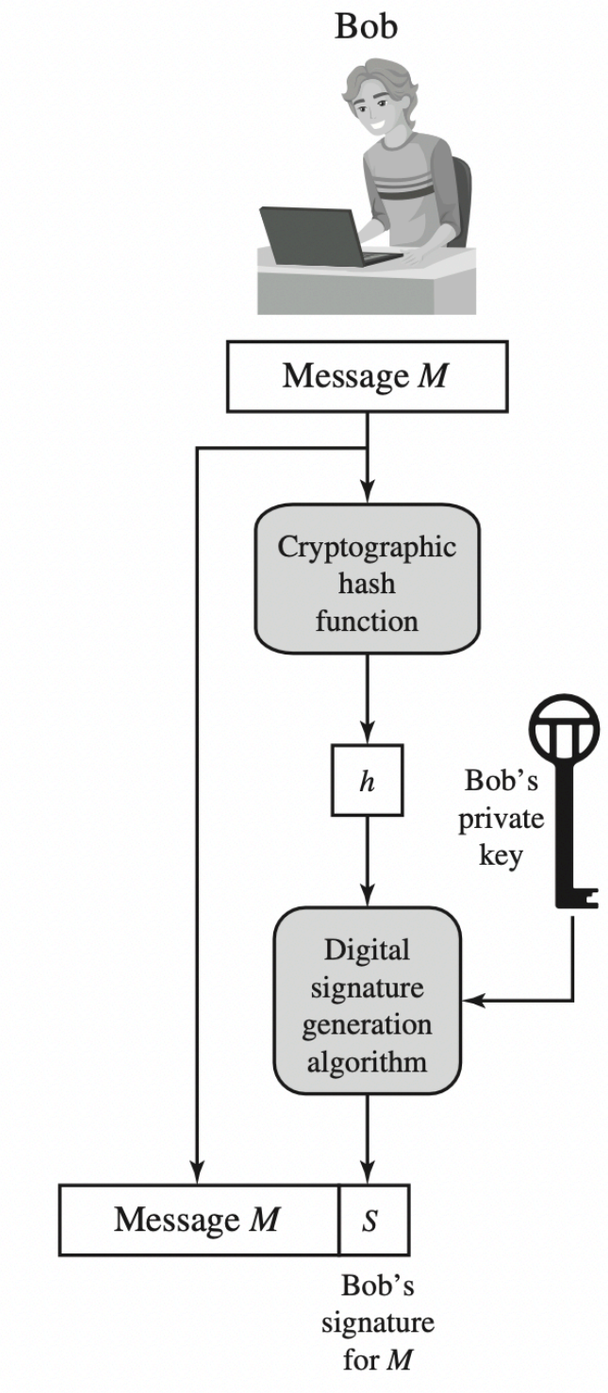
(b) Alice verifies the signature

Firma Digitale

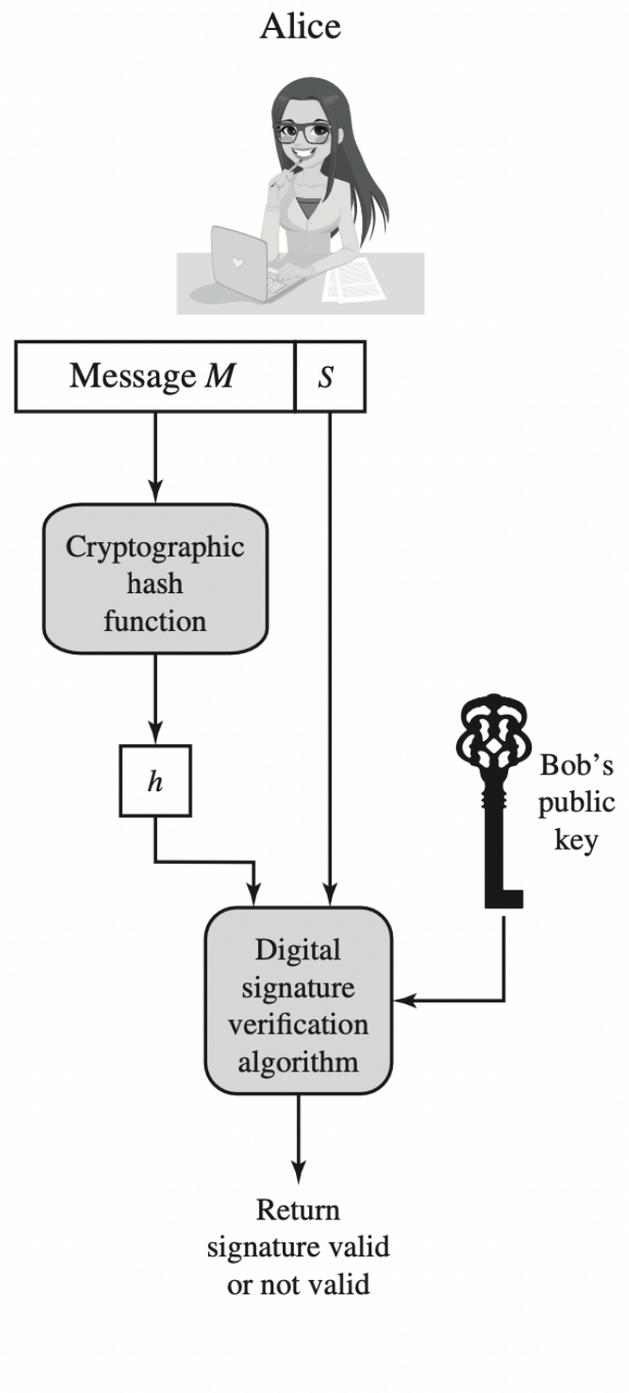
Struttura di un algoritmo di Digital Signature

Alice riceve il messaggio e la firma, calcola l'hash (digest) del messaggio e lo dà in input, insieme alla chiave pubblica di Bob, all'algoritmo di firma digitale.

L'algoritmo restituisce se la firma è valida o meno, Alice in questo modo riesce a capire se il messaggio proviene realmente da Bob o meno.



(a) Bob signs a message



(b) Alice verifies the signature

Key Management

Key Management

Certificati a chiave pubblica 1/5

La chiave pubblica deve essere trasmessa o condivisa. Purtroppo chiunque potrebbe fare un annuncio di questo tipo, rubando l'identità di Bob.

I **certificati a chiave pubblica** (o certificato pubblico) risolvono questo problema, accoppiano una chiave pubblica ad un User ID. Questo blocco di dati viene **firmato** da una **terza parte** fidata.

Key Management

Certificati a chiave pubblica 2/5

Questo certificato contiene anche il periodo di validità dello stesso, che in genere è di 1 o 2 anni.

La terza parte che emette il certificato si chiama **Certificate Authority** (CA) ed è una fonte autoritaria per tutta la base utenti.

Key Management

Certificati a chiave pubblica 3/5

1. L’utente crea una coppia di chiavi
2. L’utente prepara un certificato non firmato che include user ID e chiave pubblica
3. L’utente invia il certificato (ancora non firmato) alla CA
4. La CA firma il certificato:
    1. La CA calcola l’hash del certificato (non firmato)
    2. La CA firma il certificato con la propria chiave privata
5. La CA concatena la firma al certificato non firmato per produrre il **certificato firmato**

Key Management

Certificati a chiave pubblica 4/5

6. La CA invia il certificato firmato all'utente
7. L'utente può adesso inviare il proprio certificato agli altri
8. Gli altri utenti possono verificare la validità del certificato:
    1. Calcolando l'hash del certificato (senza la firma)
    2. Verificato la firma con la chiave pubblica della CA

Key Management

Certificati a chiave pubblica 5/5