Doppio lucchetto

Realizzazione in PARI/GP

PARI/GP

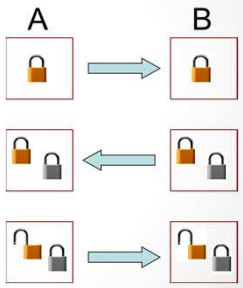
- PARI/GP è la combinazione di due ambienti:
  - PARI – libreria di funzioni (scritte in C) orientate alla teoria dei numeri
  - GP – interprete che fornisce una interfaccia a linea di comando alle funzioni di PARI che permette di fruire di un ambiente di sviluppo per applicazioni numeriche.
- L'idea alla base del linguaggio è quella di fornire per le operazioni un risultato il più corretto possibile superando le approssimazioni tipiche di altri linguaggi di programmazione.

Perché ho utilizzato PARI

- Fornisce funzioni potenti dedicate alla teoria dei numeri
- Dà la possibilità di operare con aritmetica intera con valori di lunghezza arbitraria
  - Numeri «molto grandi»

Il protocollo del doppio lucchetto

- A mette il suo messaggio per B in una scatola, che chiude con un lucchetto e invia a B.
- B mette il suo lucchetto alla scatola e la rispedisce ad A.
- A toglie il suo lucchetto e rispedisce la scatola a B.
- B toglie il suo lucchetto e legge il messaggio.



Vantaggi

- Il vantaggio principale è l'assenza dello scambio delle chiavi;
- ciò rende più difficile la decodifica del messaggio da parte di terze parti e il messaggio viaggia sempre codificato.

Svantaggi

- Il messaggio deve essere trasportato per **tre volte** quando basterebbe un unico viaggio e questo comporta una perdita di tempo.
- I due interlocutori (A e B) devono utilizzare delle tecniche di codifica similari perché altrimenti i due "lucchetti" interferiscono tra loro e B, alla fine, tolto il suo lucchetto non ottiene il messaggio originale

Applicazione

Per utilizzare il "doppio lucchetto" è necessario prima di tutto definire alcune cose:

- Il messaggio da inviare *M*
- L'alfabeto *Zn* a cui apparterrà il messaggio (e di conseguenza la lunghezza dell'alfabeto *n)*
- La funzione crittografica *f(x,Ke)* (con cui si "mette" il lucchetto *Ke*) e la sua inversa $f^{-1}(x,Kd)$ (con cui si "toglie" il lucchetto *Kd*)
- Le chiavi per criptare e decriptare Ke_A, Kd_A, Ke_B e Kd_B (sono le uniche cose che rimangono segrete durante tutto lo scambio)
- Di seguito qualche esempio.

Doppio lucchetto - cifratura con metodo di Cesare (I)

Il doppio lucchetto può essere agevolmente implementato tramite il **cifrario di Cesare**.

Definiamo allora:

- ***M***: scelto dal mittente (A)
- ***Zn***: concordato tra i due interlocutori e reso pubblico
- $\boldsymbol{Ke_A}$ e $\boldsymbol{Ke_B}$: sono scelte arbitrariamente dai due interlocutori, purchè appartengano a Zn, e sono tenute segrete
- $\boldsymbol{Kd_A = n - Ke_A}$ e $\boldsymbol{Kd_B = n - Ke_B}$
- $\boldsymbol{f(x,Ke) = (x+Ke) \bmod n}$ e $\boldsymbol{f^{-1}(x,Kd) = (x+Kd) \bmod n}$

Doppio lucchetto - cifratura con metodo di Cesare (II)

Il messaggio quindi segue questi passaggi:

- $M_1 = f(M,Ke_A) = M+Ke_A \bmod n$
- $M_2 = f(M,Ke_B) = M+Ke_A+Ke_B \bmod n$
- $M_3 = f(M,Kd_A) = M+\cancel{Ke_A}+Ke_B+\cancel{Kd_A} \bmod n = M+Ke_B \bmod n$
- $M_4 = f(M,Kd_B) = M+\cancel{Ke_B}+\cancel{Kd_B} \bmod n = M$
- Si nota quindi che durante il trasferimento il messaggio non è mai in chiaro (M_1, M_2 e M_3), mostrando il messaggio originale solo alla fine.

Doppio lucchetto - cifratura con metodo di Cesare (III)

In realtà l'utilizzo del doppio lucchetto con il cifrario di Cesare è controproducente. Infatti questo procedimento indebolisce ancora di più la (già di per sé fragile) sicurezza di Cesare.

Se infatti qualcuno riuscisse a intercettare i tre messaggi inviati, riuscirebbe facilmente a risalire alle chiavi e al messaggio originale:

- $Ke_B = M_2 - M_1 \bmod n = \cancel{M+Ke_A}+Ke_B\cancel{-M-Ke_A}$
- $Kd_B = n - Ke_B$
- $M = M_3 + Kd_B \bmod n$

Doppio lucchetto - cifratura con metodo di Massey-Omura (I)

I difetti riscontrati utilizzando per la cifratura il metodo di Cesare si possono evitare utilizzando un algoritmo più complesso come quello proposto da **Massey-Omura**.

Definiamo allora:

- ***M***: scelto dal mittente (A)
- ***Zn***: concordato tra i due interlocutori e reso pubblico
- (ricordando che n deve essere un numero primo)
- $\boldsymbol{Ke_A}$ e $\boldsymbol{Ke_B}$: sono tenute segrete e sono scelte arbitrariamente dai due interlocutori, però devono appartenere a Zn e devono essere primi rispetto a n
- $\boldsymbol{Kd_A \equiv 1/Ke_A \bmod (n-1)}$ $\boldsymbol{Kd_B \equiv 1/Kd_B \bmod (n-1)}$
- $\boldsymbol{f(x,Ke) = x^{Ke} \bmod n}$ e $\boldsymbol{f^{-1}(x,Kd) = x^{Kd} \bmod n}$

Doppio lucchetto - cifratura con metodo di Massey-Omura (II)

Il messaggio quindi segue questi passaggi:

- $M_1 = f(M,Ke_A) = M^{KeA} \bmod n$
- $M_2 = f(M,Ke_B) = M^{KeA*KeB} \bmod n$
- $M_3 = f(M,Kd_A) = M^{\cancel{KeA*KeB*KdA}} \bmod n = M^{KeB} \bmod n$
- $M_4 = f(M,Kd_B) = M^{\cancel{KeB*KdB}} \bmod n = M$
- le chiavi si annullano perchè :
- Se $\mathbf{Kd_A \equiv 1/Ke_A \bmod (n-1)}$ Allora $\mathbf{Kd_A * Ke_A = k*(n-1)+1}$
- sapendo che secondo il Piccolo Teorema di Fermat (se n è primo come nel nostro caso):
- $\mathbf{x^{n-1} \equiv 1 \bmod (n)}$
- possiamo concludere:
- $\mathbf{x^{keA*KdA} = x^{k*(n-1)+1} = x^{k*(n-1)}*x = 1*x = x}$