Corso di Laurea in Ingegneria Informatica

Corso di Reti di Calcolatori I

Roberto Canonico (roberto.canonico@unina.it)
Giorgio Ventre (giorgio.ventre@unina.it)

Sicurezza nella comunicazione in rete: integrità dei messaggi, firma elettronica, protocolli di autenticazione

I lucidi presentati al corso sono uno strumento didattico
che NON sostituisce i testi indicati nel programma del corso
I lucidi sono adattati dagli originali di J. Kurose e K. Ross e fanno riferimento al testo
Reti di calcolatori e Internet - Un approccio top-down (4a ed.)

Nota di copyright per le slide COMICS

Nota di Copyright

Integrità dei messaggi

Bob riceve un messaggio da Alice, e desidera assicurarsi che:

- il messaggio veramente è stato prodotto da Alice (e non da qualcuno che finge di essere Alice)
- il messaggio non è stato alterato da alcuno dopo che Alice lo ha inviato

Soluzione: funzioni hash crittografiche

- Una funzione hash prende un testo in ingresso m, e produce una stringa di lunghezza prefissata H(m)
  - Es. la Internet checksum usata da TCP ed UDP
- La funzione H è tale che è computazionalmente impossibile trovare 2 messaggi diversi x ed y tali che H(x) = H(y)
  - equivalentemente: noto m = H(x), non è possibile ricavare x.
  - N.B.: la Internet checksum *non soddisfa* questo requisito

Perchè la Internet checksum non può essere una funzione hash

La Internet checksum gode di alcune delle proprietà di una funzione hash:

- produce un digest di lunghezza fissa (16-bit) di un messaggio
- Uno stesso valore di checksum può essere generato da molti messaggi diversi

Ma, dato un messaggio con un dato valore di checksum, è facile trovare un altro messaggio con lo stesso valore della checksum:

messaggio	ASCII format
I O U 1	49 4F 55 31
0 0 . 9	30 30 2E 39
9 B O B	39 42 4F 42
	B2 C1 D2 AC

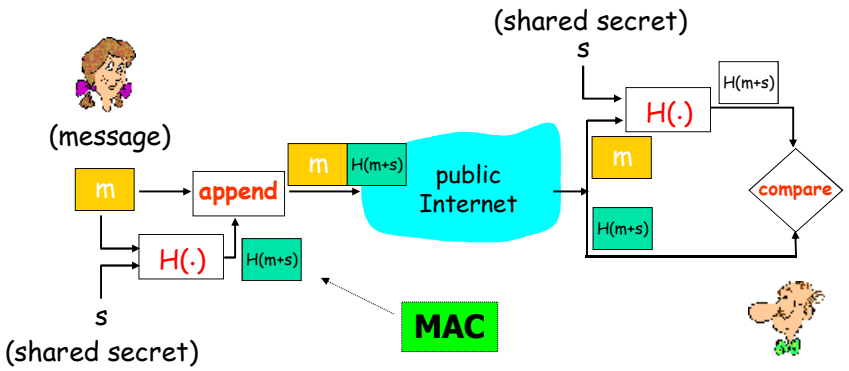
messaggio	ASCII format
I O U 9	49 4F 55 39
0 0 . 1	30 30 2E 31
9 B O B	39 42 4F 42
	B2 C1 D2 AC

Messaggi diversi ma stessa checksum!

Funzioni hash usate in pratica

- MD5 (RFC 1321)
  - ideato da Ron Rivest
  - calcola un valore hash da 128 bit in 4 passi
  - recentemente (2005) sono state trovate delle tecniche di attacco per MD5 che sono in grado di trovare due messaggi con la stesso valore di hash (*collisioni*)
- SHA-1
  - standard statunitense
  - calcola un valore hash da 160 bit

Message Authentication Code (MAC)



Consente di controllare l'integrità di un messaggio attraverso l'uso di una funzione hash ed una chiave segreta s (*chiave di autenticazione*) nota ad entrambi (*shared secret*)

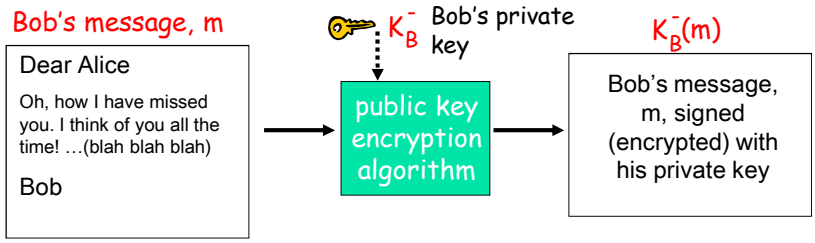
Firma digitale

- La firma digitale è una tecnica crittografica che ha gli stessi scopi della firma fatta a mano sui documenti cartacei
- Il mittente di un messaggio (Bob) appone la sua firma digitale al fine di stabilire che egli è il creatore/autore del testo in esso contenuto
- Come la firma tradizionale, anche quella digitale deve essere ***verificabile*** e ***non falsificabile***: il destinatario del messaggio (Alice) deve poter provare ad un terzo che è stato Bob, e nessun altro (inclusa la stessa Alice) ad apporre la firma al documento

Firma digitale

Una semplice tecnica di firma digitale:

- Bob "firma" il messaggio m crittografandolo con la sua chiave privata K_B^-, creando un messaggio firmato $K_B^-(m)$



Firma digitale (continua)

- Alice riceve il messaggio m, con la firma digitale $K_B^-(m)$
- Alice verifica che m sia stato effettivamente firmato da Bob decifrando con la chiave pubblica di Bob K_B^+ il testo cifrato ricevuto $K_B^-(m)$ e verifica che sia $K_B^+(K_B^-(m)) = m$
- se $K_B^+(K_B^-(m)) = m$, chiunque abbia firmato m deve possedere la chiave privata di Bob

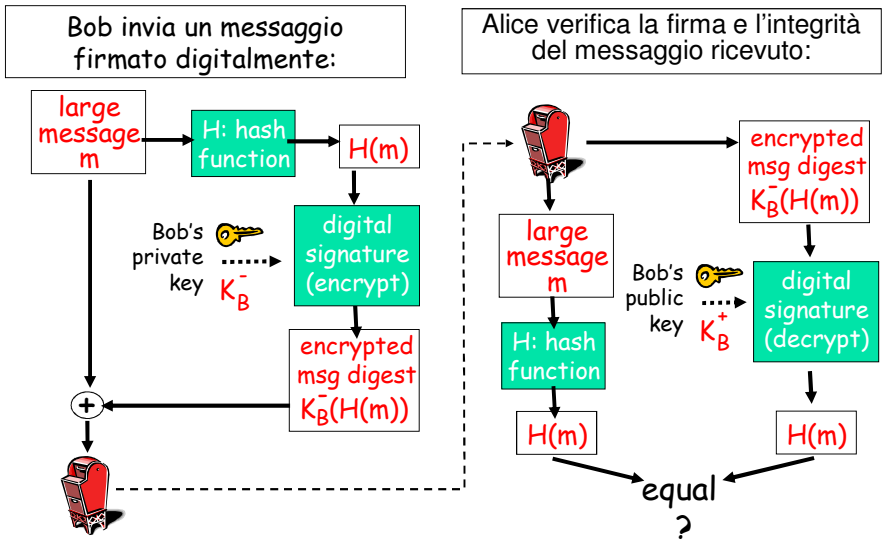
Alice così verifica che:

- Bob ha firmato m
- Nessun altro ha firmato m
- Bob ha firmato m e non m'

non-ripudio:

- ✓ Alice può portare il documento m, e la relativa firma $K_B^-(m)$ da un giudice e provare che Bob ha firmato m

Firma digitale = firma del digest



Certificazione della chiave pubblica

Problema di distribuzione affidabile delle chiavi pubbliche:

• Quando Alice ottiene la chiave pubblica di Bob (da un sito web, una email, un supporto di registrazione, ecc…), come fa ad essere sicura che sia realmente la chiave pubblica di Bob e non di un altro (es. Trudy) ?

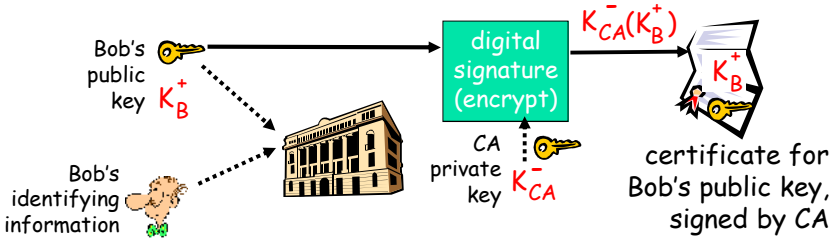
Soluzione:

• trusted certification authority (CA)

• Una CA è un ente (*trusted third party*), pubblico o privato, abilitato a rilasciare un certificato digitale tramite procedura di certificazione che segue standard internazionali e conforme alla normativa europea e nazionale in materia

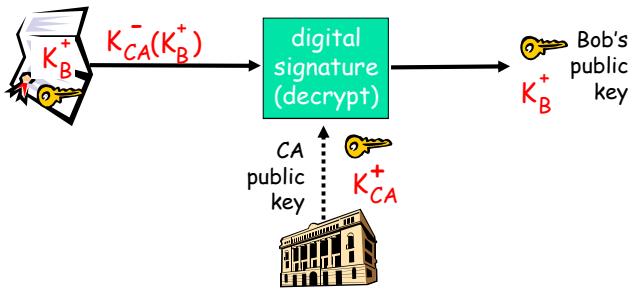
Certification Authorities

- Una CA associa (in modo sicuro) una chiave pubblica ad una particolare entità E
- E registra la sua chiave pubblica presso la CA
  - A tal fine, E fornisce la sua "prova di identita" alla CA
  - CA crea un "certificato" che collega E alla sua chiave pubblica
  - Un certificato contiene la chiave pubblica di E firmata digitalmente dalla CA: CA afferma che "Questa è la chiave pubblica di E"



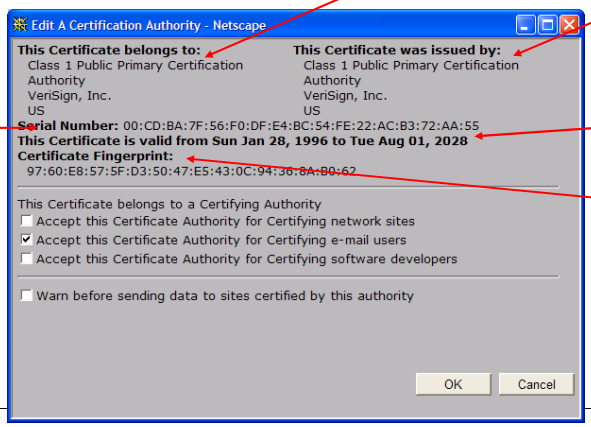
Certification Authorities

- quando Alice desidera la chiave pubblica di Bob:
  - ottiene il certificato di Bob (da Bob o da altri)
  - applica la chiave pubblica della CA al certificato di Bob, ed ottiene la chiave pubblica di Bob



Un certificato contiene:

- Un numero di serie (unico tra quelli emessi dalla CA)
- Informazioni riguardo al possessore del certificato

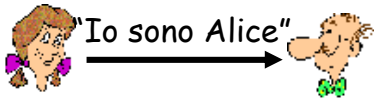


- info su chi ha emesso il certificato
- date di validità
- firma digitale di chi ha emesso il certificato

Autenticazione: protocollo ap1.0

obiettivo: Bob desidera che Alice provi la sua identità

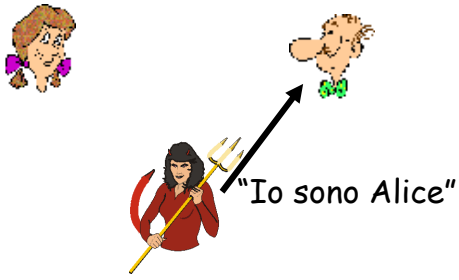
Protocollo ap1.0: Alice dice "Io sono Alice"



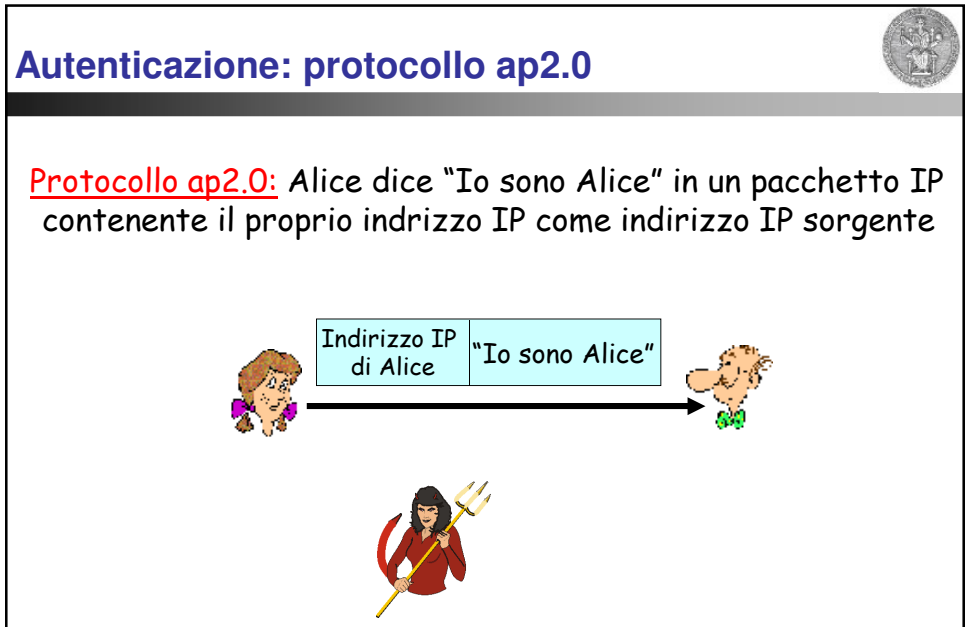
Protocollo ap1.0: fallimento

Fallimento: in una rete Bob non può "vedere" Alice, perciò Trudy può facilmente fingere di essere Alice

Protocollo ap1.0: Alice dice "Io sono Alice"



Autenticazione: protocollo ap2.0



Protocollo ap2.0: Alice dice "Io sono Alice" in un pacchetto IP contenente il proprio indrizzo IP come indirizzo IP sorgente

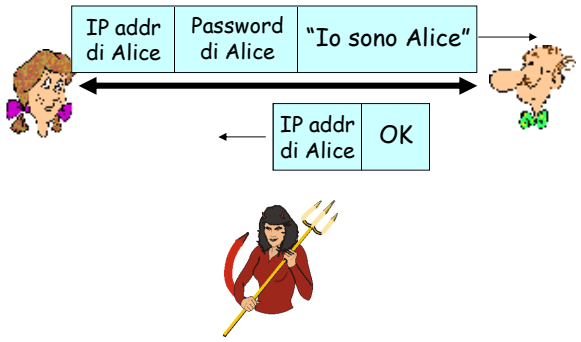
Protocollo ap2.0: fallimento

Fallimento: Trudy può creare un pacchetto IP facendo lo "spoofing" dell'indirizzo IP di Alice

Protocollo ap2.0: Alice dice "Io sono Alice" in un pacchetto IP contenente il proprio indrizzo IP come indirizzo IP sorgente

Autenticazione: protocollo ap3.0

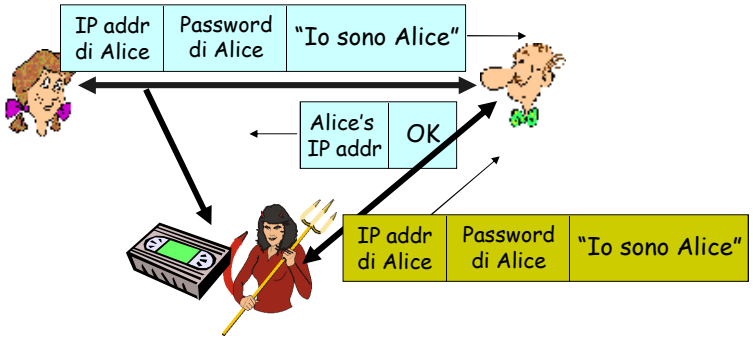
Protocollo ap3.0: Alice dice "Io sono Alice" e manda la sua password segreta per provare la sua affermazione



Protocollo ap3.0: fallimento

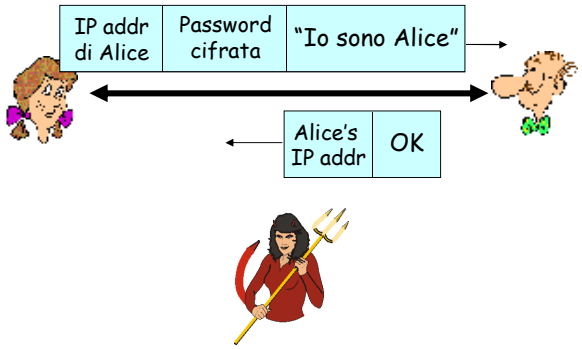
Protocollo ap3.0: Alice dice "Io sono Alice" e manda la sua password segreta per provare la sua affermazione

Attacco playback: Trudy registra il pacchetto di autenticazione di Alice e successivamente lo rimanda a Bob



Autenticazione: protocollo ap3.1

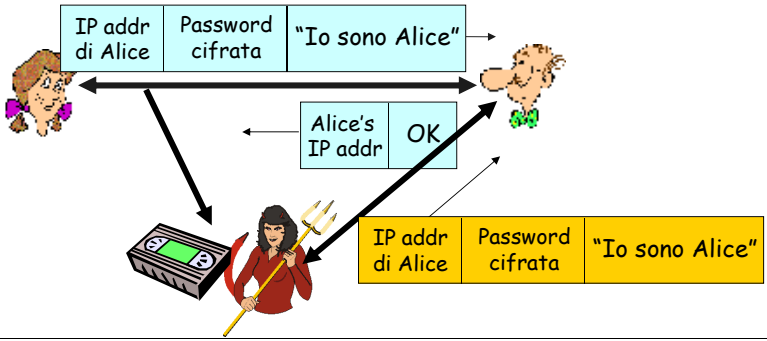
Protocollo ap3.1: Alice dice "Io sono Alice" e manda la sua password segreta cifrata per provare la sua affermazione



Protocollo ap3.1: fallimento

Protocollo ap3.1: Alice dice "Io sono Alice" e manda la sua password segreta cifrata per provare la sua affermazione

Attacco playback: Trudy registra il pacchetto di autenticazione di Alice e successivamente lo rimanda a Bob (funziona ancora!)

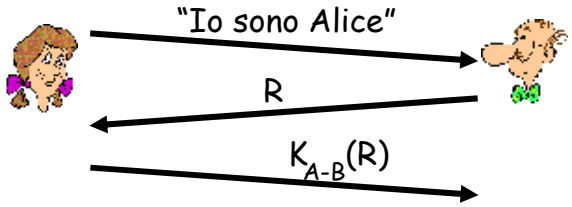


Autenticazione: protocollo ap4.0

Obiettivo: impedire l'attacco playback

Soluzione: Nonce = numero R usato *una sola volta*

ap4.0: Bob invia ad Alice un nonce, R
Alice deve restituire R, cifrato con la sua chiave segreta



Problema da risolvere: quello della crittografia a chiave simmetrica, cioè lo scambio delle chiavi

Autenticazione: protocollo ap5.0

ap5.0: Bob invia ad Alice un nonce, R
Alice deve restituire R, cifrato con la sua chiave privata

Bob calcola

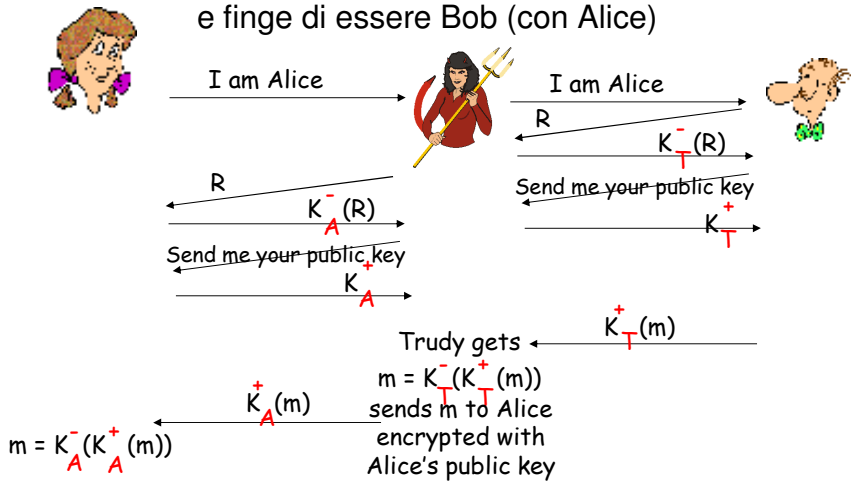
$K_A^+(K_A^-(R))$

e sa che solo Alice possiede la chiave privata tale che

$K_A^+(K_A^-(R)) = R$

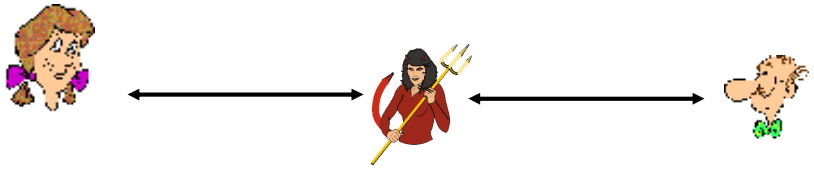
Protocollo ap5.0: difetto di sicurezza

Attacco "man in the middle":
Trudy finge di essere Alice (con Bob)
e finge di essere Bob (con Alice)



Protocollo ap5.0: difetto di sicurezza (2)

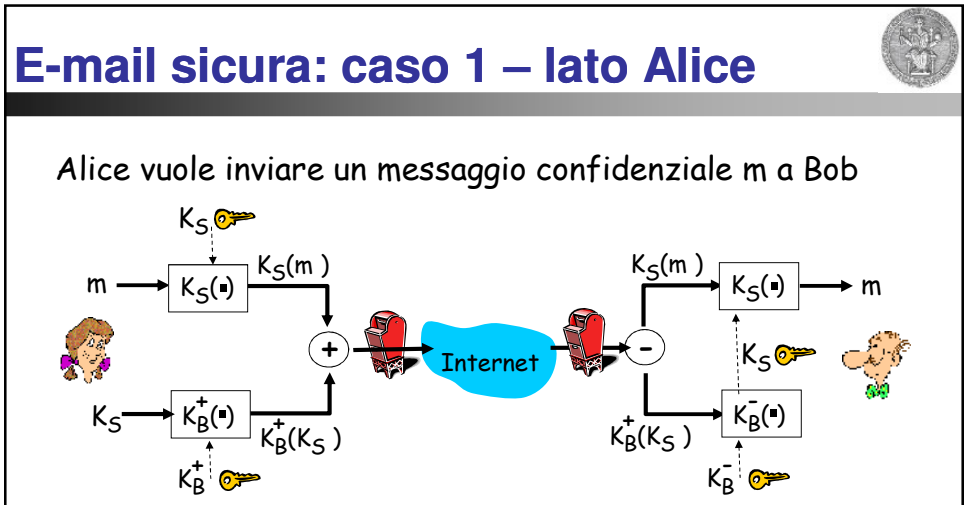
Attacco "man in the middle":
Trudy finge di essere Alice (con Bob)
e finge di essere Bob (con Alice)



Difficile da rilevare:
- Bob riceve tutto quello che Alice gli manda, e viceversa
- il problema è che anche Trudy riceve tutti i messaggi!

Il difetto di sicurezza di ap5.0 è legato alla distribuzione delle chiavi pubbliche

E-mail sicura: caso 1 – lato Alice

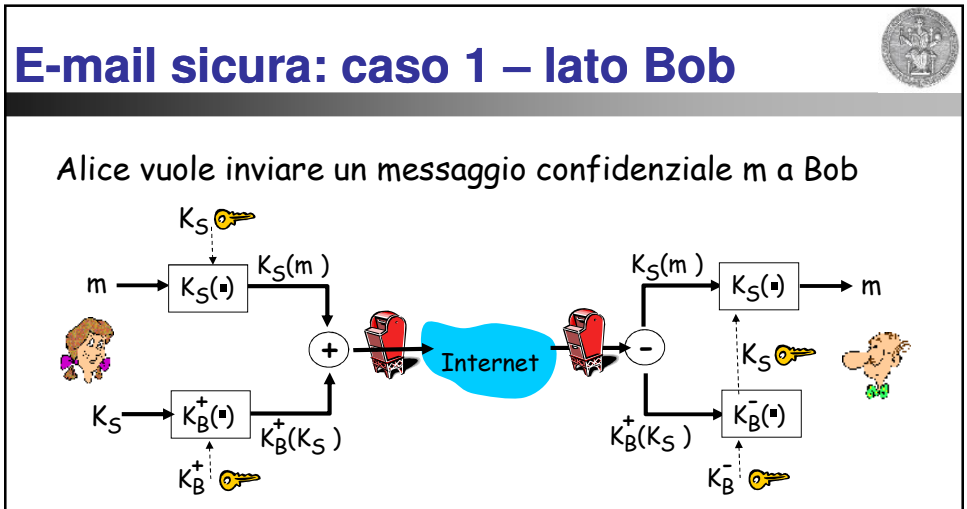


Alice vuole inviare un messaggio confidenziale m a Bob

Alice:

- genera una chiave *simmetrica* privata K_S
- cifra il messaggio m con K_S
- cifra K_S con la chiave pubblica di Bob
- invia sia $K_S(m)$ che $K_B(K_S)$ a Bob

E-mail sicura: caso 1 – lato Bob



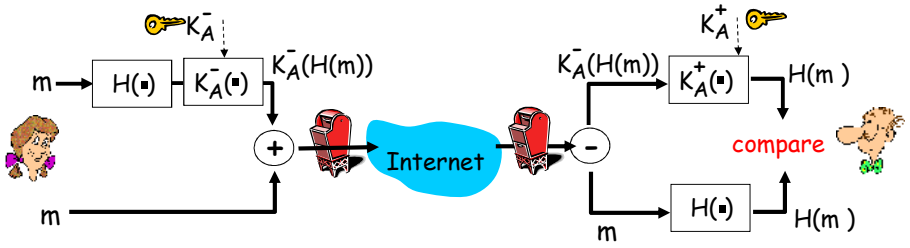
Alice vuole inviare un messaggio confidenziale m a Bob

Bob:

- usa la sua chiave privata per decifrare e recuperare K_S
- usas K_S per decifrare $K_S(m)$ per recuperare m

E-mail sicura: caso 2

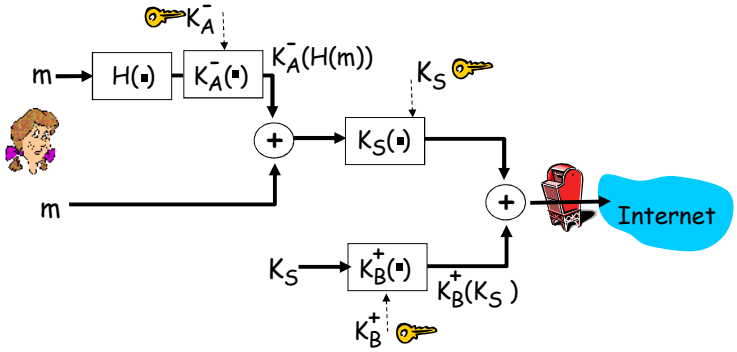
Alice vuole che Bob possa essere sicuro della identità del mittente e della integrità del messaggio ricevuto



- Alice appone la sua firma digitale al messaggio
- invia il messaggio (in chiaro) e la firma digitale

E-mail sicura: caso 3

Alice vuole inviare un messaggio confidenziale a Bob e vuole che Bob possa essere sicuro della identità del mittente e della integrità del messaggio ricevuto



Alice usa 3 chiavi: la sua chiave privata, la chiave pubblica di Bob, la chiave simmetrica appena generata Ks