Programmazione in Rete

a.a. 2005/2006

http://www.di.uniba.it/~lisi/courses/prog-rete/prog-rete0506.htm

dott.ssa Francesca A. Lisi
lisi@di.uniba.it

Orario di ricevimento: mercoledì ore 10-12

dib

Sommario della lezione di oggi: La sicurezza nelle reti (2/3)

Dai fondamenti ...

- che cosa è la sicurezza?
- crittografia
- autenticazione
- integrità e non ripudiabilità del messaggio
- distribuzione delle chiavi e certificazione

... in pratica

- a livello di applicazione: e-mail sicura
- a livello di trasporto: commercio Internet, SSL, SET
- a livello di rete: sicurezza IP

Autenticazione: primo tentativo

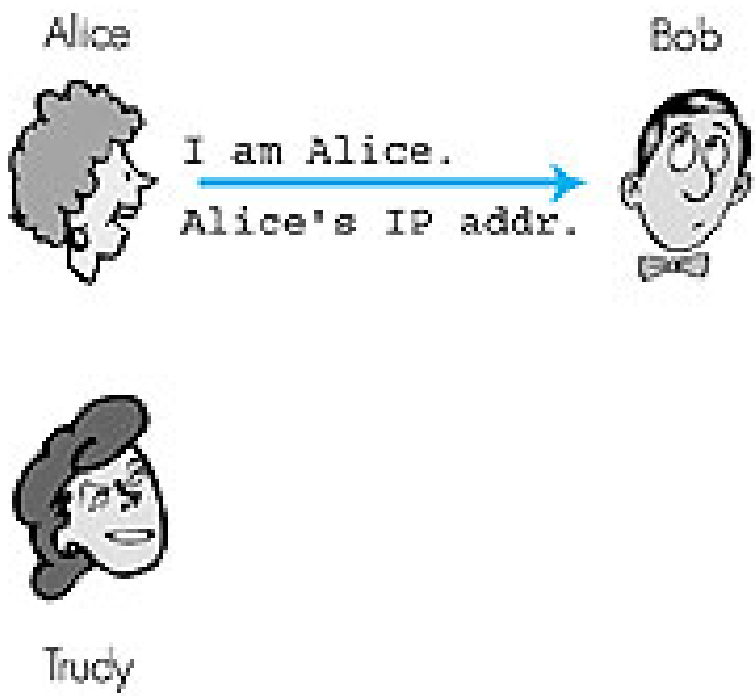
Obiettivo: Bob vuole che Alice gli "provi" la sua identità

Protocollo ap1.0: Alice dice "I am Alice"

Scenario di fallimento??

Autenticazione: secondo tentativo

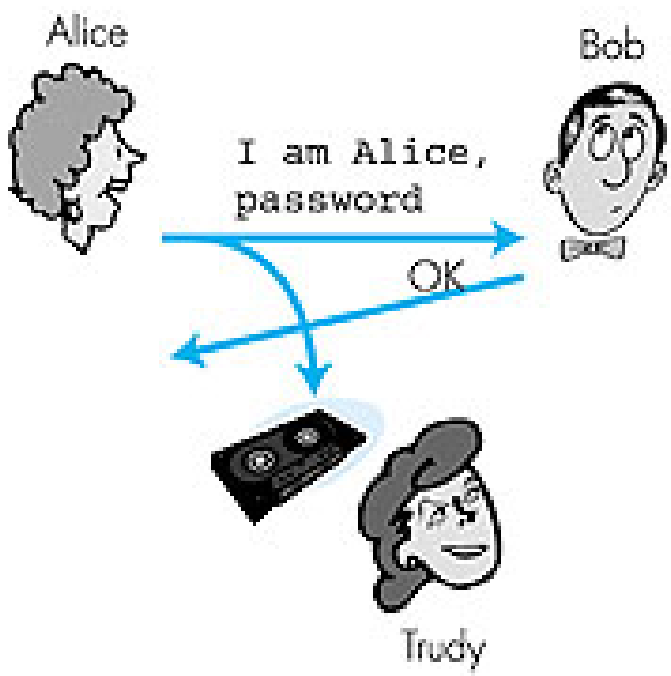
Protocollo ap2.0: Alice dice "I am Alice" ed invia il proprio indirizzo IP insieme per "provarlo".



Scenario di fallimento??

Autenticazione: terzo tentativo

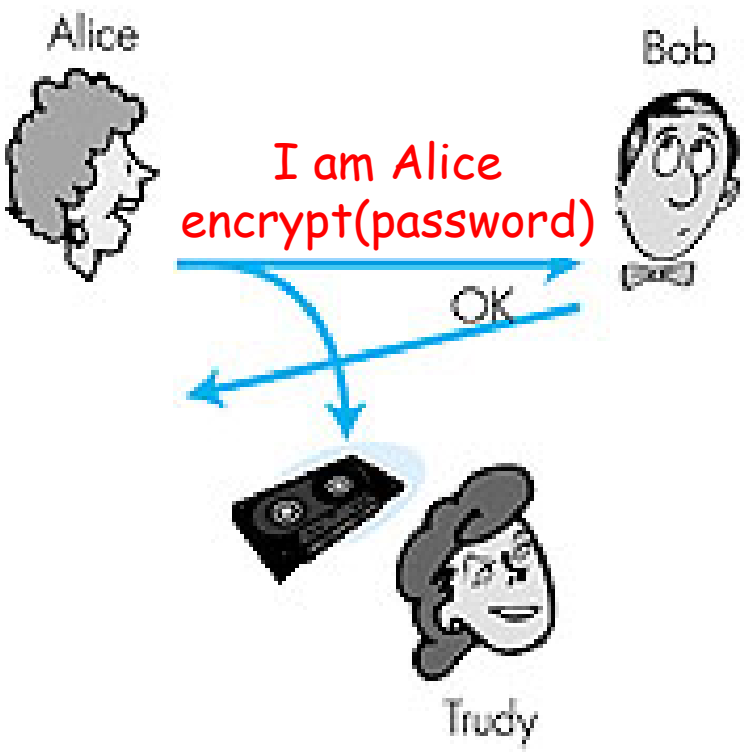
Protocollo ap3.0: Alice dice "I am Alice" ed invia la sua pwd segreta per "provarlo".



Scenario di fallimento?

Autenticazione: una variante del terzo tentativo

Protocollo ap3.1: Alice dice "I am Alice" ed invia la sua password segreta *cifrata* per "provarlo".



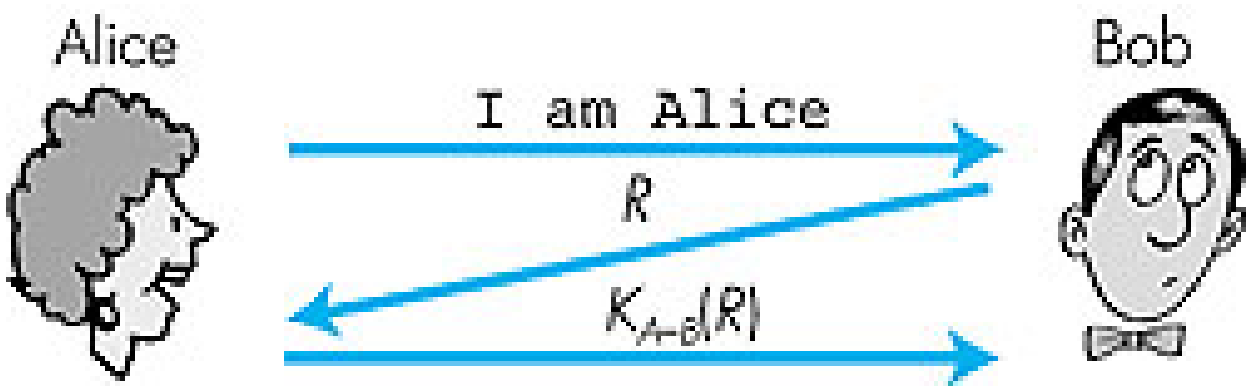
Scenario di fallimento?

Autenticazione: quarto tentativo

Obiettivo: evitare attacchi di playback

Nonce: numero (R) usato solo una volta in un ciclo di vita

ap4.0: per verificare se Alice è "viva", Bob invia ad Alice un nonce, R. Alice deve restitutire R, decodificato con una chiave segreta condivisa



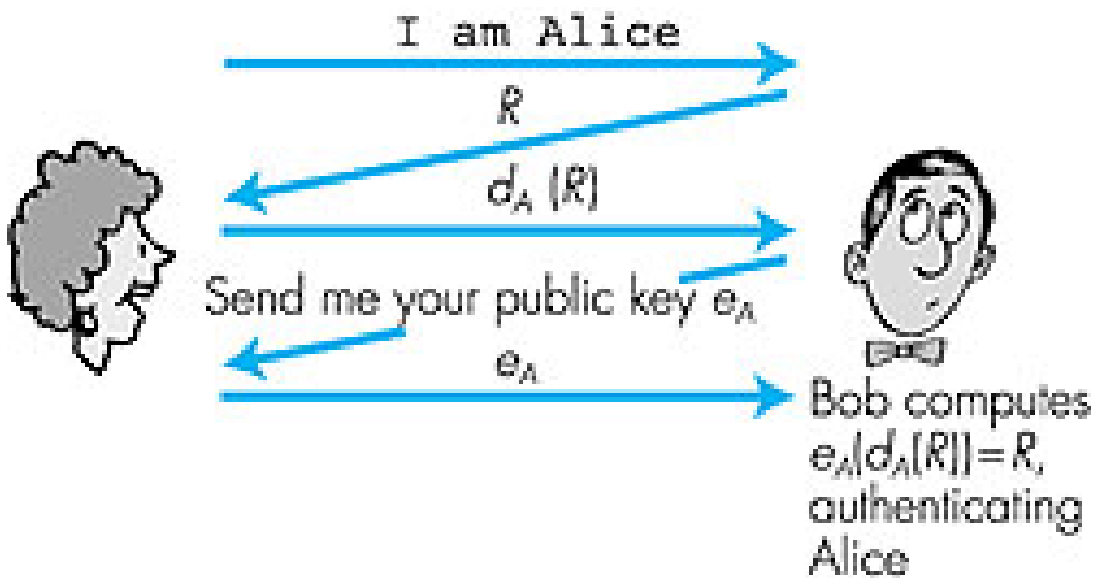
Fallimenti, effetti collaterali?

Autenticazione: quinto ed ultimo tentativo

ap4.0 richiede una chiave simmetrica

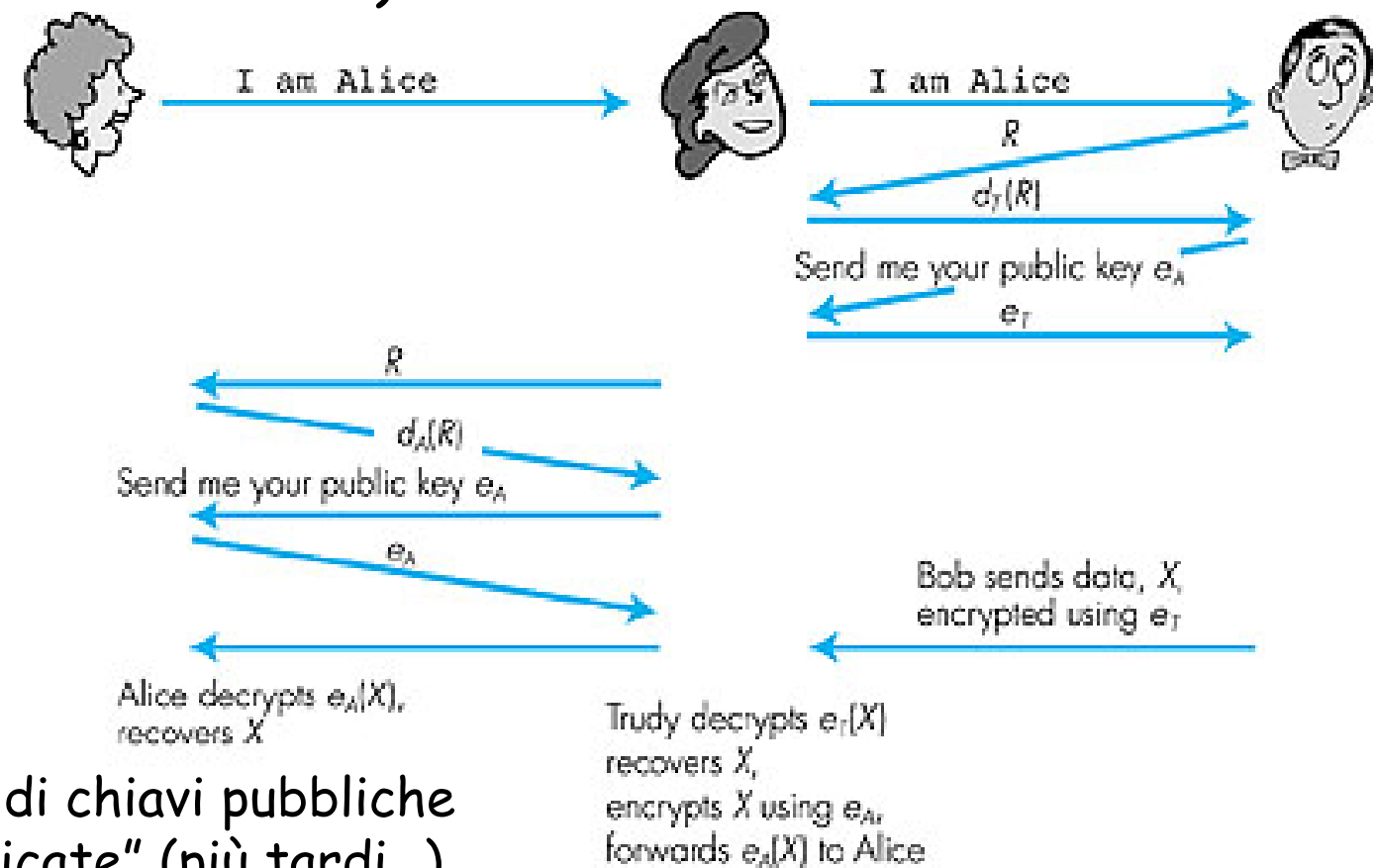
- problema: come fanno Bob e Alice ad accordarsi sulla chiave?

ap5.0: usare i nonce con crittografia a chiave pubblica



ap5.0: buco nella sicurezza

Attacco *man (woman)-in-the-middle*: Trudy si pone come Alice (nei confronti di Bob) e come Bob (nei confronti di Bob)



Bisogno di chiavi pubbliche "certificate" (più tardi…)

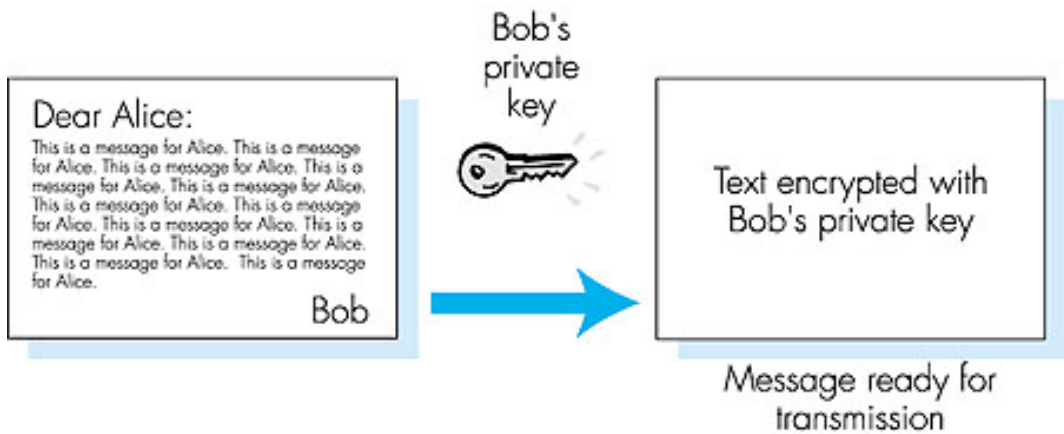
Integrità: firme digitali

Tecnica crittografica analoga alle firme scritte a mano.

- Non ripudiabile: Il mittente (Bob) firma elettronicamente il documento, stabilendo che lui è il possessore/creatore del documento.
- Verificabile, non falsificabile: il ricevente (Alice) può verificare che Bob, e nessun altro, ha firmato il documento.

Semplice firma digitale per il messaggio m:

- Bob codifica m con la sua chiave privata d_B, creando il messaggio firmato, $d_B(m)$.
- Bob invia m e $d_B(m)$ ad Alice.



Integrità: firme digitali (cont.)

- Supponiamo che Alice riceve il msg *m* e la firma digitale $d_B(m)$
- Alice verifica che *m* sia firmato da Bob applicando la chiave pubblica di Bob e_B a $d_B(m)$ *quindi* controlla se $e_B(d_B(m)) = m$.
- Se $e_B(d_B(m)) = m$, chiunque abbia firmato *m* deve aver usato la chiave privata di Bob.

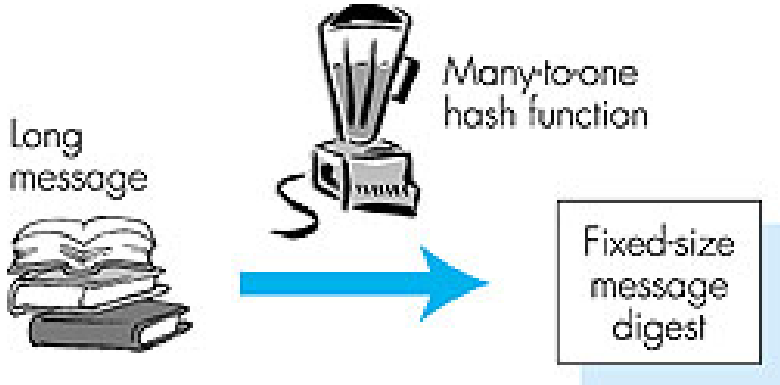
Alice pertanto verifica che:

- Bob ha firmato *m*.
- Nessun altro ha firmato *m*.
- Bob ha firmato *m* ma non *m'*.

Non ripudio:

- Alice può prendere *m*, e la firma $d_B(m)$ per corteggiare e provare che Bob ha firmato *m*.

Integrità: digest del messaggio



E' computazionalmente costoso codificare con chiave pubblica i messaggi lunghi!

Obiettivo: lunghezza fissa, facile da calcolare una firma digitale, "fingerprint"

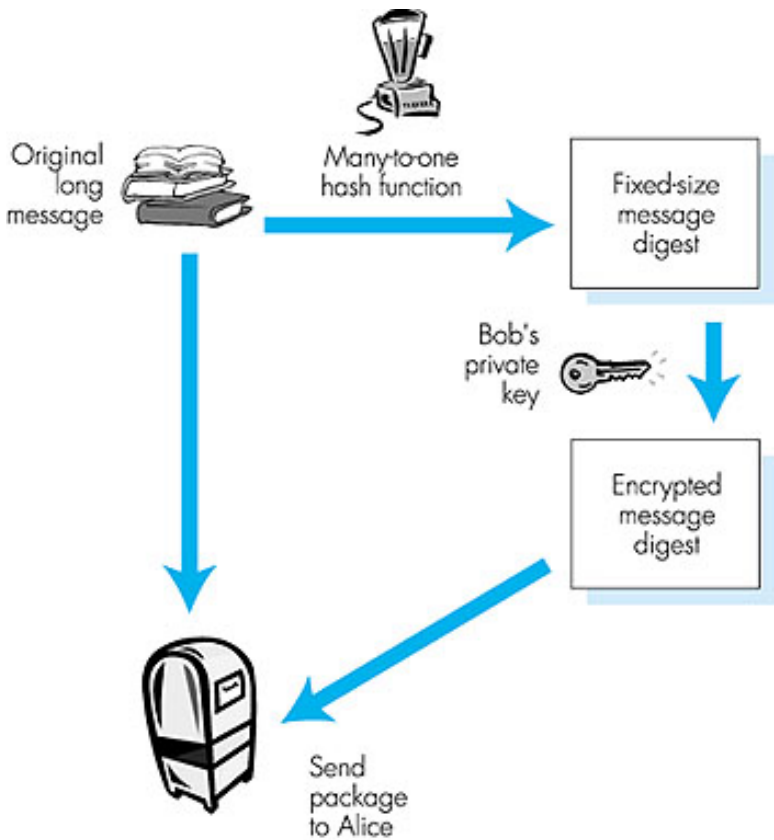
- applicare funzione hash H ad *m* per ottenere digest di msg a lungh. fissa, *H(m)*.

Proprietà delle funzioni Hash:

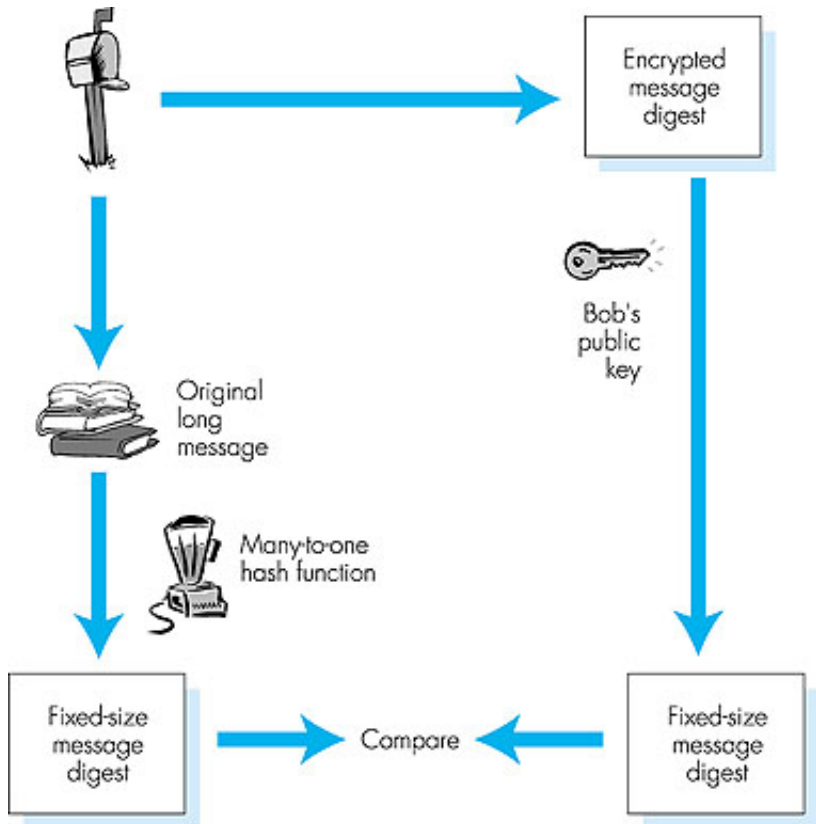
- Molti-ad-1
- Produce digest di msg a lunghezza fissa (fingerprint)
- Dato un digest di msg x, è computazionalmente inpossibile
  - trovare *m* t.c. *x = H(m)*
  - trovare due qualsiasi messaggi *m* ed *m'* t.c. *H(m) = H(m')*.

Integrità:
firma digitale = digest firmato del msg

Bob invia elettronicamente un messaggio firmato:



Alice verifica la firma e l'integrità del messaggio firmato elettronicamente:



Integrità: Algoritmi per la funzione hash

- Il checksum di Internet produrrebbe un digest di messaggio inaffidabile.
  - Troppo facile trovare due messaggi con stesso checksum!
- Funzione hash MD5 ampiamente utilizzata.
  - Calcola digest a 128-bit in 4 passi.
  - Presa una stringa arbitraria x a 128 bit, appare difficile costruire msg m il cui hash MD5 è uguale a x.
- SHA-1 è anche usata.
  - US standard
  - 160-bit message digest

Distribuzione delle chiavi: intermediari di fiducia

Problema 1):

- Come fanno due entità a stabilire una chiave segreta condivisa sulla rete?

Soluzione:

- *key distribution center* (KDC) di fiducia che agisce da intermediario fra le entità

Problema 2):

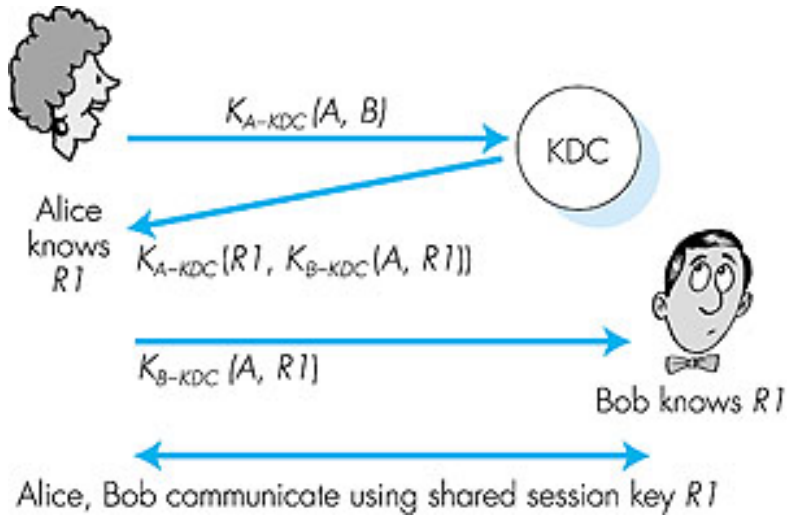
- Quando Alice ottiene la chiave pubblica di Bob (tramite un sito web, o e-mail o floppy), come fa a sapere che è proprio la chiave pubblica di Bob e non di Trudy?

Soluzione:

- *certification authority* (CA) di fiducia

Key Distribution Center (KDC)

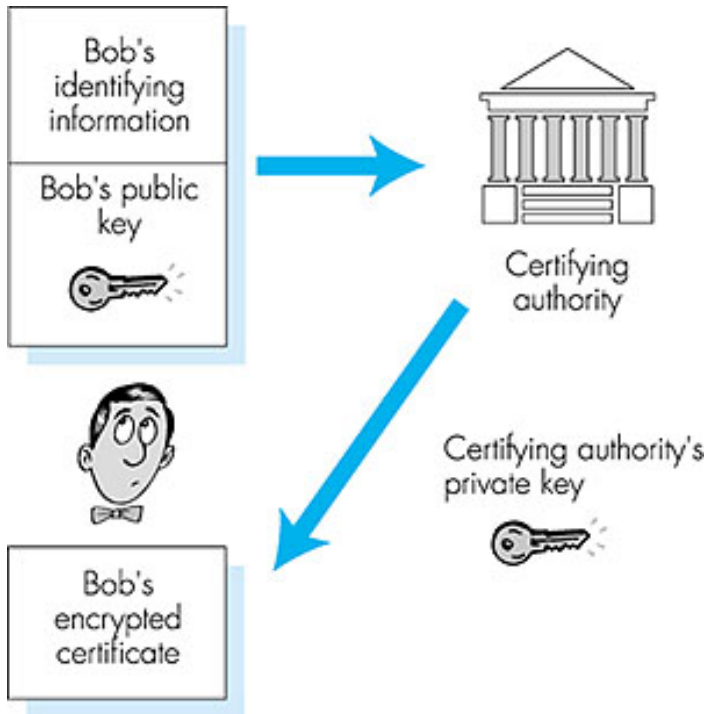
- Alice e Bob hanno bisogno di condividere una chiave simmetrica.
- il server KDC condivide una diversa chiave segreta per ogni utente registrato.
- Alice e Bob conoscono le proprie chiavi simmetriche, $K_{A\text{-}KDC}$ e $K_{B\text{-}KDC}$, per comunicare con il KDC.



- Alice comunica con il KDC, ottiene la chiave di sessione R1, e $K_{B\text{-}KDC}(A,R1)$
- Alice invia $K_{B\text{-}KDC}(A,R1)$ a Bob, Bob estrae R1
- Alice e Bob ora condividono la chiave simmetrica R1.

Certification Authorities

- I CA legano le chiavi pubbliche ad una particulare entità.
- Un'entità (persona, router, etc.) può registrare la propria chiave pubblica con un CA.
  - Entità fornisce "prova di identità" al CA.
  - CA crea certificato che lega entità ad una chiave pubblica.
  - Certificato elettronicamente firmato dal CA.



Qundo Alice vuole sapere la chiave pubblica di Bob:

- ottiene il certificato di Bob (da Bob o da qualche altra parte).
- Applica la chiave pubblica del CA al certificato di Bob ed ottiene la chiave pubblica di Bob

Sommario della prossima lezione: La sicurezza nelle reti (3/3)

Dai fondamenti ...

- che cosa è la sicurezza?
- crittografia
- autenticazione
- integrità e non ripudiabilità del messaggio
- distribuzione delle chiavi e certificazione

... in pratica

- a livello di applicazione: e-mail sicura
- a livello di trasporto: commercio Internet, SSL, SET
- a livello di rete: sicurezza IP