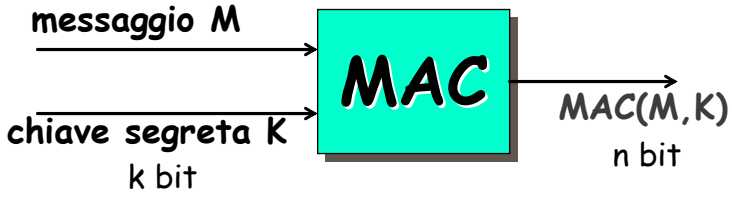
Message Authentication Code

Barbara Masucci

Dipartimento di Informatica ed Applicazioni
Università di Salerno

masucci@dia.unisa.it
http://www.dia.unisa.it/professori/masucci

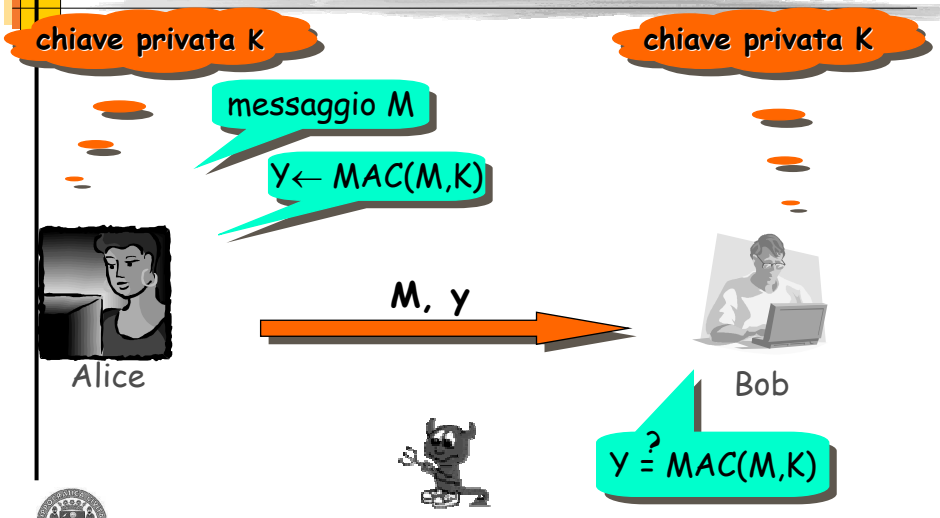
Message Authentication Code (MAC)



Applicazioni

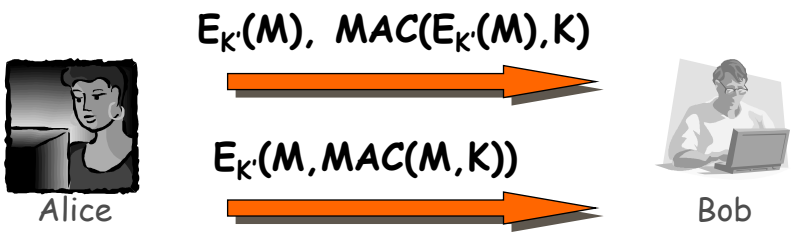
- Autenticità del messaggio **M**
- Integrità del messaggio M

Utilizzo MAC



MAC + confidenzialità

- MAC fornisce solo autenticazione ed integrità
- Se si vuole confidenzialità, si può cifrare prima/dopo con una diversa chiave condivisa K'



Sicurezza

- Cosa si intende per sicurezza di uno schema di un MAC?
- Dobbiamo definire
  - Tipo di attacco
  - Scopo dell'attacco

Tipo di attacco

- Known Message Attack
  - Oscar conosce una lista di messaggi ed i relativi MAC
- Chosen Message Attack
  - Oscar sceglie dei messaggi e chiede ad Alice (o Bob) di computarne i MAC
- Adaptive Chosen Message Attack
  - Come nel Chosen Message Attack, ma le scelte dipendono dalle risposte precedenti

Scopo dell'attacco

- Total break
  - Determinare la chiave K
- Selective forgery
  - Dato un messaggio M, determinare y tale che y=MAC(M,K)
- Existential forgery
  - Determinare una coppia (M,y) tale che y=MAC(M,K)

Ricerca esaustiva sulla chiave

- Date le coppie (M_i,y_i), con $y_i=MAC(M_i,k)$, determiniamo K
  - Supponiamo che sia k > n
    - Prova tutte le 2^k chiavi sulla coppia (M_1,y_1): circa 2^{k-n} match
    - Prova le 2^{k-n} chiavi precedenti sulla coppia (M_2,y_2): circa 2^{k-2n} match
    - …
  - In generale, se k=β·n, necessari circa β round
- Esempio: k=80 bit, n=32 bit
  - Round 1: circa 2^{48} match
  - Round 2: circa 2^{16} match
  - Round 3: 1 match

Ricerca esaustiva sul valore del MAC

- Dato M determiniamo y=MAC(M,K), senza conoscere K
  - Scegli a caso y, prob. successo: $1/2^n$
  - Ma come controlliamo che sia il valore giusto, senza avere K?

Ricerca esaustiva

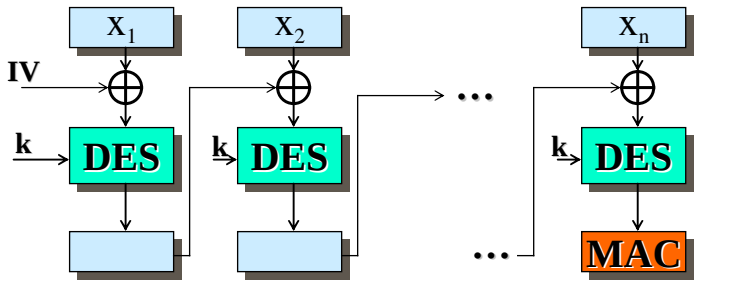
- Sforzo computazionale richiesto: $\min(2^k, 2^n)$
- Raccomandazioni: $\min(k,n) \geq 128$

MAC: Costruzioni

- MAC basati su cifrari a blocchi
  - CBC-MAC
- MAC basati su funzioni hash
  - Metodo del segreto prefisso
  - Metodo del segreto suffisso
  - HMAC

CBC-MAC

- **C**ipher **B**lock **C**haining (con IV=0)
- Uno dei più usati (FIPS PUB 113 e ANSI X9.17)
- Testo $X = X_1 X_2 \ldots X_n$ diviso in blocchi di 64 bit



Valore troncato: da 16 a 64 bit più a sinistra

MAC basati su Funzioni Hash

Vantaggi uso funzioni hash

- Sono più veloci dei cifrari a blocchi, in genere
- Sono incluse nelle funzioni di libreria, in genere
- Non ci sono restrizioni sull'esportazione dagli USA

Attenzione alla costruzione!

MAC basati su funzioni hash

- H funzione hash iterata con funzione di compressione f
- Input M, dopo il padding n blocchi $X_1X_2...X_n$

- H_0 costante iniziale
- Computazione di … $H_i = f(X_i,H_{i-1})$ …
- Valore hash $H_n = f(X_n,H_{n-1})$

computazione del valore hash

Metodo del segreto prefisso

MAC(K,M)=H(K,M)

Existential forgery attack

Per funzioni hash iterate:
Considero $M'=Mx_{n+1}$ e so calcolare
$f(x_{n+1}, H(K,M)) = H(K,Mx_{n+1}) = MAC(K,M')$

Possibile soluzione:
H(K,L,M) con L=lunghezza di M

Metodo del segreto suffisso

MAC(K,M)= H(M,K)

Existential forgery attack

Attacco compleanno:
In $O(2^{|hash(\cdot)|/2})$ passi calcola coppia
M,M': H(M) = H(M').

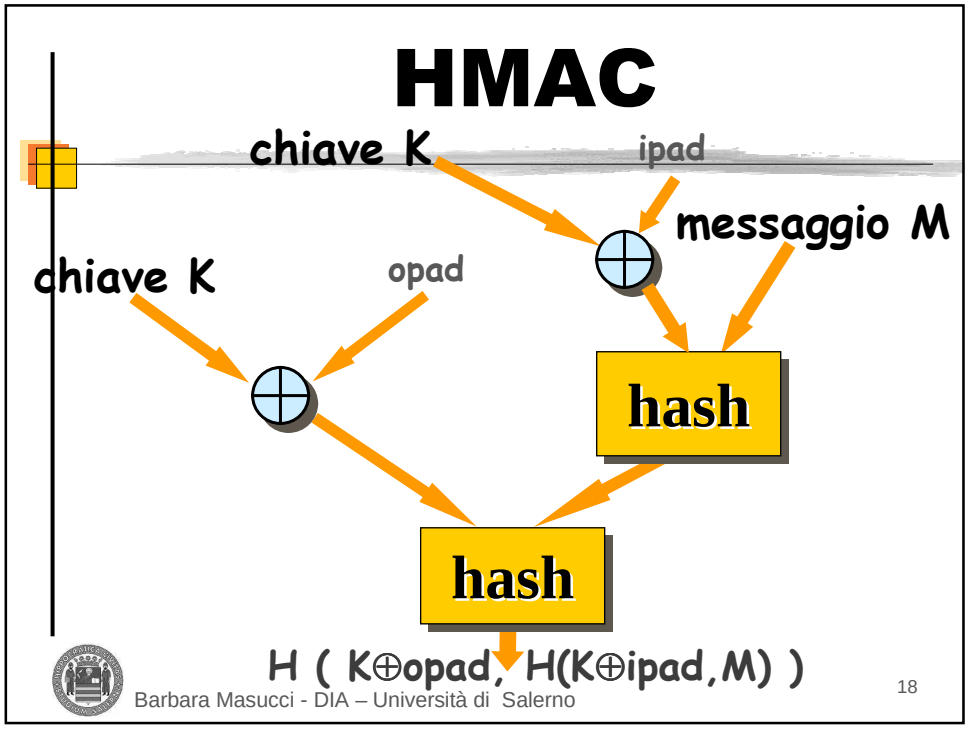
Quindi, H(M,K) = H(M',K) = MAC(M',K)

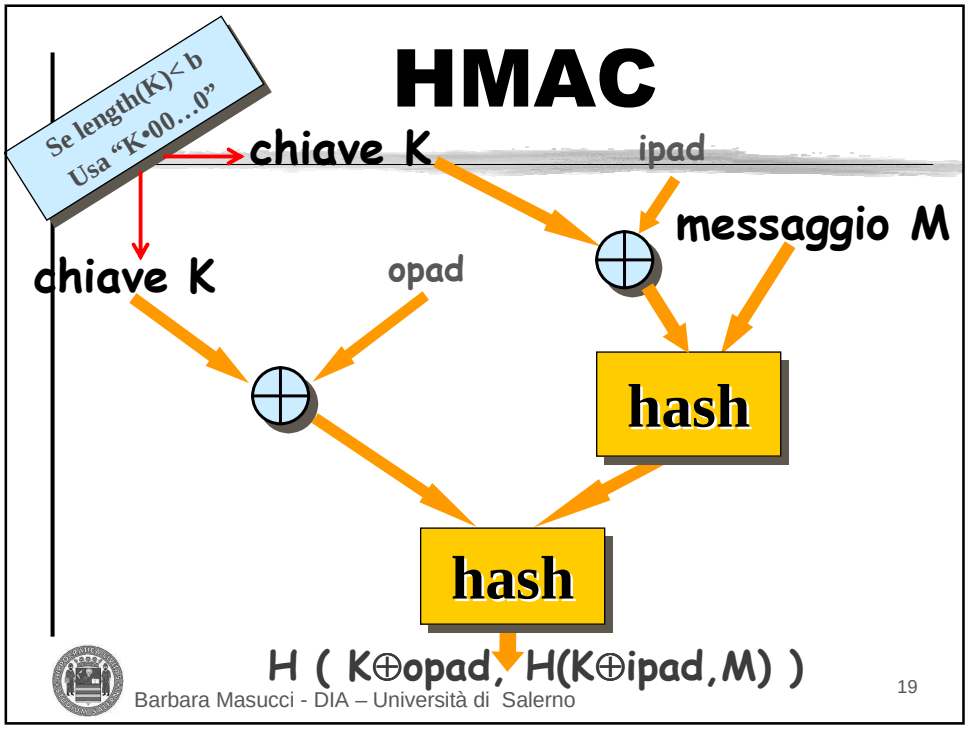
HMAC

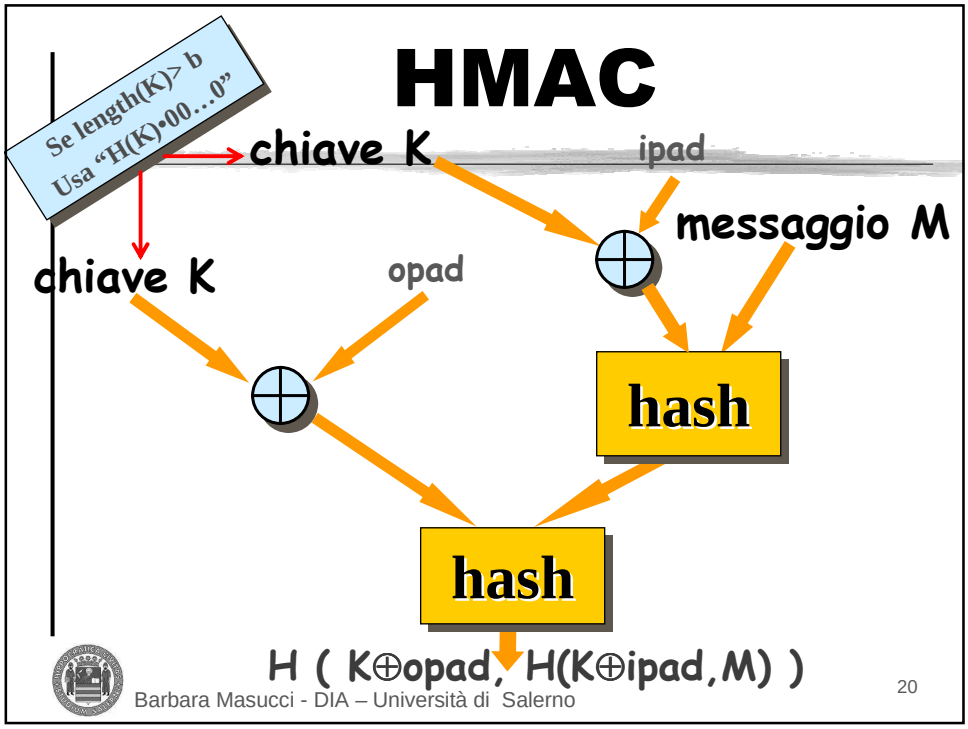
- RFC 2104, *HMAC: Keyed-Hashing for Message Authentication*, Febbraio 1997
- ANSI X9.71 *Keyed Hash Message Authentication Code, 2000*
- FIPS 198, *The Keyed-Hash Message Authentication Code (HMAC)*, pubblicato 6 marzo 2002
  - Standard effettivo dal 6 agosto 2002
  - Draft pubblicato 5 gennaio 2001, review e commenti pubblici
- Funzioni Hash usate come black-box
  - Utilizzo delle funzioni hash senza modifiche
  - Facile cambio della funzione hash (più veloci e più sicure)
- Facile utilizzo e gestione di chiavi

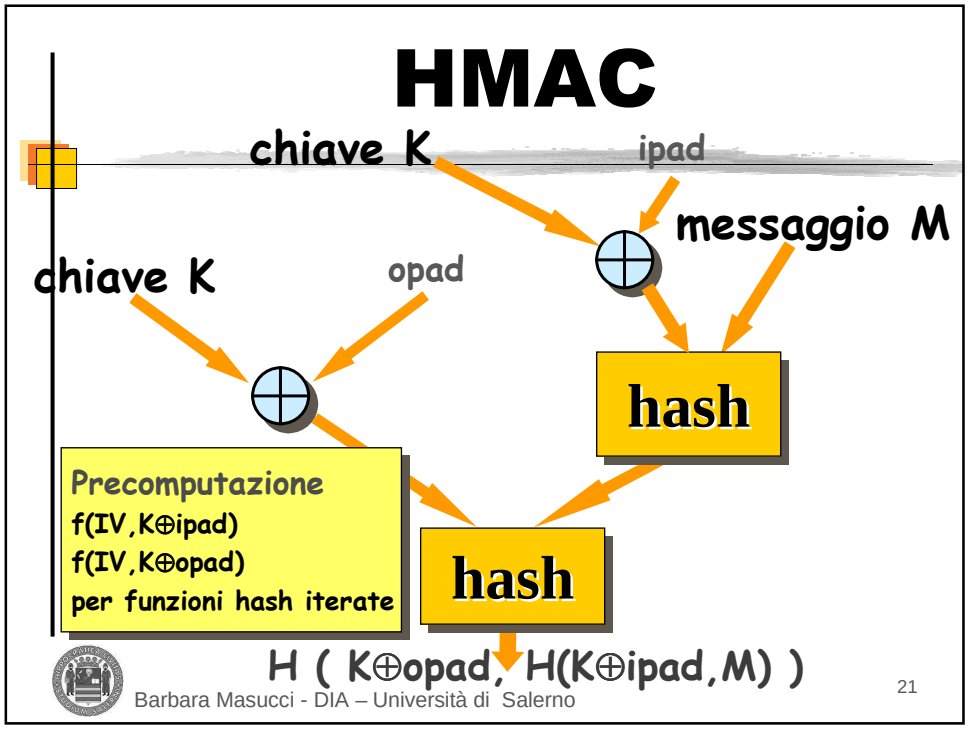
HMAC

- H: funzione hash iterata con inizialization vector IV
  - MD5, SHA-1, RIPEMD-160,…
- n: lunghezza del valore hash
  - n=128, 160,…
- M, suddiviso in L blocchi di b > n bit ciascuno
  - b=512,…
- ipad = byte 00110110 (36 in esadecimale) ripetuto b/8 volte
- opad = byte 01011100 (5C in esadecimale) ripetuto b/8 volte
- K: chiave segreta
  - Se length(K) < b, fai padding con 0…0
  - Se length(K) > b, calcola H(K) di n bit e fai padding con 0…0


HMAC
chiave K
ipad
messaggio M
chiave K
opad
hash
hash
H (K⊕opad, H(K⊕ipad,M))
Barbara Masucci - DIA – Università di Salerno
18


Se length(K)< b
Usa "K•00…0"
HMAC
chiave K
ipad
messaggio M
chiave K
opad
hash
hash
H (K⊕opad, H(K⊕ipad,M))
Barbara Masucci - DIA – Università di Salerno
19


HMAC
Se length(K)> b
Usa "H(K)•00…0"
chiave K
ipad
messaggio M
chiave K
opad
hash
hash
H (K⊕opad, H(K⊕ipad,M))
Barbara Masucci - DIA – Università di Salerno
20


HMAC
chiave K
ipad
messaggio M
chiave K
opad
hash
Precomputazione
f(IV,K⊕ipad)
f(IV,K⊕opad)
per funzioni hash iterate
hash
H (K⊕opad, H(K⊕ipad,M))
Barbara Masucci - DIA – Università di Salerno
21

Output troncato

- Diverse volte si usano solo i primi t bit dell'hash
- Esempi:
  - HMAC-SHA1-80 (solo i primi 80 dei 160 bit)
  - HMAC-MD5 (tutti i 128 bit)
- Raccomandazioni:
  - $t \geq n/2$ per una funzione hash di n bit
  - Comunque, $t \geq 80$ (RFC 2104), $t \geq 32$ (FIPS 198)

Sicurezza HMAC

- Sicurezza dipende dalle proprietà della funzione hash usata da HMAC
- Se  ha successo in un attacco ad HMAC allora:
  - Può computare l'output della funzione di compressione anche quando IV è casuale e sconosciuto all'attaccante
  - Può computare collisioni nella funzione hash anche quando IV è casuale e sconosciuto all'attaccante

Attacchi ad HMAC

- Miglior attacco conosciuto [1995,1996] basato sul paradosso del compleanno
  - Occorrono $2^{|hash(\cdot)|/2}$ coppie $(M, HMAC_K(M))$
- Esempio:

2^{64} coppie

$(M, HMAC\text{-}MD5_K(M))$

Stessa K, diversi M

chiave K

Bibliografia

- Cryptography and Network Security by W. Stallings (2003)
  - cap. 11 (MAC) e 12 (HMAC)
- Tesina di Sicurezza su reti
  - Message Authentication Code