




cefriel
 Consorzio per la Formazione e la Ricerca
 in Ingegneria dell'Informazione
 Politecnico di Milano




**POLITECNICO
 DI MILANO**

Funzioni di hash sicure: MD5 e SHA-1

Davide Cerri
 CEFRIEL - Politecnico di Milano
 cerri@cefriel.it
<http://www.cefriel.it/~cerri/>



Funzioni di hash



- Una **funzione di hash** (o message digest) è una funzione H che, dato un input M di dimensione qualsiasi, produce un output h (l'hash) di **dimensione fissa**:

$$h = H(M)$$
- Lo scopo di una funzione di hash è di creare un'**impronta** (fingerprint) del messaggio.
 - ▶ Dato che l'output ha lunghezza fissa (in genere dell'ordine di una o poche centinaia di bit), esisteranno necessariamente diversi messaggi che generano lo stesso hash... ma non è detto che si riesca a trovarli...

Funzioni di hash sicure: MD5 e SHA-1

- 2 -

Davide Cerri



Unidirezionalità



- Una funzione di hash sicura deve essere **unidirezionale** (one-way), cioè:
 - ▶ dato un qualsiasi M , deve essere computazionalmente semplice calcolare $H(M)$;
 - ▶ dato un qualsiasi h , deve essere computazionalmente impraticabile calcolare un M tale che $h = H(M)$;
- L'idea è che l'hash di un messaggio sia in sostanza una corrispondenza pseudocasuale, per cui non è possibile prevedere in alcun modo il risultato se non calcolando l'hash.
 - ▶ Messaggi diversi, anche per un solo bit, devono generare hash **non correlati**.



Assenza di collisioni (1)



- Una funzione di hash sicura deve essere **senza collisioni** (collision-resistant):
 - ▶ dato un qualsiasi M , deve essere computazionalmente impraticabile trovare un $M' \neq M$ tale che $H(M') = H(M)$ (funzione **debolmente senza collisioni**);
 - ▶ deve essere computazionalmente impraticabile trovare una coppia (M, M') , con $M' \neq M$, tale che $H(M) = H(M')$ (funzione **fortemente senza collisioni**).
- L'idea è che, anche se necessariamente esistono messaggi diversi che producono lo stesso hash, non è praticabile trovarli.



Assenza di collisioni (2)




- Le due proprietà sull'assenza di collisioni (debole e forte), corrispondono a due diversi tipi di attacchi a forza bruta:
 - ▶ **attacco a forza bruta "semplice"**: trovare un messaggio che produca un dato hash (cioè un hash uguale a quello di un messaggio dato);
 - ▶ **attacco del compleanno** (birthday attack): trovare due messaggi che producano lo stesso hash, indipendentemente dal valore di questo hash.
- La complessità dei due attacchi è molto diversa!
 - ▶ se l'hash è lungo m bit, la complessità del primo è 2^m , quella del secondo è $2^{m/2}$.




Che cosa NON è un hash



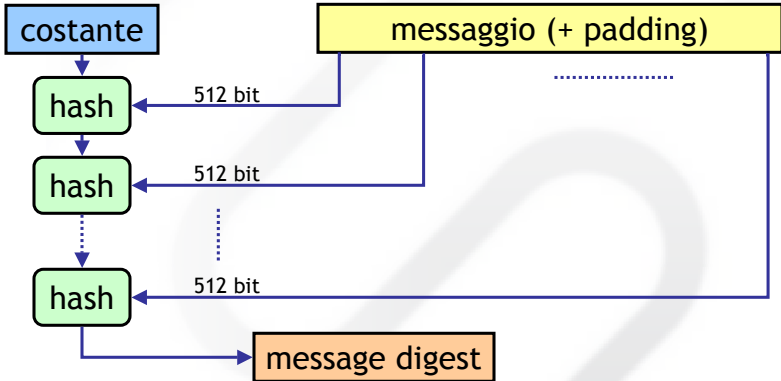
- Un hash **non è né una firma né un MAC** (Message Authentication Code), di per sé non garantisce autenticazione e/o integrità.
 - ▶ Nel calcolo di un hash non si inserisce nessuna informazione segreta, per cui chiunque può generare l'hash corretto di qualunque messaggio.
- Una funzione di hash **non è un algoritmo di cifratura**.
 - ▶ Calcolare l'hash di un messaggio non equivale a cifrarlo: il calcolo di un hash non include nessuna chiave, e soprattutto è un'operazione **non invertibile**.




MD5 e SHA-1




Le due funzioni di hash sicure attualmente più diffuse sono **MD5** e **SHA-1**, che hanno una struttura simile.



Funzioni di hash sicure: MD5 e SHA-1
- 7 -
Davide Cerri




MD5




- **MD5** (Message Digest 5) è stato progettato da Ron Rivest, ed è definito in RFC 1321 (1992).
- Il messaggio viene elaborato a blocchi di 512 bit.
- L'hash è di **128 bit**.
- Ad ogni iterazione si calcola una funzione che prende in ingresso il blocco corrente del messaggio e il valore dell'hash all'iterazione precedente.
- L'hash finale è quello risultante dall'ultima iterazione.
- Non è più considerato molto sicuro (128 bit di hash non sono molti).

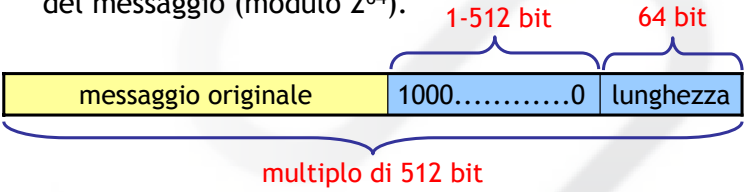
Funzioni di hash sicure: MD5 e SHA-1
- 8 -
Davide Cerri



MD5: padding




- Prima di iniziare l'elaborazione, si aggiunge al messaggio un **padding** in modo che la lunghezza totale risulti un multiplo di 512 bit:
 - ▶ si aggiunge un bit a 1 e poi tanti bit a 0 quanto basta perché la lunghezza risulti di 64 bit minore rispetto a un multiplo di 512 bit (se la lunghezza originale è già corretta si aggiungono comunque 512 bit);
 - ▶ si aggiungono 64 bit contenenti la lunghezza originale del messaggio (modulo 2^{64}).




Funzioni di hash sicure: MD5 e SHA-1

- 9 -

Davide Cerri



Elaborazione MD5 (1)




- Un'iterazione MD5 (cioè l'elaborazione su un blocco del messaggio) è composta da **4 passi**.
- Indichiamo con:
 - ▶ d_i : i -esima parola (32 bit) del digest ($i=0-3$);
 - ▶ m_i : i -esima parola di un blocco di messaggio ($i=0-15$);
 - ▶ $T_i = \lfloor 2^{32} \cdot \sin i \rfloor$, ($i=1-64$);
 - ▶ $\ll x$: rotazione sinistra di x posizioni (su 32 bit);
 - ▶ $+$: somma modulo 2^{32} ;
 - ▶ $\wedge, \vee, \sim, \oplus$: rispettivamente AND, OR, NOT, XOR (calcolati bit a bit).


Funzioni di hash sicure: MD5 e SHA-1

- 10 -

Davide Cerri



Elaborazione MD5 (2)




- Il valore del digest viene così **inizializzato**:
 $d_0 = 0x67452301$, $d_1 = 0xEFCDAB89$,
 $d_2 = 0x98BADCFE$, $d_3 = 0x10325476$.
- Ogni passo prende in ingresso il **valore corrente del digest** e il **blocco corrente del messaggio**, e modifica (fornendolo in uscita) il valore corrente del digest.
- Il valore del digest in uscita dal quarto passo viene poi **sommato** (indipendentemente per le 4 parole d_0, d_1, d_2, d_3) al valore di ingresso al primo passo dell'iterazione corrente; il risultato è il valore di uscita dell'iterazione.


Funzioni di hash sicure: MD5 e SHA-1

- 11 -

Davide Cerri



Elaborazione MD5 (3)



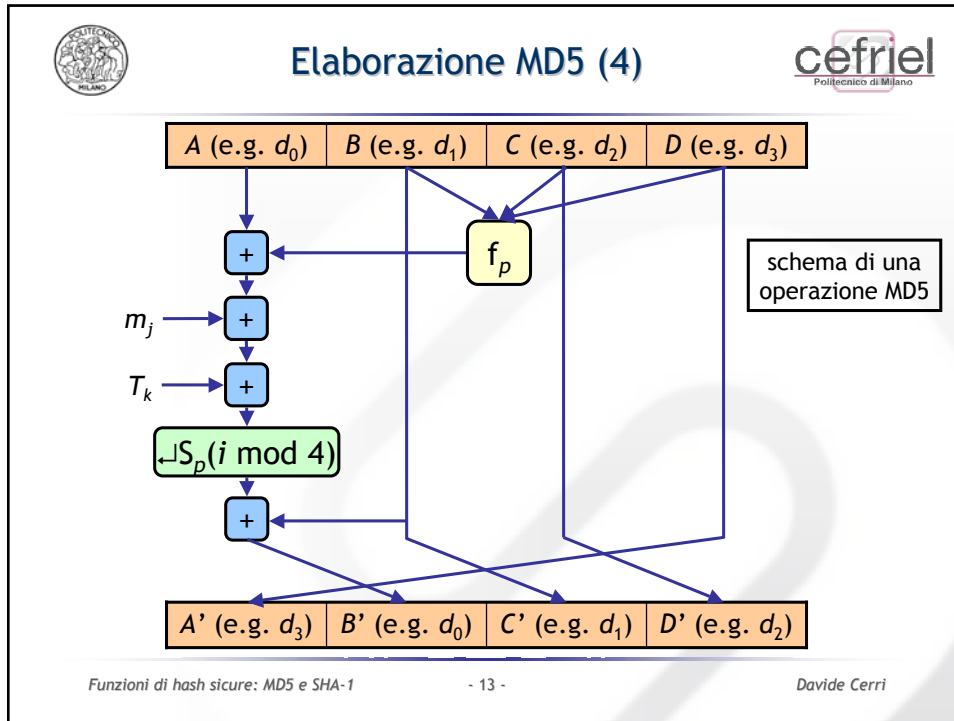
- Ogni passo è composto da **16 operazioni**.
- La i -esima ($i=0-15$) operazione ha la forma:

$$d_{(-i) \bmod 4} = d_{(1-i) \bmod 4} + (d_{(-i) \bmod 4} + f_p(d_{(1-i) \bmod 4}, d_{(2-i) \bmod 4}, d_{(3-i) \bmod 4}) + m_j + T_k) \ll S_p(i \bmod 4)$$
- ogni operazione:
 - ▶ aggiorna uno dei 4 blocchi del digest (in ogni passo ogni blocco del digest viene aggiornato 4 volte);
 - ▶ utilizza uno dei 16 blocchi del messaggio m_j (in ogni passo ogni blocco viene usato una volta);
 - ▶ utilizza una delle 64 costanti T_k ;
 - ▶ utilizza una funzione f_p diversa per ogni passo;
 - ▶ effettua una rotazione il cui valore è determinato da una funzione S_p diversa per ogni passo;

Funzioni di hash sicure: MD5 e SHA-1

- 12 -

Davide Cerri



Elaborazione MD5: passo 1

cefriel
Politecnico di Milano

- Definiamo $F(x,y,z) = (x \wedge y) \vee (\neg x \wedge z)$.
- Per $i=0-15$ si esegue:

$$d_{(-i) \bmod 4} = d_{(1-i) \bmod 4} + (d_{(-i) \bmod 4} + F(d_{(1-i) \bmod 4}, d_{(2-i) \bmod 4}, d_{(3-i) \bmod 4}) + m_i + T_{i+1}) \ll S_1(i \bmod 4)$$
 dove $S_1(0) = 7, S_1(1) = 12, S_1(2) = 17, S_1(3) = 22$.
- Le prime 5 di 16 operazioni sono quindi:

$$d_0 = d_1 + (d_0 + F(d_1, d_2, d_3) + m_0 + T_1) \ll 7 \quad (i=0)$$

$$d_3 = d_0 + (d_3 + F(d_0, d_1, d_2) + m_1 + T_2) \ll 12 \quad (i=1)$$

$$d_2 = d_3 + (d_2 + F(d_3, d_0, d_1) + m_2 + T_3) \ll 17 \quad (i=2)$$

$$d_1 = d_2 + (d_1 + F(d_2, d_3, d_0) + m_3 + T_4) \ll 22 \quad (i=3)$$

$$d_0 = d_1 + (d_0 + F(d_1, d_2, d_3) + m_4 + T_5) \ll 7 \quad (i=4)$$

Funzioni di hash sicure: MD5 e SHA-1 - 14 - Davide Cerri



Elaborazione MD5: passo 2



- Definiamo $G(x,y,z) = (x \wedge z) \vee (y \wedge \sim z)$.
- Per $i=0-15$ si esegue:

$$d_{(-i) \bmod 4} = d_{(1-i) \bmod 4} + (d_{(-i) \bmod 4} + G(d_{(1-i) \bmod 4}, d_{(2-i) \bmod 4}, d_{(3-i) \bmod 4}) + m_{(5i+1) \bmod 16} + T_{i+17}) \ll S_2(i \bmod 4)$$

dove $S_2(0) = 5, S_2(1) = 9, S_2(2) = 14, S_2(3) = 20$.

- Le prime 5 di 16 operazioni sono quindi:

$$d_0 = d_1 + (d_0 + G(d_1, d_2, d_3) + m_1 + T_{17}) \ll 5 \quad (i=0)$$

$$d_3 = d_0 + (d_3 + G(d_0, d_1, d_2) + m_6 + T_{18}) \ll 9 \quad (i=1)$$

$$d_2 = d_3 + (d_2 + G(d_3, d_0, d_1) + m_{11} + T_{19}) \ll 14 \quad (i=2)$$

$$d_1 = d_2 + (d_1 + G(d_2, d_3, d_0) + m_0 + T_{20}) \ll 20 \quad (i=3)$$

$$d_0 = d_1 + (d_0 + G(d_1, d_2, d_3) + m_5 + T_{21}) \ll 5 \quad (i=4)$$



Elaborazione MD5: passo 3



- Definiamo $H(x,y,z) = x \oplus y \oplus z$.
- Per $i=0-15$ si esegue:

$$d_{(-i) \bmod 4} = d_{(1-i) \bmod 4} + (d_{(-i) \bmod 4} + H(d_{(1-i) \bmod 4}, d_{(2-i) \bmod 4}, d_{(3-i) \bmod 4}) + m_{(3i+5) \bmod 16} + T_{i+33}) \ll S_3(i \bmod 4)$$

dove $S_3(0) = 4, S_3(1) = 11, S_3(2) = 16, S_3(3) = 23$.

- Le prime 5 di 16 operazioni sono quindi:


$$d_0 = d_1 + (d_0 + H(d_1, d_2, d_3) + m_5 + T_{33}) \ll 4 \quad (i=0)$$

$$d_3 = d_0 + (d_3 + H(d_0, d_1, d_2) + m_8 + T_{34}) \ll 11 \quad (i=1)$$


$$d_2 = d_3 + (d_2 + H(d_3, d_0, d_1) + m_{11} + T_{35}) \ll 16 \quad (i=2)$$

$$d_1 = d_2 + (d_1 + H(d_2, d_3, d_0) + m_{14} + T_{36}) \ll 23 \quad (i=3)$$

$$d_0 = d_1 + (d_0 + H(d_1, d_2, d_3) + m_1 + T_{37}) \ll 4 \quad (i=4)$$



Elaborazione MD5: passo 4




- Definiamo $l(x,y,z) = y \oplus (x \vee \sim z)$.
- Per $i=0-15$ si esegue:


$$d_{(-i) \bmod 4} = d_{(1-i) \bmod 4} + (d_{(-i) \bmod 4} + l(d_{(1-i) \bmod 4}, d_{(2-i) \bmod 4}, d_{(3-i) \bmod 4}) + m_{(7i) \bmod 16} + T_{i+49}) \ll S_4(i \bmod 4)$$
 dove $S_4(0) = 6, S_4(1) = 10, S_4(2) = 15, S_4(3) = 21$.
- Le prime 5 di 16 operazioni sono quindi:

$d_0 = d_1 + (d_0 + l(d_1, d_2, d_3) + m_0 + T_{49}) \ll 6$	$(i=0)$
$d_3 = d_0 + (d_3 + l(d_0, d_1, d_2) + m_7 + T_{50}) \ll 10$	$(i=1)$
$d_2 = d_3 + (d_2 + l(d_3, d_0, d_1) + m_{14} + T_{51}) \ll 15$	$(i=2)$
$d_1 = d_2 + (d_1 + l(d_2, d_3, d_0) + m_5 + T_{52}) \ll 21$	$(i=3)$
$d_0 = d_1 + (d_0 + l(d_1, d_2, d_3) + m_{12} + T_{53}) \ll 6$	$(i=4)$

Funzioni di hash sicure: MD5 e SHA-1
- 17 -
Davide Cerri




SHA-1




- **SHA-1** (Secure Hash Algorithm 1) è stato progettato dal NIST (l'algorithmo SHA originale è stato sostituito con SHA-1 dal NIST stesso per via di una vulnerabilità non pubblicata).
- Il messaggio (che deve essere di lunghezza inferiore a 2^{64} bit) viene elaborato a blocchi di 512 bit.
- L'hash è di **160 bit**.
- Ad ogni iterazione si calcola una funzione che dipende dal blocco corrente del messaggio e dal valore dell'hash all'iterazione precedente; il risultato viene sommato (per singola parola) al valore dell'iterazione precedente.

Funzioni di hash sicure: MD5 e SHA-1
- 18 -
Davide Cerri



SHA-1: inizializzazione




- Al messaggio viene aggiunto un **padding** (uguale a quello di MD5)
- Le 5 parole di 32 bit del digest (A, B, C, D, E) vengono inizializzate con: $A = 0x67452301$, $B = 0xEFCDAB89$, $C = 0x98BADCFE$, $D = 0x10325476$, $E = 0xC3D2E1F0$.
- Ogni iterazione calcola dei valori di A, B, C, D, E che vengono **sommati** ai precedenti prima di passare all'iterazione successiva.
- Alla fine, la concatenazione di A, B, C, D, E costituisce il digest del messaggio.


Funzioni di hash sicure: MD5 e SHA-1

- 19 -

Davide Cerri



SHA-1: schema iterazione (1)




- Il blocco corrente di messaggio (512 bit) viene utilizzato per costruire una stringa di 5×512 bit (80 parole di 32 bit).
 - ▶ Le prime 16 parole coincidono con il blocco del messaggio, le successive sono calcolate così:

$$W_i = (W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16}) \ll 1 \quad (i=16-79)$$
 - la rotazione di un bit è stata aggiunta in SHA-1 (è l'unica differenza tra SHA e SHA-1).
- Si scorre la stringa di 80 parole una parola alla volta (80 operazioni), calcolando i nuovi valori A', B', C', D', E' . Alla fine, i nuovi valori verranno aggiunti agli A, B, C, D, E precedenti.


Funzioni di hash sicure: MD5 e SHA-1

- 20 -

Davide Cerri



SHA-1: schema iterazione (2)



- Ogni iterazione è composta da 80 operazioni.
Per $i=0-79$:
 - ▶ B', C', D', E' sono calcolati così:
 $B' = A \quad C' = B \ll 30 \quad D' = C \quad E' = D$
 - ▶ A' viene calcolato così:
 $A' = E + (A \ll 5) + W_i + K_i + f_i(B, C, D)$
 dove K_i vale:

$\lfloor 2^{30} \sqrt{2} \rfloor$ ($i = 0-19$)	$\lfloor 2^{30} \sqrt{3} \rfloor$ ($i = 20-39$)
$\lfloor 2^{30} \sqrt{5} \rfloor$ ($i = 40-59$)	$\lfloor 2^{30} \sqrt{10} \rfloor$ ($i = 60-79$)

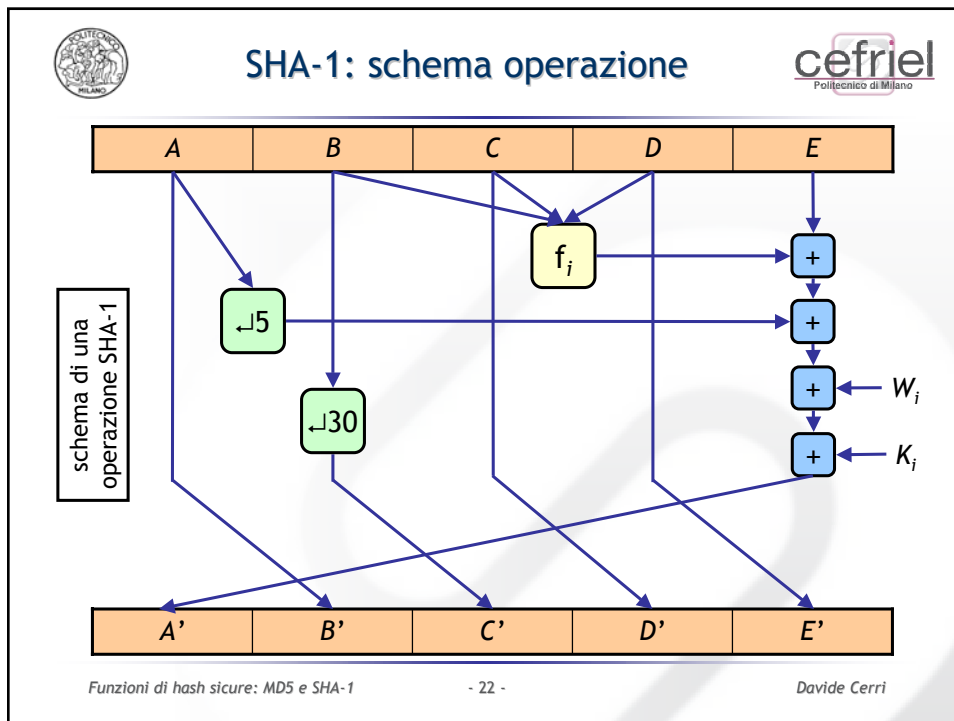
 e $f_i(B, C, D)$ vale:


$(B \wedge C) \vee (\sim B \wedge D)$	($i = 0-19$)
$B \oplus C \oplus D$	($i = 20-39, i = 60-79$)
$(B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$	($i = 40-59$)

Funzioni di hash sicure: MD5 e SHA-1


- 21 -

Davide Cerri





MD5 vs. SHA-1



	MD5	SHA-1
lunghezza hash	128 bit	160 bit
lunghezza massima messaggio	illimitata	$2^{64} - 1$ bit
dimensione blocco messaggio	512 bit	512 bit
numero operazioni	64	80
numero costanti additive	64	4
numero funzioni primitive	4	4 (di cui 2 uguali)

Funzioni di hash sicure: MD5 e SHA-1
- 23 -
Davide Cerri