Lecture: protocollo di Needham-Schroeder

Riccardo Torre

3 dicembre 2025

$$
\begin{aligned}
A \to S&: \quad A, B \\
S \to A&: \quad \mathrm{K}_{AB} \\
B \to A&: \quad \mathrm{K}_{AB}, A
\end{aligned}
$$

Assunzione di sicurezza 1 *L'avversario è capace di ascoltare tutti i messaggi in un protocollo di sicurezza.*

Dunque è necessaria la crittografia dei messaggi.

Assunzione di sicurezza 2 (Perfect cryptography assumption) *L'attaccante in generale non può rompere la crittografia.*

$$
\begin{aligned}
A \to S&: \quad A, B \\
S \to A&: \quad E_{\mathrm{K}_{AS}}(\mathrm{K}_{AB}), E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}) \\
A \to B&: \quad E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}), A \quad \longrightarrow \quad \text{C intercetta il messaggio} \\
C \to B&: \quad E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}), D
\end{aligned}
$$

C'è un attacco peggiore di questo. In questo caso, C è autenticato (è un insider del sistema, perché S accetta la sua richiesta).

$$
\begin{aligned}
A \to S&: \quad A, B \quad \text{C intercetta il messaggio} \\
C \to S&: \quad A, C \\
S \to C&: \quad E_{\mathrm{K}_{AS}}(\mathrm{K}_{AC}), E_{\mathrm{K}_{CS}}(\mathrm{K}_{AC}) \\
C \to A&: \quad E_{\mathrm{K}_{AS}}(\mathrm{K}_{AC}), E_{\mathrm{K}_{CS}}(\mathrm{K}_{AC}) \quad \text{A non si accorge che C sta nel mezzo} \\
A \to C&: \quad E_{\mathrm{K}_{CS}}(\mathrm{K}_{AC}), A
\end{aligned}
$$

Le chiavi non sono legate crittograficamente all'identità di chi deve farne uso (**binding attack**). L'attacco appena descritto è inoltre un **man-in-the-middle** e uno **spoofing attack**.

Assunzione di sicurezza 3 *L'avversario può essere un partecipante legittimo del protocollo (un insider), o una parte esterna (un outsider) o una combinazione delle precedenti.*

$$
\begin{aligned}
A \to S\colon &\quad A, B \\
S \to A\colon &\quad E_{\mathrm{K}_{AS}}(\mathrm{K}_{AB}, B), E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}, A) \\
A \to B\colon &\quad E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}, A)
\end{aligned}
$$

Questo fix lega crittograficamente la chiave all'entità che deve utilizzarla. Ma questo protocollo è ancora vulnerabile ad un **replay attack**.

$$
\begin{aligned}
A \to S\colon &\quad A, B \quad \longrightarrow \text{C intercetta la richiesta, la blocca e si finge di essere il server} \\
C \to A\colon &\quad E_{\mathrm{K}_{AS}}(\mathrm{K}_{AB}{}', B), E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}{}', A) \quad \longrightarrow \begin{array}{c}\text{dove } \mathrm{K}_{AB}{}' \text{ è una vecchia chiave di sessione} \\ \text{intercettata da } C \text{ in una vecchia interazione} \\ \text{tra } A \text{ e } S\end{array} \\
A \to B\colon &\quad E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}{}', A)
\end{aligned}
$$

La risposta in seguito alla richiesta di A ad S è di una vecchia interazione tra A e S, solo che A non se ne accorge perché manca questa informazione nel protocollo. Bisogna legare le richieste alle risposte con i **nonce** o **challenge-responce**. Arriviamo così al protocollo di Needham-Schroeder.

Il protocollo di Needham-Schroeder

$$
\begin{aligned}
A \to S\colon &\quad A, B, \mathrm{N}_A \\
S \to A\colon &\quad E_{\mathrm{K}_{AS}}(\mathrm{K}_{AB}, B, \mathrm{N}_A, E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}, A)) \\
A \to B\colon &\quad E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}, A) \\
B \to A\colon &\quad E_{\mathrm{K}_{AB}}(\mathrm{N}_B) \quad \longrightarrow \text{viene fatto una sorta di acknowledgement} \\
A \to B\colon &\quad E_{\mathrm{K}_{AB}}(\mathrm{N}_B - 1)
\end{aligned}
$$

Anche questo protocollo è vulnerabile al **replay attack** sul messaggio $B \to A\colon E_{\mathrm{K}_{AB}}(\mathrm{N}_B)$. A ha veramente la garanzia che la chiave è fresh, ma B no!

Attacco al protocollo Needham-Schroeder In questo scenario, viene indicata con $\mathrm{K}_{\overline{AB}}$ la chiave compromessa, che C utilizza per compiere il **replay attack** verso B.

$$
\begin{aligned}
A \to S\colon &\quad A, B, \mathrm{N}_A \\
S \to A\colon &\quad E_{\mathrm{K}_{AS}}(\mathrm{K}_{AB}, B, \mathrm{N}_A, E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}, A)) \\
C \to B\colon &\quad E_{\mathrm{K}_{BS}}(\mathrm{K}_{\overline{AB}}, A) \\
B \to C\colon &\quad E_{\mathrm{K}_{\overline{AB}}}(\mathrm{N}_B) \quad \longrightarrow \text{viene fatto una sorta di acknowledgement} \\
C \to B\colon &\quad E_{\mathrm{K}_{\overline{AB}}}(\mathrm{N}_B - 1)
\end{aligned}
$$

B è convinto che sia stato A a mandargli la $\mathrm{K}_{\overline{AB}}$. Qui il nonce N_B viene inviato dopo che è stata ricevuta la chiave! Questo non permette di concludere che la chiave ricevuta provenga da un'interazione recente!

Il problema è che B non si accorge che la chiave è vecchia perché il nonce N_B non è legato crittograficamente alla chiave K_{AB} e viene usato più come un acknowledgment che come un qualcosa per verificare la freschezza della chiave di sessione. Invece A può rendersi conto che la chiave di sessione è fresca perché il nonce che ha inviato, N_A, è legato crittograficamente alla chiave K_{AB} e non può essere modificato da un attaccante nell'ipotesi che valga l'assunzione di sicurezza 2.

Fix a questo attacco

$$\begin{aligned} B \to A: &\quad B, \mathrm{N}_B \\ A \to S: &\quad A, B, \mathrm{N}_A, \mathrm{N}_B \\ S \to A: &\quad E_{\mathrm{K}_{AS}}(\mathrm{K}_{AB}, B, \mathrm{N}_A), E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}, A, \mathrm{N}_B) \\ A \to B: &\quad E_{\mathrm{K}_{BS}}(\mathrm{K}_{AB}, A, \mathrm{N}_B) \end{aligned}$$

In questo caso i nonce inviati da A e B vengono legati crittograficamente alla chiave e un attacco come il precedente non può avvenire.

Il protocollo di Needham-Schroedera chiave pubblica Questa è la versione di Needham-Schroederusando chiavi asimmetriche! L'obbiettivo di Needham-Schroederin generale è di creare una sorta di *ping autenticato*. In questo caso, l'interazione con il serve viene omessa.

$$\begin{aligned} A \to B: &\quad E_{\mathrm{K}_B}(\mathrm{N}_A, A) \\ B \to A: &\quad E_{\mathrm{K}_A}(\mathrm{N}_A, \mathrm{N}_B) \\ A \to B: &\quad E_{\mathrm{K}_B}(\mathrm{N}_B) \end{aligned}$$

Attacco man-in-the-middle a Needham-Schroedera chiave pubblica C è un insider e A non sa che è cattivo. A decide di aprire una connessione con lui.

$$\begin{aligned} A \to C: &\quad E_{\mathrm{K}_C}(\mathrm{N}_A, A) \\ C \to B: &\quad E_{\mathrm{K}_B}(\mathrm{N}_A, A) \\ B \to C: &\quad E_{\mathrm{K}_A}(\mathrm{N}_A, \mathrm{N}_B) \quad \longrightarrow \quad C \text{ non può aprire il messaggio per leggere } \mathrm{N}_B \text{ perché codificato con } \mathrm{K}_A \\ C \to A: &\quad E_{\mathrm{K}_A}(\mathrm{N}_A, \mathrm{N}_B) \\ A \to C: &\quad E_{\mathrm{K}_C}(\mathrm{N}_B) \\ C \to B: &\quad E_{\mathrm{K}_B}(\mathrm{N}_B) \end{aligned}$$

L'attaccante ha fatto uno spoofing attack nei confronti di Bob, spacciandosi per Alice. Inoltre questo è un **binding attack**.

Soluzione di Lowe

$$\begin{aligned} A \to C: &\quad E_{\mathrm{K}_C}(\mathrm{N}_A, A) \\ C \to B: &\quad E_{\mathrm{K}_B}(\mathrm{N}_A, A) \\ B \to C: &\quad E_{\mathrm{K}_A}(\mathrm{N}_A, \mathrm{N}_B, B) \quad \longrightarrow \quad B \text{ risponde con un messaggio corretto ovvero lega crittograficamente il nonce da lui prodotto alla sua identità} \\ C \to A: &\quad E_{\mathrm{K}_A}(\mathrm{N}_A, \mathrm{N}_B, B) \quad \longrightarrow \quad \text{Bob? Ma io volevo parlare con Charlie!} \end{aligned}$$

`! Abort del protocollo !`

Il protocollo non è sound.

Type flaw attack Questo è un attacco man-in-the-middle condotto da M che consiste in un **oracle attack** per fare un **type-flaw attack**.

$$
\begin{aligned}
M \to B: &\quad E_{\mathrm{K}_B}(\mathrm{N}_A, A) \\
B \to S: &\quad B, A \\
S \to B: &\quad E_{\mathrm{K}_{S^{-1}}}(A, \mathrm{K}_A) \\
B \to M: &\quad E_{\mathrm{K}_A}(\mathrm{N}_A, \underbrace{\boxed{\mathrm{N}_B, B}}_{\text{id}}) \\
M \to A: &\quad E_{\mathrm{K}_A}(\mathrm{N}_A, \boxed{\mathrm{N}_B, B}) \quad \longrightarrow \quad \text{Oracle attack + Type flaw attack!} \\
A \to M: &\quad A, \boxed{\mathrm{N}_B, B} \quad \longrightarrow \quad \mathrm{N}_B, B \text{ viene mandato in chiaro!} \\
M \to B: &\quad E_{\mathrm{K}_B}(\mathrm{N}_B)
\end{aligned}
$$

L'**oracle attack** consiste nell'inoltrare il messaggio ad A per farselo decifrare (A è l'unico a poterlo fare!). Il **type flaw attack** invece, consiste nello sfruttare una vulnerabilità nel protocollo, ovvero che i messaggi non sono tipati, dunque il nonce di B e l'identificatore di B, che sono una stringa di bit, vengono confusi da A per un identificativo perché A pensa che N_B, B stia aprendo il protocollo con lui, dicendo *"questo è il mio nonce, voglio parlare con B"*. Quindi A invia la richiesta di certificato in chiaro. Ma in realtà A sta inviando informazioni che non dovrebbero essere divulgate ossia il nonce di B e l'identificativo di B.