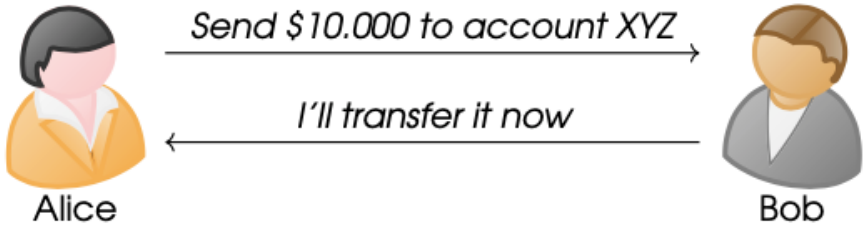
Protocollo Crittografici

Indice

- Motivazioni e introduzione ai protocolli crittografici
- Definizioni di base
- Notazioni ed esempi
- Assunzioni e obiettivi dell'analisi di protocolli
- Tipi di attacchi
- Analisi dei protocolli NSPK e NSSK.

Protocolli crittografici: motivazioni

- Esempio: rendere sicura una applicazione di e-banking:



- Come fa Bob a sapere che il messaggio arriva da Alice?
- Come fa Bob a sapere che Alice ha dichiarato proprio quello che gli arriva nel messaggio?

Protocolli crittografici: motivazioni /2

- Altri esempi possono essere relativi a rendere sicuri:
  - Una rete di sensori
  - Uno schema di micropagamento per il parcheggio
  - Un sistema di controllo degli accessi per un complesso sciistico

Come costruire algoritmi distribuiti sicuri

per fornire le opportune garanzie?

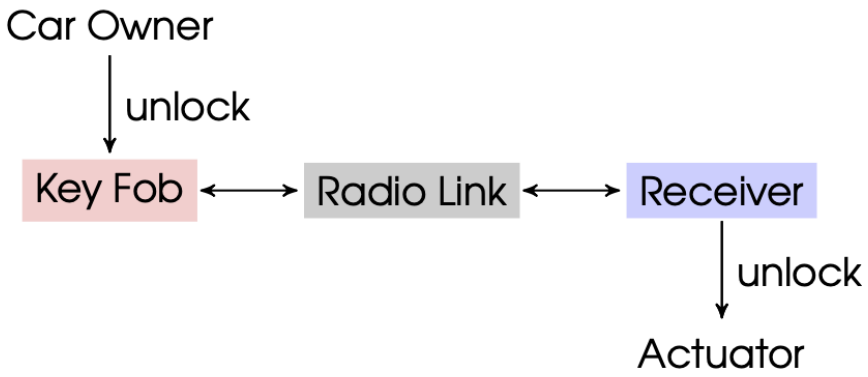
Definizioni

- Un **protocollo** è formato da un insieme di regole che regolano lo scambio di messaggi tra due o più parti. In sostanza, un protocollo è un algoritmo distribuito con particolare enfasi sulla comunicazione.
- Un **protocollo di sicurezza** (o crittografico) usa meccanismi di cifratura per garantire determinate proprietà di sicurezza come ad esempio l'autenticazione del messaggio o dell'entità, l'integrità, la non ripudiablità, ecc.

Messaggi

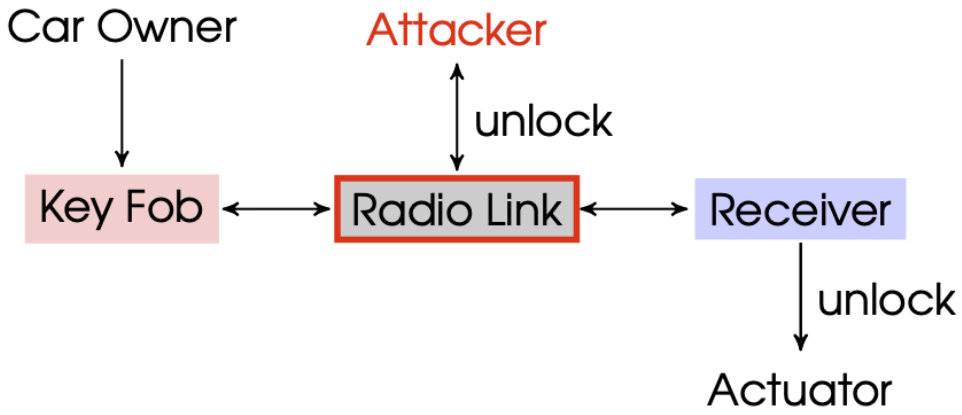
- I costruttori di un messaggio sono:
  - Nomi: A, B oppure Alice, Bob, …
  - Chiavi: K (chiave pubblica) e l'inversa K^{-1} (chiave privata)
  - Cifratura: $\{M\}_K$; esempio: cifratura con la chiave pubblica di A: $\{M\}_{KA}$
  - Firma: $\{M\}_K^{-1}$; esempio: firma con la chiave privata di A $\{M\}_K^{-1}$
  - Chiavi simmetriche: $\{M\}_{KAB}$
  - Nonce: N_A
  - Timestamp: T
  - Concatenazione di messaggi: $\{M_1, M_2\}$
  - Esempio: $\{A, T_A, K_{AB}\}_{KB}$

Esempio: sistema a chiave wireless



- Obiettivo di sicurezza: il ricevente manda un comando di sblocco all'Actuator solo se il Car Owner ha precedentemente schiacciato il bottone di sblocco sul Key Fob.

Sistema a chiave wireless /2



- Problema: il Radio Link potrebbe essere controllato da un attaccante e mandare il comando di sblocco. Come costruire un protocollo che sia robusto verso un Radio Link attaccabile?

Sistema a chiave wireless /3

- Possiamo assumere che ci sia un segreto condiviso (SN) tra Key Fob (KF) e Receiver (R).
- In questo caso KF manda [SN, unlock] a R:

$$1.\ KF \rightarrow R : unlock, SN$$

Sistema a chiave wireless /3

- Possiamo assumere che ci sia un segreto condiviso (SN) tra Key Fob (KF) e Receiver (R).
- In questo caso KF manda [SN, unlock] a R:

$$1.\ KF \rightarrow R : unlock, SN$$

- Non è una buona idea: l'attaccante può ascoltare il messaggi con SN e rimandarlo successivamente spacciandosi per il KF.
- Problemi:
  - La segretezza di SN è compromessa
  - R non può verificare l'autenticità della richiesta.

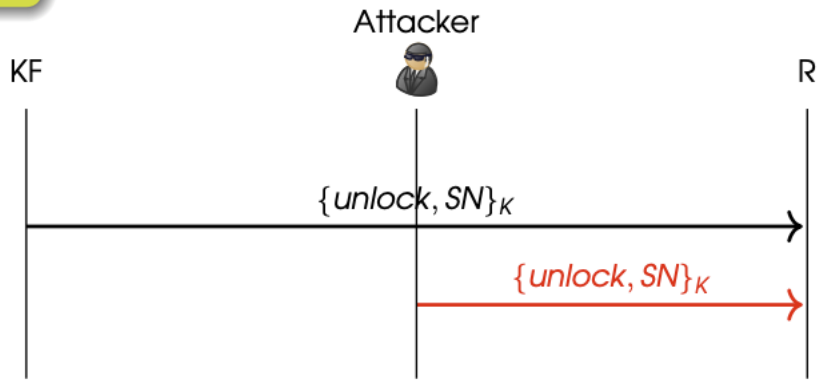
Sistema a chiave wireless /4

- Altra possibilità: proteggere la segretezza di SN.
- KF cifra la richiesta con una chiave condivisa K e manda ad R il risultato.

$$1.\ KF \rightarrow R : \{lock, SN\}_K$$

Ma:

- L'attaccante può facilmente ascoltare il messaggio e replicarlo successivamente.
- La segretezza di SN è garantita ma l'attacco è possibile comunque.



Obiettivo di sicurezza: freschezza

- Vecchio obiettivo: il ricevente manda un comando di sblocco all'Actuator solo se il Car Owner ha precedentemente schiacciato il bottone di sblocco sul Key Fob.
- Occorre aggiungere che il messaggio sia fresco, ovvero non già ricevuto in precedenza da R.
- Riformuliamo: il ricevente manda un comando di sblocco all'Actuator solo se il Car Owner ha recentemente schiacciato il bottone di sblocco sul Key Fob.

Sistema a chiave wireless: timestamp

- KF cifra un timestamp con la chiave condivisa K e manda il risultato ad R.

$$1.\ KF \rightarrow R : \{unlock, T\}_K$$

  - Non occorre mandare SN poiché KF è identificato dalla chiave condivisa.
  - Il timestamp previene il replay attack ma richiede che KF ed R abbiano clock sincronizzati.

Sistema a chiave wireless: senza timestamp

- Dopo la ricezione di un segnale, R manda a KF una challenge (ovvero un nonce N) e KF invia N cifrato con la chiave condivisa.

1. $KF \rightarrow R : hello$
2. $R \rightarrow KF : N$
3. $KF \rightarrow R : \{unlock, N\}_K$

- L'utilizzo del nonce evita replay attack.
- Non sono richiesti clock sincronizzati tra R e KF, ma sono necessari più passi del protocollo

Notazione Alice & Bob

- In un protocollo crittografico, gli eventi fondamentali sono le comunicazioni tra le parti (detti *principals*):

1. $A \rightarrow B : \{A, T_A, K\}_{K_B}$
2. $B \rightarrow A : \{B, A\}_K$

- A e B sono detti ruoli. Possono essere istanziati da ogni parte che entra in gioco nel protocollo
- La comunicazione è asincrona.
- I nomi del mittente e del ricevente non sono parte del messaggio.
- Il protocollo specifica le azioni delle parti e, conseguentemente, definisce un insieme di sequenze di eventi (dette tracce).

Assunzioni sugli agenti

- *I principals:*
  - Conoscono la propria chiave privata e quelle pubbliche degli altri.
  - Possono generare nonce e timestamp, cifrare e decifrare usando chiavi note.
  - Se onesti, implementano il protocollo in maniera corretta
- L'attaccante ha pieno controllo sulla rete ma non può violare la crittografia.

Obiettivi del protocollo e attaccante

- Un protocollo ha specifici obiettivi quali:
  - Autenticare i messaggi e legarli a chi li ha inviati
  - Garantire l'ordine temporale dei messaggi
  - Garantire la segretezza di certe parti del messaggio, come ad esempio le chiavi generate.
- Un attaccante:
  - Conosce il protocollo ma non può rompere la cifratura (Standard)
  - Può essere passivo (non fare nulla), ma può leggere le comunicazioni degli altri
  - Può essere attivo, può intercettare e generare messaggi
  - Può essere uno dei *principal* che esegue il protocollo!

Modello standard di attaccante

- Viene chiamato attaccante di tipo Dolev-Yao.
- L'attaccante
  - è attivo, ovvero può intercettare tutti i messaggi.
  - Può scomporre il messaggio nelle sue componenti ma la cifratura è sicura, ovvero per decifrare occorre avere la chiave inversa.
  - Può creare messaggi con differenti costruttori e può mandare messaggi in ogni momento.
- È il più potente modello di attaccante possibile e viene usato per analizzare le proprietà di sicurezza garantite dai protocolli.

Tipi di attacco

- Replay (or freshness) attack: riusa parti di precedenti messaggi.
- Man-in-the-middle (or parallel sessions) attack:

$$A \leftrightarrow \mathrm{M} \leftrightarrow B.$$

- Reflection attack: rimanda indietro informazioni - precedentemente trasmesse - a chi le ha inviate.
- Type flaw attack: sostituisce una parte di un messaggio con un contenuto di un altro tipo. Ad esempio, modifica una chiave o un nome con un nonce.

Esempio: Needham–Schroeder Public-Key Protocol

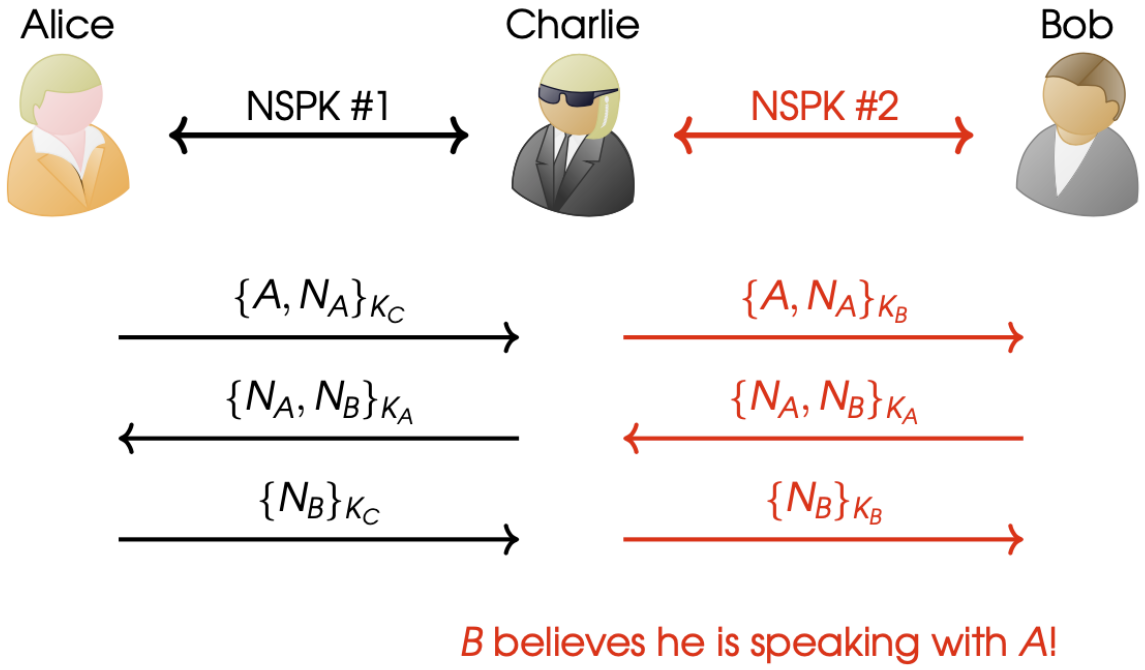
- Obiettivo: garantire la mutua autenticazione.
- Correttezza (descrizione informale):
  1. Sono Alice ed ho scelto un nonce N_{alice}
  2. Ecco il tuo nonce N_{alice}. Poiché io posso leggerlo, devo per forza essere Bob. Ho anche una challenge N_{bob} per te.
  3. Mi hai mandato N_{bob}, poiché solo Alice può averlo letto, io sono Alice.
- Notate che i *principal* possono essere coinvolte in più esecuzioni concorrenti del protocollo. L'obiettivo di sicurezza deve valere su tutte le esecuzioni concorrenti del protocollo.

1. $A \rightarrow B : \{A, N_A\}_{K_B}$
2. $B \rightarrow A : \{N_A, N_B\}_{K_A}$
3. $A \rightarrow B : \{N_B\}_{K_B}$

Needham–Schroeder Public-Key Protocol: attacco

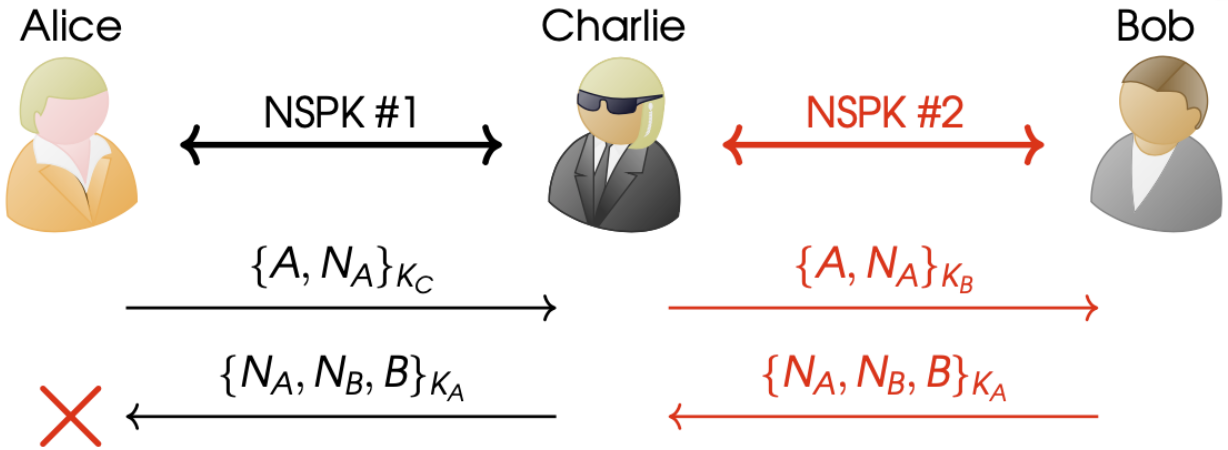
Se l'attaccante (Charlie) esegue due sessioni parallele del protocollo con Alice e Bob, alla fine riesce a far credere a Bob di parlare con Alice mentre parla con lui!

L'attacco è di tipo Man-In-The-Middle, e il protocollo, usato per decadi, non garantisce effettivamente mutua autenticazione!



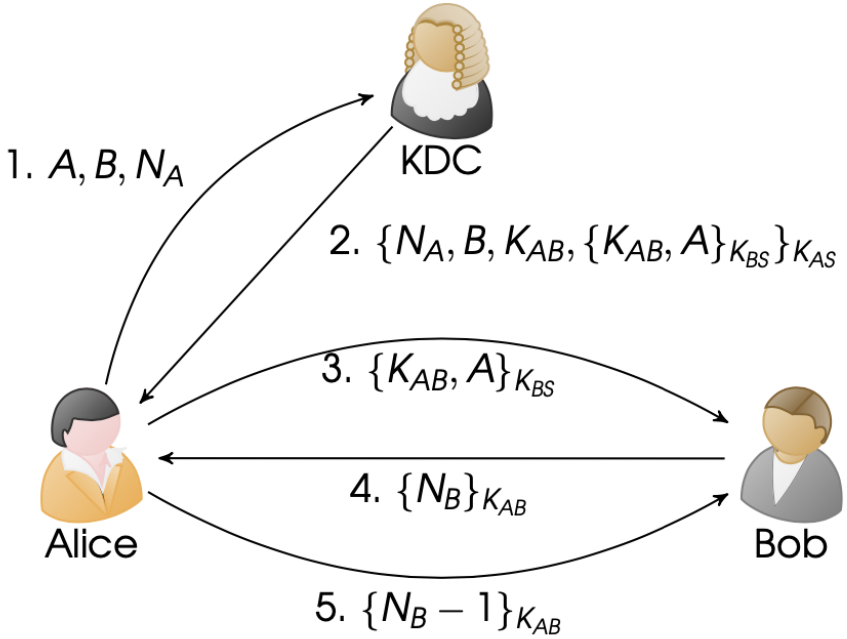
Migliorare il protocollo: la proposta di Lowe

1. $A \rightarrow B : \{A, N_A\}_{K_B}$
2. $B \rightarrow A : \{N_A, N_B, B\}_{K_A}$
3. $A \rightarrow B : \{N_B\}_{K_B}$



Il protocollo Needham-Schroeder Shared-Key

- Obiettivo: scambio di chiavi autenticato



Debolezze di Needham–Schroeder Public-Key Protocol

- *Se la chiave di sessione è* K_{AB} e viene compromessa, un attaccante può forzare *B ad accettare* nuovamente K_{AB} replicando il messaggio $\{K_{AB}, A\}K_B$. Ovvero:
  - Un attaccante può ascoltare il messaggi tra A e B e catturare il messaggio $\{K_{AB}, A\}K_{BS}$
  - Successivamente impersonificare A e iniziare una sessione con B inviando il messaggio intercettato e forzando B ad utilizzare di nuovo K_{AB}.
- Questa debolezza può venir risolta con l'aggiunta di un timestamp.