ESERCITAZIONI DI LABORATORIO 1

L'ALGORITMO DI CIFRATURA RSA IN C++

RSA

◀ RSA ▶ è un algoritmo di cifratura a chiave pubblica che permette di cifrare un messaggio sfruttando alcune proprietà elementari dei numeri primi. Questo cifrario prende il nome dalle iniziali dei ricercatori del MIT che lo idearono nel 1978: Rivest, Shamir e Adleman. La loro idea fu quella di sfruttare la difficoltà nel fattorizzare un numero intero: infatti la chiave pubblica utilizzate è rappresentata da un numero N ottenuto moltiplicando due numeri primi (segreti) molto grandi.

◀ Il sistema di crittografia si basa sull'esistenza di due chiavi distinte, che vengono usate per cifrare e decifrare. Se la prima chiave viene usata per la cifratura, la seconda deve necessariamente essere utilizzata per la decifratura e viceversa. La questione fondamentale è che nonostante le due chiavi siano fra loro dipendenti, non sia possibile risalire dall'una all'altra, in modo che, se anche si è a conoscenza di una delle due chiavi, non si possa risalire all'altra, garantendo in questo modo l'integrità della crittografia. ▶

Per aumentare la sicurezza del metodo è consigliabile fare uso di chiavi binarie minimo di 2048 bit, le chiavi a 512 bit sono infatti craccabili in poche ore. Le chiavi a 1024 bit sono ancor oggi largamente diffuse e utilizzate in sistemi di sicurezza di una certa importanza, infatti mediante l'algoritmo GNFS (General Number Field Sieve) si riescono a ◀ fattorizzare ▶ numeri N dell'ordine di 2^{768}-10^{231}.

◀ Attraverso l'uso di un computer cluster molto potente, del costo di centinaia di migliaia di Euro, potrebbero essere ricavate chiavi a 768 bit (ordine di 2^{768}) all'incirca dopo 6 mesi di attacco. Pertanto chiavi a 1024 bit o superiori sono al momento fuori dalla portata di un attacco realistico. ▶

Algoritmo RSA

Vediamo il funzionamento dell'algoritmo RSA:

1. Si scelgono due numeri primi, p e q sufficientemente grandi. Ricordiamo a titolo di esempio che i numeri primi utilizzati per chiavi a 2048 bit utilizzano oltre 300 cifre decimali.
2. Si calcolano due prodotti:
   - il prodotto n = p * q , chiamato modulo n
   - il prodotto f(n) = (p – 1) * (q – 1)

Come possiamo notare il calcolo di n si ottiene in pochi centesimi di secondo, mentre invece la scomposizione di n, necessaria per un attacco con il quale ottenere p e q, richiede un tempo elevatissimo: questo fa di RSA un algoritmo assai robusto.

3. Si sceglie un numero e chiamato esponente pubblico, co-primo con f(n), inferiore rispetto a f(n). Ricordiamo che e e f(n) sono primi tra di loro, cioè l'MCD tra e e f(n) è 1.
4. Si calcola il numero d chiamato esponente privato tale che (e * d) % f(n) = 1.

Esistono due metodi, i generale, per attaccare l'algoritmo RSA:
- mediante un algoritmo che, dato un testo cifrato c e la chiave pubblica (N; e), restituisca il testo decifrato m senza tuttavia ottenere la chiave privata (N; d);
- mediante un algoritmo che ricavi la chiave privata (N; d) da quella pubblica (N; e).

ESEMPIO ***Generazione chiave pubblica e privata***

Vediamo adesso un esempio che mostra come generare le due chiavi, pubblica e privata:
1. Si scelgono 2 numeri primi (in questo caso molto piccoli per facilitare il calcolo): p = 11 e q = 31.
2. Calcoliamo n = p * q = 11 * 31 = 341.
3. Calcoliamo f(n) = (p – 1) * (q – 1) = 10 * 30 = 300.
4. Prendiamo e = 17, dato che e < 300 ed e è coprimo di 300.
5. Scegliamo d = 2753.
6. Chiave pubblica k = (n, e) = (341, 17);
   Chiave privata h = (n, d) = (341, 2753).

Per cifrare il testo in chiaro (chiamato m) utilizziamo la formula seguente che utilizza la chiave pubblica:

$$c = (m^e) \% n$$

Per decifrare il testo cifrato (c) e ottenere nuovamente m, cioè il testo decifrato, utilizziamo la chiave privata:

$$m = (c^d) \% n$$

Codifica del programma

Vediamo come decodificare in linguaggio C++ l'algoritmo di cifratura secondo il metodo RSA. Si tratta di un programma a titolo di esempio, non è pertanto ammissibile l'inserimento di p e q troppo grandi, che potrebbero causare l'overflow del tipo unsigned long: in generale devono essere minori di 3232.
Inoltre sia nella cifratura che nella decifratura del messaggio, abbiamo utilizzato la formula (m % n)^e al posto di m^e % n.
Vediamo innanzi tutto la funzione che calcola l'MCD in modo ricorsivo, tramite la funzione MCD() che restituisce il massimo comun divisore tra i due parametri (a e b) passati:

```
unsigned long MCD(unsigned long a, unsigned long b)
{
  //Funzione ricorsiva che calcola MCD tra a e b
  unsigned long r = a % b;
  if (r == 0)
    return b;
  return MCD(b,r);
}
```

La funzione seguente (cifratura()) effettua la cifratura del messaggio m passato unitamente all'array k[] che contiene come nelle posizioni 0 e 1 i due valori della chiave pubblica:

```
unsigned long cifratura(unsigned long k[], unsigned long m)
{
  //funzione che cifra il messaggio m con la chiave pubblica k[]
  unsigned long s = 1;
  for (unsigned long i = 0; i < k[1]; i++)
  {
    s = (s * m) % k[0];
  }
  return s;
}
```

La funzione decifratura() invece effettua l'operazione opposta rispetto alla funzione precedente. Per fare questo riceve come parametro il messaggio cifrato (c) e la chiave privata h[], anche in questo caso rappresentata da un Array:

```
unsigned long decifratura(unsigned long h[], unsigned long c)
{
  //funzione che decifra il messaggio c con la chiave privata h[]
  unsigned long s = 1;
  for (unsigned long i = 0; i < h[1]; i++)
  {
    s = (s * c) % h[0];
  }
  return s;
}
```

Vediamo adesso la parte iniziale del programma, come possiamo notare sono inclusi i file iostream e cmath, quindi vengono indicati i prototipi delle funzioni. Le variabili di tipo unsigned long consentono di avere numeri interi positivi molto grandi, anche se non sufficienti per rendere l'algoritmo seriamente utilizzabile. Successivamente vengono letti i valori di p e q, ricordando che l'utente può inserire dei numeri primi inferiori a 3232. Quindi viene effettuato il calcolo di n e f(n).

```
#include <iostream>
#include <cmath>
using namespace std;
//Prototipi funzioni
unsigned long cifratura(unsigned long k[], unsigned long m);
unsigned long MCD(unsigned long a, unsigned long b);
unsigned long decifratura(unsigned long h[], unsigned long c);
//Main program
int main()
{
  unsigned long p;
  unsigned long q;
  //Inserimento numeri primi p e q
  cout << "Inserisci p: ";
  cin >> p;
  cout << "Inserisci q: ";
  cin >> q;
  //Calcolo del prodotto tra p e q = n
  unsigned long n = p * q;
  unsigned long f = (p - 1) * (q - 1);
  cout << "n    = p * q          : " << n << endl;
  cout << "f(n) = (p-1)*(q-1)    : " << f << endl;
```

Il codice successivo dopo aver fissato un valore predefinito per l'esponente pubblico e, verifica se il valore calcolato di f è compatibile (MCD tra e e f(n) è diverso da zero e e < f(n)). Quindi viene effettuato il controllo sull'esponente privato d. Infine viene cifrato e decifrato il messaggio attraverso la chiamata delle funzioni cifratura e decifratura:

```
  //e inserito nel programma a titolo di esempio
  unsigned long e = 17;
  //Se l'MCD tra e ed f(n) è 1 il programma prosegue altrimenti esce
  if (!(MCD(e, f) == 1 && e < f))
  {
    //Forza l'uscita dal programma
    return -1;
  }
  cout << "Esponente pubblico e     : " << e << endl;
  unsigned long d = 2753;
  //Se d (e*d)%f(n)=1 il programma prosegue altrimenti esce
  if (!((e * d) % f == 1))
  {
    //Forza l'uscita dal programma
    return -1;
  }
  cout << "Esponente privato d      : " << d << endl;
  //Array k[] chiave pubblica array h[] chiave privata
  unsigned long k[] = {n, e};
  unsigned long h[] = {n, d};
  cout << "Chiave pubblica          : k(" << k[0] << "," << k[1] << ")" << endl;
  cout << "Chiave privata           : h" << h[0] << "," << h[1] << ")" << endl;
  //1) Inserimento del numero da cifrare (max 3232)
  unsigned long m;
  cout << "Messaggio da cifrare     : m=";
  cin >> m;
  unsigned long c = cifratura(k, m);
  cout << "Cifratura di m           : c= " << c << endl;
  m = decifratura(h, c);
  cout << "Decifratura di c         : m= " << m << endl;
  system("PAUSE");
  return 0;
}
```

L'esecuzione del programma con p=11 e q=31 è la seguente:

```
Inserisci p: 11
Inserisci q: 31
n     = p * q            : 341
f(n) = (p-1)*(q-1)       : 300
Esponente pubblico e     : 17
Esponente privato d      : 2753
Chiave pubblica           : k(341,17)
Chiave privata            : h341,2753)
Messaggio da cifrare      : m=15
Cifratura di m            : c= 302
Decifratura di c          : m= 15
Premere un tasto per continuare . . .
```