

# Autenticazione dei messaggi e firma digitale

Alessandro Armando

Laboratorio di sicurezza informatica (CSec)  
DIBRIS, Università di Genova

Sicurezza del computer



## 1 Integrità del messaggio e funzioni di autenticazione

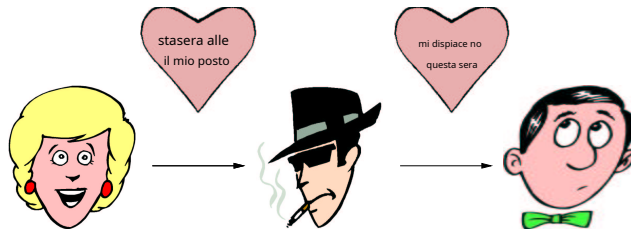
- Crittografia dei messaggi
- Codice di autenticazione del messaggio
- Funzioni di hash crittografico

## 2 Firma digitale

- Firma Digitale Diretta
- Firma Digitale Arbitrale

## 3 Abbastanza buona privacy (PGP)





L'autenticazione dei messaggi riguarda:

- proteggere l'integrità di un messaggio convalidare
- l'identità del mittente non ripudiare l'origine
- (risoluzione delle controversie)

Qualsiasi meccanismo di autenticazione dei messaggi o firma digitale si basa su una funzione di autenticazione per generare un *autenticatore*, cioè un valore utilizzato per autenticare un messaggio.

Prenderemo in considerazione le seguenti funzioni di autenticazione:

**Crittografia dei messaggi.** Il testo cifrato dell'intero messaggio funge da autenticatore.

**Codice di autenticazione del messaggio.** Una funzione del messaggio e una chiave segreta che produce un valore a lunghezza fissa che funge da autenticatore.

**Funzione di hash crittografico.** Una funzione che mappa un messaggio di qualsiasi lunghezza in a valore hash a lunghezza fissa, che funge da autenticatore.



## 1 Integrità del messaggio e funzioni di autenticazione

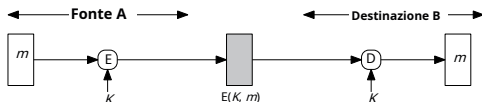
- Crittografia dei messaggi
- Codice di autenticazione del messaggio
- Funzioni di hash crittografico

## 2 Firma digitale

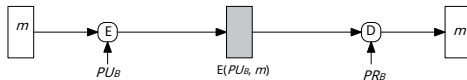
- Firma Digitale Diretta
- Firma Digitale Arbitrale

## 3 Abbastanza buona privacy (PGP)

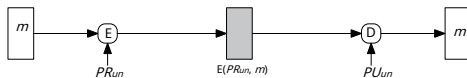
# Crittografia dei messaggi



(a) Crittografia simmetrica: riservatezza e autenticazione



(b) Crittografia a chiave pubblica: riservatezza



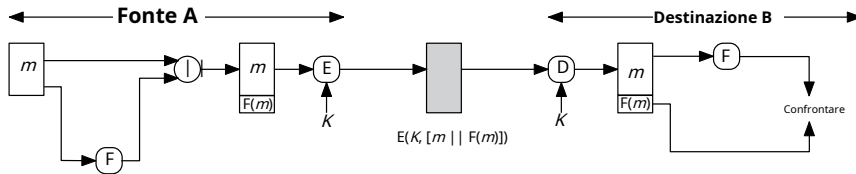
(c) Crittografia a chiave pubblica: autenticazione e sntuogn-unRTetuPRtuediazione



## Controllo degli errori con la crittografia dei messaggi

Per i precedenti approcci al successo ci devono essere alcuni mezzi automatici per determinare se il testo cifrato in arrivo viene decifrato in testo in chiaro intelligibile.

Una soluzione a questo problema è dare al testo in chiaro una struttura che sia facilmente riconoscibile ma che non possa essere replicata senza ricorrere alla funzione di cifratura, ad esempio aggiungendo un checksum al messaggio prima della cifratura.



(a) Controllo degli errori interni

**Tabella 11.1** Implicazioni sulla riservatezza e sull'autenticazione della crittografia dei messaggi (vedere la Figura 11.1)

$UN \rightarrow ESSERE(K, m)$

- Fornisce riservatezza
  - Solo A e B condividono  $K$
- Fornisce un grado di autenticazione
  - Potrebbe venire solo da A
  - Non è stato alterato durante il trasporto
  - Richiede formattazione/ridondanza
- Non fornisce la firma
  - Il destinatario potrebbe falsificare il messaggio
  - Il mittente potrebbe negare il messaggio

(a) Crittografia simmetrica

$UN \rightarrow ESSERE(PU_B, m)$

- Fornisce riservatezza
  - Solo B ha  $PR_B$  decifrare
- Non fornisce autenticazione
  - Qualsiasi partito potrebbe usare  $PU_B$  per crittografare il messaggio e affermare di essere A

(b) Crittografia a chiave pubblica (asimmetrica): riservatezza

$UN \rightarrow ESSERE(PR_{un}, m)$

- Fornisce autenticazione e firma
  - Solo A ha  $PR_{un}$  crittografare
  - Non è stato alterato durante il trasporto
  - Richiede formattazione/ridondanza
  - Qualsiasi parte può usare  $PU_{un}$  per verificare la firma

(c) Crittografia a chiave pubblica: autenticazione e firma

$UN \rightarrow ESSERE(PU_B, E(PR_{un}, m))$

- Fornisce riservatezza a causa di  $PU_B$
- Fornisce autenticazione e firma a causa di  $PR_{un}$

(d) Crittografia a chiave pubblica



## 1 Integrità del messaggio e funzioni di autenticazione

- Crittografia dei messaggi
- Codice di autenticazione del messaggio
- Funzioni di hash crittografico

## 2 Firma digitale

- Firma Digitale Diretta
- Firma Digitale Arbitrale

## 3 Abbastanza buona privacy (PGP)



- Una funzione MAC accetta un messaggio di dimensioni variabili  $m$  e una chiave segreta  $K$  come input e produce un output di dimensione fissa  $C(M, K)$ .
- È una funzione molti a uno poiché potenzialmente molti messaggi hanno lo stesso MAC ma trovarli è molto difficile
- $C(M, K)$  è chiamato *codice di autenticazione del messaggio* (MAC) o *checksum crittografico*.



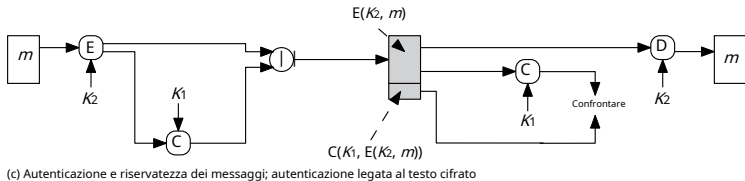
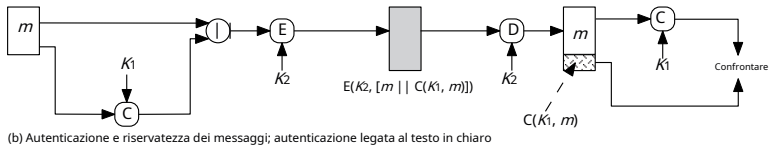
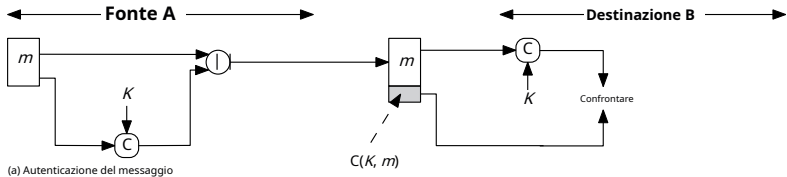
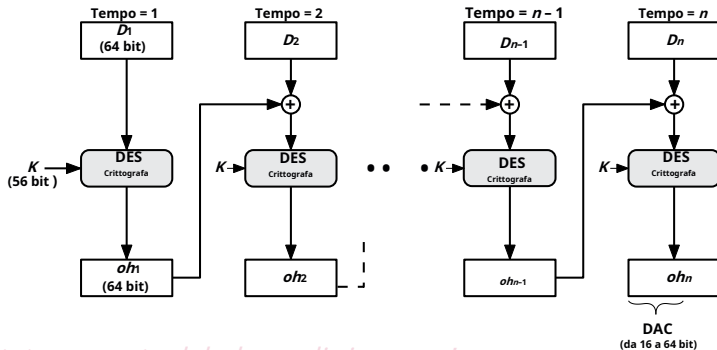


Figura 11.4 Usi di base del codice di autenticazione dei messaggi (MAC)

# Algoritmo di autenticazione dei dati

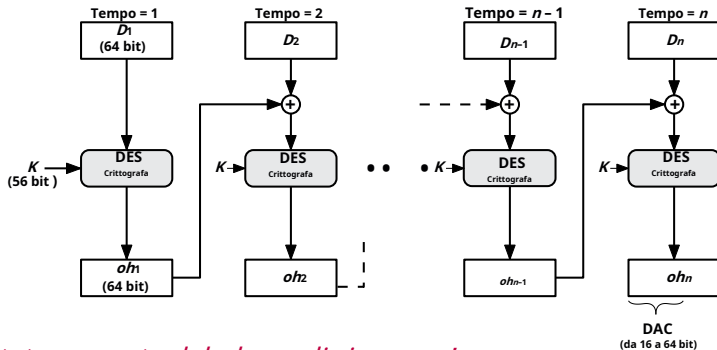
- Basato su DES
- Uno dei MAC più utilizzati da diversi anni Messaggio di
- input suddiviso in blocchi a 64 bit  $m = D_1 D_2 \dots D_n$



● Ma, *sono state scoperte debolezze di sicurezza!*

# Algoritmo di autenticazione dei dati

- Basato su DES
- Uno dei MAC più utilizzati da diversi anni Messaggio di
- input suddiviso in blocchi a 64 bit  $m = D_1 D_2 \dots D_n$



● Ma, *sono state scoperte debolezze di sicurezza!*

## 1 Integrità del messaggio e funzioni di autenticazione

- Crittografia dei messaggi
- Codice di autenticazione del messaggio
- Funzioni di hash crittografico

## 2 Firma digitale

- Firma Digitale Diretta
- Firma Digitale Arbitrale

## 3 Abbastanza buona privacy (PGP)

# Funzioni di hash crittografico

- Come con MAC, una funzione hash accetta un messaggio di dimensioni variabili  $m$  come input e produce un output di dimensione fissa  $h(m)$ .
- A differenza di MAC, una funzione hash non utilizza una chiave.
- $h(m)$  è chiamato *codice hash* (ma anche *riassunto del messaggio*).







# Requisiti per una funzione di hash crittografico

Lo scopo di una funzione di hash è produrre una "impronta digitale" di un file, messaggio o altro blocco di dati.

Per essere utile una funzione hash  $h$  deve avere le seguenti proprietà:

- 1  $h$  può essere applicato a un blocco di dati di qualsiasi
- 2 **dimensione.**  $h$  produce un output a lunghezza fissa.
- 3  $h(X)$  è relativamente facile da calcolare per qualsiasi dato  $X$ .
- 4 **(Proprietà a senso unico)** Per ogni dato valore  $s_i$ , è computazionalmente impossibile da trovare  $X$  tale che  $h(X) = s_i$ .
- 5 **(Debole resistenza alla collisione o resistenza pre-immagine)** Per qualsiasi valore  $X$ , è computazionalmente impossibile da trovare  $s_i \neq X$  tale che  $h(s_i) = h(X)$ .
- 6 **(Forte resistenza alla collisione o 2a resistenza pre-immagine)** È computazionalmente impossibile trovare qualsiasi coppia  $(x, y)$  tale che  $h(x) = h(y)$ .



# Requisiti per una funzione di hash crittografico

Lo scopo di una funzione di hash è produrre una "impronta digitale" di un file, messaggio o altro blocco di dati.

Per essere utile una funzione hash  $h$  deve avere le seguenti proprietà:

- 1  $h$  può essere applicato a un blocco di dati di qualsiasi
- 2 dimensione.  $h$  produce un output a lunghezza fissa.
- 3  $h(X)$  è relativamente facile da calcolare per qualsiasi dato  $X$ .
- 4 **(Proprietà a senso unico)** Per ogni dato valore  $s_i$ , è computazionalmente impossibile da trovare  $X$  tale che  $h(X) = s_i$ .
- 5 **(Debole resistenza alla collisione o resistenza pre-immagine)** Per qualsiasi valore  $X$ , è computazionalmente impossibile da trovare  $s_i \neq X$  tale che  $h(s_i) = h(X)$ .
- 6 **(Forte resistenza alla collisione o 2a resistenza pre-immagine)** È computazionalmente impossibile trovare qualsiasi coppia  $(x, y)$  tale che  $h(x) = h(y)$ .



# Requisiti per una funzione di hash crittografico

Lo scopo di una funzione di hash è produrre una "impronta digitale" di un file, messaggio o altro blocco di dati.

Per essere utile una funzione hash  $h$  deve avere le seguenti proprietà:

- 1  $h$  può essere applicato a un blocco di dati di qualsiasi
- 2 dimensione.  $h$  produce un output a lunghezza fissa.
- 3  $h(X)$  è relativamente facile da calcolare per qualsiasi dato  $X$ .
- 4 **(Proprietà a senso unico)** Per ogni dato valore  $s_i$ , è computazionalmente impossibile da trovare  $X$  tale che  $h(X) = s_i$ .
- 5 **(Debole resistenza alla collisione o resistenza pre-immagine)** Per qualsiasi valore  $X$ , è computazionalmente impossibile da trovare  $s_i \neq X$  tale che  $h(s_i) = h(X)$ .
- 6 **(Forte resistenza alla collisione o 2a resistenza pre-immagine)** È computazionalmente impossibile trovare qualsiasi coppia  $(x, y)$  tale che  $h(x) = h(y)$ .



# Requisiti per una funzione di hash crittografico

Lo scopo di una funzione di hash è produrre una "impronta digitale" di un file, messaggio o altro blocco di dati.

Per essere utile una funzione hash  $h$  deve avere le seguenti proprietà:

- 1  $h$  può essere applicato a un blocco di dati di qualsiasi
- 2 dimensione.  $h$  produce un output a lunghezza fissa.
- 3  $h(X)$  è relativamente facile da calcolare per qualsiasi dato  $X$ .
- 4 **(Proprietà a senso unico)** Per ogni dato valore  $s_i$ , è computazionalmente impossibile da trovare  $X$  tale che  $h(X) = s_i$ .
- 5 **(Debole resistenza alla collisione o resistenza pre-immagine)** Per qualsiasi valore  $X$ , è computazionalmente impossibile da trovare  $s_i \neq X$  tale che  $h(s_i) = h(X)$ .
- 6 **(Forte resistenza alla collisione o 2a resistenza pre-immagine)** È computazionalmente impossibile trovare qualsiasi coppia  $(x, y)$  tale che  $h(x) = h(y)$ .



# Requisiti per una funzione di hash crittografico

Lo scopo di una funzione di hash è produrre una "impronta digitale" di un file, messaggio o altro blocco di dati.

Per essere utile una funzione hash  $h$  deve avere le seguenti proprietà:

- 1  $h$  può essere applicato a un blocco di dati di qualsiasi dimensione.  $h$  produce un output a lunghezza fissa.
- 2  $h(X)$  è relativamente facile da calcolare per qualsiasi dato  $X$ .
- 3 **(Proprietà a senso unico)** Per ogni dato valore  $s_i$ , è computazionalmente impossibile da trovare  $X$  tale che  $h(X) = s_i$ .
- 4 **(Debole resistenza alla collisione o resistenza pre-immagine)** Per qualsiasi valore  $X$ , è computazionalmente impossibile da trovare  $s_i \neq X$  tale che  $h(s_i) = h(X)$ .
- 5 **(Forte resistenza alla collisione o 2a resistenza pre-immagine)** È computazionalmente impossibile trovare qualsiasi coppia  $(x, y)$  tale che  $h(x) = h(y)$ .



# Requisiti per una funzione di hash crittografico

Lo scopo di una funzione di hash è produrre una "impronta digitale" di un file, messaggio o altro blocco di dati.

Per essere utile una funzione hash  $h$  deve avere le seguenti proprietà:

- 1  $h$  può essere applicato a un blocco di dati di qualsiasi
- 2 dimensione.  $h$  produce un output a lunghezza fissa.
- 3  $h(X)$  è relativamente facile da calcolare per qualsiasi dato  $X$ .
- 4 **(Proprietà a senso unico)** Per ogni dato valore  $s_i$ , è computazionalmente impossibile da trovare  $X$  tale che  $h(X) = s_i$ .
- 5 **(Debole resistenza alla collisione o resistenza pre-immagine)** Per qualsiasi valore  $X$ , è computazionalmente impossibile da trovare  $s_i \neq X$  tale che  $h(s_i) = h(X)$ .
- 6 **(Forte resistenza alla collisione o 2a resistenza pre-immagine)** È computazionalmente impossibile trovare qualsiasi coppia  $(x, y)$  tale che  $h(x) = h(y)$ .



- La proprietà unidirezionale è importante nelle tecniche di autenticazione dei messaggi che comportano l'uso di un valore segreto, ad esempio in

$UN \rightarrow B: \text{mio}h(\text{mio}S)$       dove  $S$  è un segreto tra  $UN$  e  $B$

- La debole resistenza alle collisioni impedisce la contraffazione quando viene utilizzato un codice hash crittografato, ad esempio in

$UN \rightarrow B: \text{mio}E(K, H(m))$        $UN \rightarrow B: \text{mio}E(PR_{un}, h(m))$

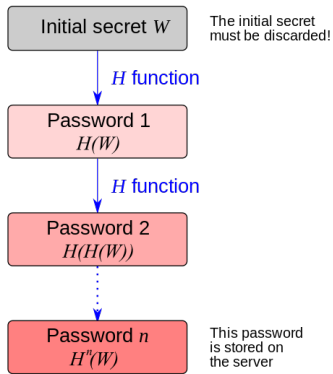
- Debole resistenza alle collisioni utile anche per proteggere i file di password
  - Per la password  $P$ , negozio  $h(P)$  nel file delle password.
  - Spesso combinato con *sale*  $S$ , cioè coppia di negozi ( $SH(SioP)$ ), per proteggersi da *dizionario attacchi*.

(Esempi di dizionari “utili” possono essere trovati qui  
<https://wiki.skullsecurity.org/Passwords>)

- Forte resistenza agli urti utile contro il *attacco di compleanno*.

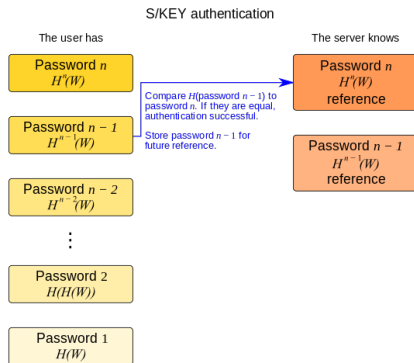
- Sistema di password monouso sviluppato da terminali stupidi e computer pubblici non attendibili che non richiede di digitare una password a lungo termine.
- Poiché ogni password viene utilizzata solo una volta, sono inutili per gli sniffer di password.
- Le password vengono stampate o calcolate da un dispositivo portatile (elemento sicuro)

## S/KEY password generation





- L'utente fornisce  $pwd_{io}$  per  $io = n-1, \dots, 1$ .
- Il server calcola  $h(pwd_{io})$  e confronta il risultato con  $pwd_{io+1}$ , che viene archiviato come riferimento sul server.
- Avviso, la prima password ( $pwd_n$ ) viene scartato dall'utente.



- Quante persone devono essere in una stanza tale che la probabilità  $P$  quello ha il tuo compleanno è  $p > 0.5$ ?
- Quante persone devono essere in una stanza tale che la probabilità  $P$  che due qualsiasi condividano lo stesso compleanno è  $p > 0.5$ ?
- **Generalizzazione:** Permettere  $h$  avere  $2^m$  possibili uscite.  $h$  deve essere applicato a  $2^{m/2}$  input in modo che la probabilità di una collisione sia maggiore di 0,5.



- Quante persone devono essere in una stanza tale che la probabilità  $P$  quello ha il tuo compleanno è  $p > 0.5$ ?  
La probabilità che uno di  $n$  la gente ha il tuo compleanno è  $n/365$ , quindi  $p > 0.5$  per  $n \geq 183$ .
- Quante persone devono essere in una stanza tale che la probabilità  $P$  che due qualsiasi condividano lo stesso compleanno è  $p > 0.5$ ?
- **Generalizzazione:** Permettere  $h$  avere  $2^m$  possibili uscite.  $h$  deve essere applicato a  $2^{m/2}$  input in modo che la probabilità di una collisione sia maggiore di 0,5.



- Quante persone devono essere in una stanza tale che la probabilità  $P$  quello ha il tuo compleanno è  $p > 0.5$ ?  
La probabilità che uno di  $n$  la gente ha il tuo compleanno è  $n/365$ , quindi  $p > 0.5$  per  $n \geq 183$ .
- Quante persone devono essere in una stanza tale che la probabilità  $P$  che due qualsiasi condividano lo stesso compleanno è  $p > 0.5$ ? **23 persone!**
- **Generalizzazione:** Permettere  $h$  avere  $2^m$  possibili uscite.  $h$  deve essere applicato a  $2^{m/2}$  input in modo che la probabilità di una collisione sia maggiore di 0,5.



- Quante persone devono essere in una stanza tale che la probabilità  $P$  quello ha il tuo compleanno è  $p > 0.5$ ?  
La probabilità che uno di  $n$  la gente ha il tuo compleanno è  $n/365$ , quindi  $p > 0.5$  per  $n \geq 183$ .
- Quante persone devono essere in una stanza tale che la probabilità  $P$  che due qualsiasi condividano lo stesso compleanno è  $p > 0.5$ ? **23 persone!**
- **Generalizzazione:** Permettere  $h$  avere  $2^m$  possibili uscite.  $h$  deve essere applicato a  $2^{m/2}$  input in modo che la probabilità di una collisione sia maggiore di 0,5.



Il numero di modi  $n$  possiamo avere  $K$  valori nell'intervallo  $1..n$  senza duplicati:

$$n = n \times (n-1) \times \dots \times (n-k+1) = \frac{n!}{(n-k)!}$$

Ci sono  $n_K$  modi per selezionare  $K$  articoli da  $1..n$  permettendo ripetizioni. Quindi probabilità di nessun duplicato durante la selezione  $K$  articoli della stessa gamma è

$$Q(n, k) = \frac{n!}{(n-k)! \times n_K}$$

Quindi,  $P(n, k)$  = probabilità di almeno un duplicato in  $K$  gli articoli sono

$$P(n, k) = 1 - Q(n, k) = 1 - \frac{n!}{(n-k)! \times n_K}$$

Può calcolare  $P(365, 23) = .5073$ .

In generale duplicati incontrati in  $oh(\frac{n}{2})$ . Se  $n = 2^m$ , poi  $oh(2^{m/2})$



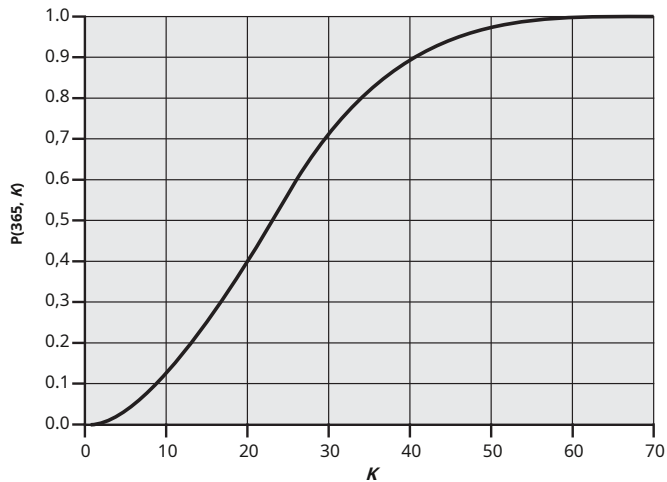


Figura 11.10 Il paradosso del compleanno

- Si potrebbe pensare che un codice hash a 64 bit sia sicuro. Resistenza preimmagine significa, in media  $2^{63}$  i messaggi devono essere provati.
- Attacco di compleanno per trovare collisioni
  - supponiamo  $UN$  è disposto a firmare un contratto con  $B$ . Quindi  $B$  prepara la versione  $X$ , buono per  $UN$  e versione  $s$ , che la manda in bancarotta.
    - $B$  genera messaggi "buoni"  $X_1, \dots$
    - Allo stesso modo unisce la generazione dei "cattivi"  $s_1, \dots$ . Si
    - ferma quando  $h(X_{i_0}) = h(s_{j_0})$ .
  - $UN$  segni  $h(X_{i_0})$ . Dopo  $B$  usa la firma per  $s_{j_0}$ .
- In media, il lavoro richiesto è dell'ordine di  $2^{32}$ .

Quindi raddoppia la dimensione dell'hash se è importante evitare le collisioni!





# Una lettera in 237 variazioni

Caro Antonio,

-Questa lettera è-  
Sto scrivendo - presentare -tu a -Sig.- Alfredo -P.-  
-a sib- - - -

Barton, il -  
-neo nominato- capo -  
-anziano- compratore di gioielli per - Nostro-  
-il-

Settentrionale -Area europea -prenderà- -il-  
-Europa -di isione- Lui-ha preso- terminato - - -

responsabilità per - Tutti - i nostri interessi in - orologi e gioielli -  
-l'intero- -gioielleria e orologi-

nel -la zona- Per favore -permettersi- Ciascuno ogni -  
-regione- dare - tutti i - aiuto lui -Maggio bisogno-  
-bisogni -

cercare fuori- moderno  
trovare la, maggior parte -aggiornato- linee per il - superiore- fine di il

mercato. Egli è -potenziato- ricevere per nostro conto - campioni -  
-autorizzato- -campioni- del

-più recente- orologi e gioielli- prodotti, su - limite -  
-più recente- gioielli e orologi- -soggetto- ad a -massimo-

di diecimila dollari. lui -ca rry- una copia firmata di questa lettera -  
-presa - documento-

come prova di identità. Un ordine con la sua firma, che è... in allegato-  
-Allegata- -

-autorizza-  
consente - di addebitare il costo a questa società al - sopra -  
-testa ufficio-

indirizzo. Noi -completamente- aspetta che il nostro - livello -  
-volume- degli ordini aumenterà in

il -a seguire - anno e -fiducia- che il nuovo appuntamento sarà -essere -  
-prossimo- -speranza- dimostrare-

-vantaggioso-  
un vantaggio -m S L G e Au zione e firma digitale

# Costruire una funzione hash crittografica

- Schema generale proposto da Merkle
- Dividi il messaggio in blocchi di dimensioni fisse  $m = s_0, \dots, s_{L-1}$  e applica

$$CV_0 = IV$$

$$CV_{io} = F(CV_{io-1}, s_{io-1})$$

per  $io = 1, \dots, L$

$$h(m) = CV_L$$

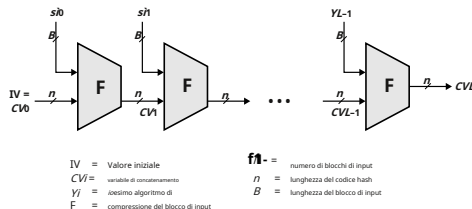


Figura 11.9 Struttura generale del codice hash sicuro

- **MD5** (Rivest): hash a 128 bit.  
Debolezza nota.
- **SHA** (Standard USA): hash a 160 bit.  
Assunto sicuro.




## 1 Integrità del messaggio e funzioni di autenticazione

- Crittografia dei messaggi
- Codice di autenticazione del messaggio
- Funzioni di hash crittografico

## 2 Firma digitale

- Firma Digitale Diretta
- Firma Digitale Arbitrale

## 3 Abbastanza buona privacy (PGP)

- L'autenticazione dei messaggi protegge due parti che si scambiano messaggi da terze parti.
- Però, *non protegge le due parti l'una contro l'altra!*
- Sono possibili diverse forme di controversia tra i due.
- Ad esempio, supponiamo che A invii un messaggio autenticato a B utilizzando uno schema per l'autenticazione del messaggio mostrato su . Potrebbero sorgere le seguenti controversie:
  - B potrebbe creare un messaggio diverso e affermare che proveniva da A.
  - A può negare l'invio del messaggio. Poiché è possibile per B falsificare un messaggio, non c'è modo di dimostrare che A abbia effettivamente inviato il messaggio.



Una firma digitale deve

- fornire i mezzi per verificare l'autore e la data e l'ora della firma autenticare
- il contenuto di al momento della firma
- essere verificabile da terzi, per risolvere controversie

# Requisiti su un meccanismo di firma digitale

- Una firma digitale deve essere un modello di bit che dipende dal messaggio che viene firmato.
- Una firma digitale deve utilizzare alcune informazioni univoche per il mittente, per prevenire sia la falsificazione che il rifiuto.
- Deve essere relativamente facile produrre la firma digitale.
- Deve essere relativamente facile riconoscere e verificare la firma digitale.
- Deve essere computazionalmente irrealizzabile forgiare una firma digitale, costruendo un nuovo messaggio per una firma digitale esistente o costruendo una firma digitale fraudolenta per un dato messaggio.
- Deve essere pratico conservare una copia della firma digitale in archivio.



## 1 Integrità del messaggio e funzioni di autenticazione

- Crittografia dei messaggi
- Codice di autenticazione del messaggio
- Funzioni di hash crittografico

## 2 Firma digitale

- Firma Digitale Diretta
- Firma digitale arbitrale

## 3 Abbastanza buona privacy (PGP)



- La destinazione deve conoscere la chiave pubblica della
- sorgente. La firma è formata o
  - crittografando l'intero messaggio con la chiave privata del mittente come in Figura (c) su o diapositiva 6
  - crittografando il codice hash del messaggio con la chiave privata del mittente come in Figura (c) su diapositiva 16.
- La sicurezza dello schema dipende dalla sicurezza della chiave privata del mittente.
- Ogni messaggio firmato dovrebbe contenere un timestamp (data e ora) e le chiavi compromesse dovrebbero essere tempestivamente segnalate all'autorità centrale.
- Ma un avversario può rubare una chiave privata alla volta  $T$  e poi falsificare messaggi timbrati con un tempo prima  $T$ ...





## 1 Integrità del messaggio e funzioni di autenticazione

- Crittografia dei messaggi
- Codice di autenticazione del messaggio
- Funzioni di hash crittografico

## 2 Firma digitale

- Firma Digitale Diretta
- Firma digitale arbitrale

## 3 Abbastanza buona privacy (PGP)

Il problema associato alle firme digitali dirette può essere affrontato utilizzando un arbitro (terzo di fiducia).

**Tabella 13.1 Tecniche arbitrali di firma digitale**

(1)  $X \rightarrow UN: m \parallel E(K_{xa}, [ID_X \parallel H(m)])$   
(2)  $UN \rightarrow Y: E(K_{ay}, [ID_X \parallel m \parallel E(K_{xa}, [ID_X \parallel H(m)])]) \parallel T$

(a) Crittografia convenzionale, l'arbitro vede il messaggio

(1)  $X \rightarrow UN: ID_X \parallel E(K_{xy}, m) \parallel E(K_{xa}, [ID_X \parallel LUI(K_{xy}, m)])$   
(2)  $UN \rightarrow Y: E(K_{ay}, [ID_X \parallel E(K_{xy}, m)]) \parallel E(K_{xa}, [ID_X \parallel LUI(K_{xy}, m)]) \parallel T$

(b) Crittografia convenzionale, l'arbitro non vede il messaggio

(1)  $X \rightarrow UN: ID_X \parallel E(PR_X, [ID_X \parallel E(PU_{Si}, E(PR_X, m))])$   
(2)  $UN \rightarrow Y: E(PR_{un}, [ID_X \parallel E(PU_{Si}, E(PR_X, m))]) \parallel T$

(c) Crittografia a chiave pubblica, l'arbitro non vede il messaggio

Notazione:

$X$  = mittente

$m$  = Messaggio



## 1 Integrità del messaggio e funzioni di autenticazione

- Crittografia dei messaggi
- Codice di autenticazione del messaggio
- Funzioni di hash crittografico

## 2 Firma digitale

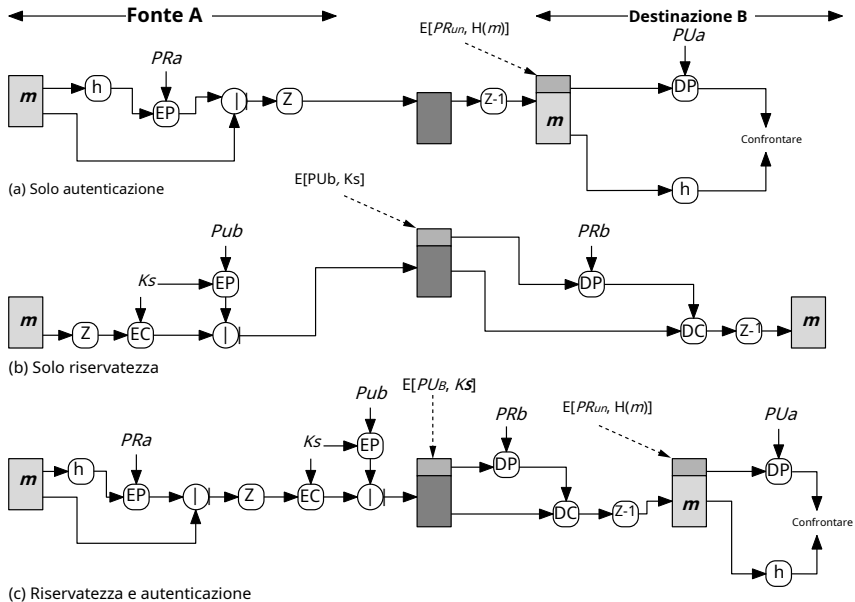
- Firma Digitale Diretta
- Firma Digitale Arbitrale

## 3 Abbastanza buona privacy (PGP)

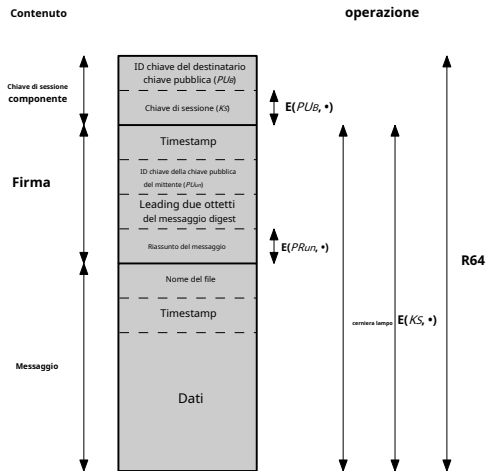
**Tabella 15.1 Riepilogo dei servizi PGP**

Funzione	Algoritmi utilizzati	Descrizione
Firma digitale	DSS/SHA o RSA/SHA	Un codice hash di un messaggio viene creato utilizzando SHA-1. Questo digest del messaggio è crittografato utilizzando DSS o RSA con la chiave privata del mittente e incluso con il Messaggio.
Crittografia dei messaggi	CAST o IDEA o Triplo DES . a tre tasti con Diffie-Hellman o RSA	Un messaggio viene crittografato utilizzando CAST-128 o IDEA o 3DES con una chiave di sessione monouso generata dal mittente. La chiave di sessione viene crittografata utilizzando Diffie-Hellman o RSA con la chiave pubblica del destinatario e inclusa nel messaggio.
Compressione	cerniera lampo	Un messaggio può essere compresso, per l'archiviazione o la trasmissione, utilizzando ZIP.
Compatibilità e-mail	Conversione Radix 64	Per fornire trasparenza alle applicazioni di posta elettronica, un messaggio crittografato può essere convertito in una stringa ASCII utilizzando la conversione radix 64.
Segmentazione	—	Per soddisfare i limiti di dimensione massima dei messaggi, PGP esegue la segmentazione e il riassemblaggio.





**Figura 15.1 Funzioni crittografiche PGP**



**Notazione:**

$E(PU_B, \bullet)$  = crittografia con la chiave pubblica dell'utente b  
 $E(PU_A, \bullet)$  = crittografia con chiave privata dell'utente a  
 $E(KS, \bullet)$  = crittografia con chiave di sessione = funzione di

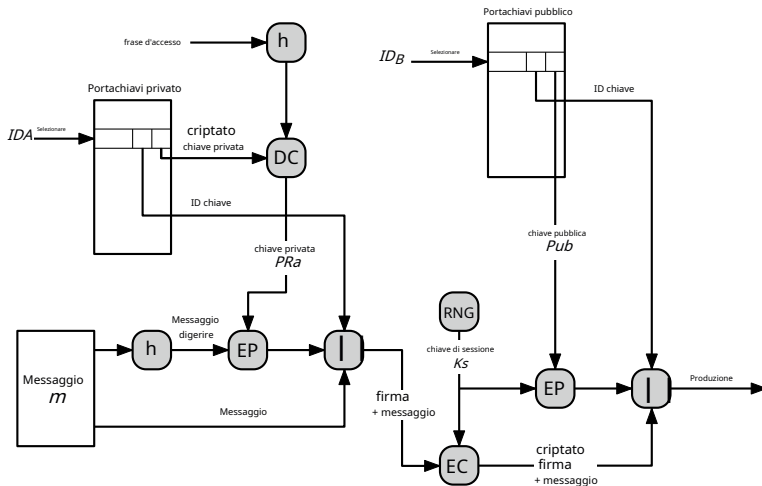
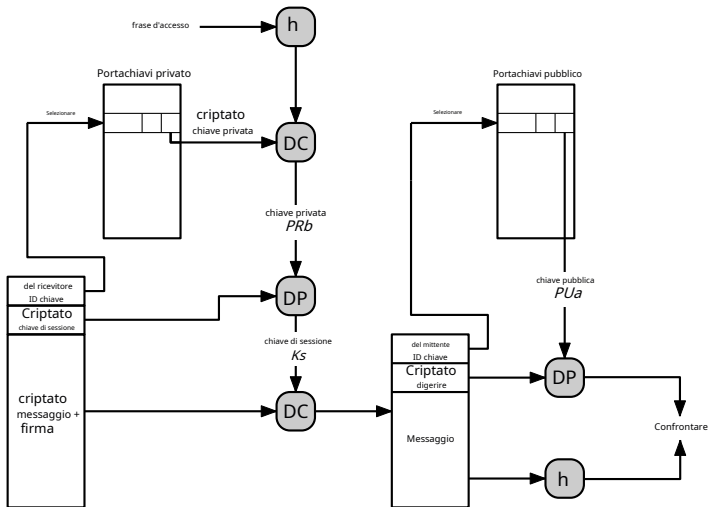


Figura 15.5 Generazione di messaggi PGP (da utente A a utente B; nessuna compressione o conversione radix 64)



**Figura 15.6** Ricezione messaggi PGP (da utente A a utente B; nessuna compressione o conversione radix 64)