

Cristiano Teodoro

(ex Istituto Superiore delle Comunicazione e delle Tecnologie dell'Informazione)

E-mail: cristiano.teodoro@virgilio.it

LA GENERAZIONE DELLA CHIAVE PRIVATA NELL'ALGORITMO CRITTOGRAFICO RSA A CHIAVE PUBBLICA

(PRIVATE KEY IMPLEMENTATION IN THE RSA PUBLIC KEY CRYPTOGRAPHIC METHOD)

CHIAVE PRIVATA :

215848789810065513208302874205921122935334449334942657295536075214506868195979917314395792274353168
956274124986211114596913065463429612455215420383072827555777122991700187967008591109257601820645413
373921927977000310981480265384701404949044705674957052534439113216316015300413232895718188959262368
760601426187

Sommario: scopo di questo articolo è quello di descrivere la realizzazione della CHIAVE PRIVATA nell'algoritmo RSA a chiave pubblica con l'utilizzo delle congruenze lineari. In effetti l'uso delle congruenze lineari viene impiegato non solo nel suddetto algoritmo ma anche in diversi altri metodi crittografici, come ad esempio nell'algoritmo DSA e nell'algoritmo di EL GAMAL, per la generazione e la verifica della firma digitale. Si inizierà pertanto con l'illustrare un algoritmo per la risoluzione della congruenza lineare $Ax = C \pmod{B}$ o dell'equazione lineare diofantea $A \cdot x - B \cdot y = C$ dove A , B e C sono numeri interi qualsiasi positivi o negativi. Come caso particolare si prenderà in considerazione la congruenza del tipo $Ax \equiv 1 \pmod{B}$ per il calcolo della Chiave Privata. Dopo una breve premessa viene descritto lo sviluppo di un numero razionale in frazione continua arrivando al calcolo del MCD di due numeri. Si passa quindi ad introdurre l'algoritmo riguardante la risoluzione di questo tipo di equazioni o congruenze attraverso i seguenti passi: calcolo delle ridotte; condizioni di risolubilità dell'equazione; risoluzione delle equazione $A \cdot x - B \cdot y = \pm 1$ e quindi delle equazione più gene-

rale $A \cdot x - B \cdot y = C$. Viene data poi una succinta panoramica dell'impiego di questo tipo di congruenza negli algoritmi crittografici citati, illustrando poi in dettaglio una sua applicazione riguardante il calcolo della Chiave Privata nell'algoritmo RSA. Per avere dei risultati concreti si sono realizzati due programma in linguaggio Qbasic. Il primo (Allegato 1) riguarda sia soluzioni della congruenza lineare generica con valori numerici di A , B e C qualsiasi, ma ciascuno in valore assoluto $< 10^5$ sia il calcolo della chiave privata nell'algoritmo RSA sempre con valori numerici $< 10^5$ per ogni primo impiegato, in quanto ci si limita nei calcoli di questo programma all'uso della doppia precisione. I risultati ottenibili per la generazione della chiave privata sono quindi da ritenersi di significato unicamente dimostrativo. Il secondo programma (Allegato 2), è dedicato espressamente per l'RSA al calcolo di chiavi private, costituite anche da centinaia di cifre decimali. Con l'impiego pertanto di quest'ultimo programma, in cui si utilizza una aritmetica a precisione multipla, si possono ottenere valori di Chiavi Private riguardanti un loro effettivo e reale impiego nel campo della Crittografia.

Abstract: the aim of this paper is to illustrate the PRIVATE KEY implementation in the RSA public key cryptographic method, by using linear congruences. Indeed their employment is used not only in the aforesaid algorithm, but also in other important cryptographic systems, such as the DSA and the ElGamal, regarding the Digital Signature generation and verification. So we start explaining an algorithm for solution of the linear congruence $Ax = C \pmod{B}$ or linear diophantine equation $A \cdot x - B \cdot y = C$ where A, B, C are whole positive or negative numbers. After a concise introduction

we explain the continued fraction expansion of rational number, attaining the GCD of two numbers. Then we illustrate the algorithm for the solution of the linear congruences by the next steps: the convergents computation; the equation resolvability conditions; the $A \cdot x - B \cdot y = \pm 1$ and $A \cdot x - B \cdot y = C$ solutions.

We give also a short survey of the linear congruences employed in the mentioned cryptographic methods and then we explain the particular computation regarding the private key of the RSA algorithm. We have

implemented two programs in Qbasic language. The former (Allegato 1) considers either the linear congruences solution with A, B, C , each $< 10^5$, or the private key computation in the RSA algorithm with every prime numerical value $< 10^5$. The private key values in this case are only demonstrative values

The second program (Allegato 2) considers expressly the private keys computation, composed by several tens or some hundreds of digit. So this program permits, by using a multiple - precision arithmetic, the achievement of numeric value for private keys, that can be of actual utilization in the cryptographic field.

1. Premessa

Per illustrare la realizzazione della **chiave privata** nell' algoritmo RSA risulta necessario illustrare innanzitutto un algoritmo dedicato alla risoluzione delle equazioni lineari diofantee $A \cdot x - B \cdot y = C$ dove A, B e C sono numeri interi qualsiasi positivi o negativi.

Risolvere queste equazioni significa trovare per le incognite x e y dei valori numerici interi che la soddisfano.

Ma perché la risoluzione di questo tipo di equazione può interessare il campo della crittografia?

Basterà per ora accennare che in diversi importanti algoritmi crittografici a chiave pubblica quali l' algoritmo RSA e l' algoritmo DSA (Digital Signature Algorithm) si deve risolvere l' equazione $A \cdot x - B \cdot y = 1$ o l' equivalente congruenza lineare $Ax = 1 \pmod{B}$ con A e B interi positivi per il calcolo di alcune grandezze o parametri riguardanti sia la generazione della **chiave privata** nell' algoritmo RSA, sia la generazione e la verifica della **Firma Digitale** nell' algoritmo DSA. Si rimanda nel seguito a maggiori dettagli sull' impiego dell' equazione in questo campo. Sotto il titolo

è riportato un esempio di valore numerico di chiave privata relativo all' algoritmo RSA, valore che può considerarsi di effettivo e reale utilizzo.

Questa nota inizierà quindi illustrando la risoluzione delle equazioni lineari diofantee

$$A \cdot x - B \cdot y = C$$

2. Sviluppo di un numero razionale in frazione continua

Per poter risolvere l' equazione lineare $A \cdot x - B \cdot y = C$ in questione occorre venire a conoscenza di alcuni argomenti essenziali riguardanti lo sviluppo di un numero razionale in frazione continua, che sono qui di seguito illustrati.

Per i vari tipi di notazioni e simboli utilizzati come pure per la validità delle formule e delle relazioni impiegate si fa riferimento a [Old].

Ogni numero razionale è una frazione della forma $\frac{A}{B}$ con A e B interi e $B \neq 0$

Si dimostra [Old] che ogni frazione, cioè ogni numero razionale lo si può esprimere nella forma seguente:

$$\frac{A}{B} = a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4 + \dots + \frac{1}{a_{n-1} + \frac{1}{a_n}}}}}$$

con un numero limitato di termini a_i (da a_1 a a_n) dove il termine a_i può essere sia un intero positivo che negativo o nullo e gli altri valori a_i sono degli interi positivi.

I vari a_i si ricavano eseguendo le seguenti divisioni successive:

$$\frac{A}{B} = a_1 + \frac{r_1}{B} \quad \text{con } a_1 \text{ e } r_1 \text{ rispettivamente}$$

quoziente e resto della divisione di A per B e quindi $0 < r_1 < B$

$$\frac{B}{r_1} = a_2 + \frac{r_2}{r_1} \quad \text{con } r_2 \text{ resto della divisione di}$$

B per r_1 e quindi $0 < r_2 < r_1$

$$\frac{r_1}{r_2} = a_3 + \frac{r_3}{r_2} \quad \text{con } r_3 \text{ resto della divisione di}$$

r_1 per r_2 e quindi $0 < r_3 < r_2$

.....

$$\frac{r_{n-3}}{r_{n-2}} = a_{n-1} + \frac{r_{n-1}}{r_{n-2}} \quad \text{con } r_{n-1} \text{ resto della divisione}$$

di r_{n-3} per r_{n-2} e quindi $0 < r_{n-1} < r_{n-2}$

$$\frac{r_{n-2}}{r_{n-1}} = a_n + \frac{0}{r_{n-1}} = a_n \quad \text{con } r_n = 0$$

I resti $r_1, r_2, r_3, \dots, r_{n-1}, r_n$ sono di valore decrescente e costituiscono una successione finita di termini il cui ultimo termine r_n è sempre di valore 0 [Old].

Sarà pertanto finita anche la successione dei termini a_i che prendono il nome di *quozienti parziali*.

La formula (1) si pone per convenzione sotto la seguente forma più pratica e concisa:

$$\frac{A}{B} = [a_1, a_2, a_3, a_4, \dots, a_{n-1}, a_n]$$

Facciamo un semplice esempio.

Sia da sviluppare in frazione continua il numero

razionale $\frac{A}{B} = \frac{1327}{271}$

Eseguito le divisioni successive come si è sopra indicato si ottengono per i diversi a_i ed r_i i seguenti valori:

$a_1 = 4$	$r_1 = 243$
$a_2 = 1$	$r_2 = 28$
$a_3 = 8$	$r_3 = 19$
$a_4 = 1$	$r_4 = 9$
$a_5 = 2$	$r_5 = 1$
$a_6 = 9$	$r_6 = 0$

pertanto si ha: $\frac{A}{B} = \frac{1327}{271} = [4, 1, 8, 1, 2, 9]$

Se si volesse invece sviluppare in frazione continua

il $\frac{271}{1327}$ è facile vedere che si ha:

$$\frac{271}{1327} = [0, 4, 1, 8, 1, 2, 9] \quad \text{che si differenzia}$$

dalla (1c) per un quoziente parziale in più: il primo quoziente che risulta di valore 0.

Questo procedimento delle divisioni successive sopra illustrato viene notoriamente utilizzato con efficacia per il calcolo del Massimo Comun Divisore di due numeri A e B che sarà indicato nel seguito nella seguente maniera: (A,B). E' questo infatti il ben noto algorithmo Euclideo (Euclidean algorithm) per il calcolo del Massimo Comun Divisore fra due numeri.

In effetti considerati due numeri interi A e B, si può dimostrare [Old] che il più piccolo resto non nullo della successione r_1, r_2, r_3, \dots è il loro (A,B).

3. Equazioni e Congruenze lineari

3.1 Calcolo delle ridotte

Tenendo presente lo sviluppo di $\frac{A}{B}$ consideriamo ora la seguente successione di grandezze

c_i che prendono il nome di *ridotte*:

$$c_1 = [a_1]; \quad c_2 = [a_1, a_2]; \quad c_3 = [a_1, a_2, a_3];$$

$$c_4 = [a_1, a_2, a_3, a_4];$$

.....

$$c_{n-1} = [a_1, a_2, a_3, a_4, \dots, a_{n-1}];$$

$$c_n = [a_1, a_2, a_3, a_4, \dots, a_{n-1}, a_n]$$

Esplicitando i due primi termini si ha:

$$c_1 = a_1 = \frac{p_1}{q_1} \quad \text{dove si è posto } p_1 = a_1 \text{ e } q_1 = 1$$

$$c_2 = a_1 + \frac{1}{a_2} = \frac{a_1 \cdot a_2 + 1}{a_2} = \frac{p_2}{q_2} \quad \text{avendo}$$

$$\text{posto } p_2 = a_1 \cdot a_2 + 1 \quad \text{e } q_2 = a_2$$

Per la successiva ridotta dopo opportuni passaggi e manipolazioni si perviene alle seguente espressione

$$c_3 = a_1 + \frac{1}{a_2 + \frac{1}{a_3}} = \frac{a_1 \cdot a_2 \cdot a_3 + a_1 + a_3}{a_2 \cdot a_3 + 1} =$$

$$= \frac{a_3 \cdot (a_1 \cdot a_2 + 1) + a_1}{a_3 \cdot a_2 + 1} = \frac{a_3 \cdot p_2 + p_1}{a_3 \cdot q_2 + q_1} = \frac{p_3}{q_3}$$

con $p_3 = a_3 \cdot p_2$ e $q_3 = a_3 \cdot q_2 + q_1$; analogamente per c_4 e c_5 dopo opportuni passaggi e manipolazioni, si potrà pervenire anche qui rispettivamente ai seguenti risultati:

$$c_4 = \frac{a_4 \cdot p_3 + p_2}{a_4 \cdot q_3 + q_2} = \frac{p_4}{q_4} \quad \text{avendo posto}$$

$$p_4 = a_4 \cdot p_3 + p_2 \quad \text{e } q_4 = a_4 \cdot q_3 + q_2$$

$$c_5 = \frac{a_5 \cdot p_4 + p_3}{a_5 \cdot q_4 + q_3} = \frac{p_5}{q_5} \quad \text{avendo posto}$$

$$p_5 = a_5 \cdot p_4 + p_3 \quad \text{e } q_5 = a_5 \cdot q_4 + q_3$$

e così via per tutti le altre ridotte sino a quella in corrispondenza della quale il valore del resto risulta di valore nullo ($r_n = 0$):

$$c_n = \frac{A}{B} = \frac{a_n \cdot p_{n-1} - p_{n-2}}{a_n \cdot q_{n-1} - q_{n-2}} = \frac{p_n}{q_n} \quad \text{con}$$

$$A = p_n = a_n \cdot p_{n-1} - p_{n-2} \quad \text{e } B = q_n = a_n \cdot q_{n-1} - q_{n-2}$$

In generale si dimostra per induzione [Old] che: i numeratori p_i e q_i delle ridotte c_i relative alla frazione continua $[a_1, a_2, a_3, a_4, \dots, a_{n-1}, a_n]$ soddisfano le uguaglianze:

$$p_i = a_i \cdot p_{i-1} + p_{i-2}$$

$$q_i = a_i \cdot q_{i-1} + q_{i-2}$$

per $i = 3, 4, 5, \dots, n$

e con i valori iniziali $p_1 = a_1$; $q_1 = 1$;

$$p_2 = a_2 \cdot a_1 + 1 \quad ; \quad q_2 = a_2$$

Da quanto illustrato si vede pertanto che si può impostare un algoritmo di tipo iterativo, ad esempio con il loop riportato nel riquadro, una volta posti $N = A$; $D = B$ e le condizioni iniziali:

$$i = 2; p_1 = a_1; q_1 = 1; p_2 = a_2 \cdot a_1 + 1; q_2 = a_2$$

```
Inizio loop : i = i + 1
              a_i = floor(N/D) : r_i = N - a_i * D
              p_i = a_i * p_{i-1} + p_{i-2}
              q_i = a_i * q_{i-1} + q_{i-2}
              se r_i = 0 esci dal loop
              N = D
              D = r_i
              p_{i-2} = p_{i-1} : p_i = p_{i-1}
              q_{i-2} = q_{i-1} : q_i = q_{i-1}
              torna a inizio loop
              n = i: REM r_n = 0
              fine
              M.C.D. (A, B) = r_{n-1}
```

Algoritmo per il calcolo dei valori p_i e q_i fino alla iterazione $i = n$ in corrispondenza della quale si ha resto $r_n = 0$

4 - Risoluzione della equazione

$$A \cdot x - B \cdot y = C$$

4.1 Condizioni di risolvibilità

Si vogliono trovare per x e y i valori interi soddisfacenti la suddetta equazione.

L'equazione può essere scritta così:

$$A \cdot x = y \cdot B + C, \text{ mettendo con ciò in evidenza}$$

che y e C rappresentano rispettivamente il quoziente ed il resto della divisione di $A \cdot x$ per B .

Tale equazione si può esprimere anche come congruenza assumendo in tal caso la seguente forma:

$$A \cdot x = C \pmod{B}$$

e si enuncia dicendo che $A \cdot x$ è congruo a C modulo B.

Quando si trattano congruenze il valore del modulo B è da considerarsi di valore positivo.

Questa espressione può essere anche messa sotto la seguente forma: $x = C \cdot A^{-1} \pmod{B}$;

per $C = 1$ si ha $x = A^{-1} \pmod{B}$; in questo caso x viene chiamato *inverso moltiplicativo di A*.
Portando C nel primo membro di (3b) si ha

$$A \cdot x - C = 0 \pmod{B};$$

si dice allora che $A \cdot x - C$ è congruo a 0 modulo B.

La prima cosa da appurare è vedere se la (3a) o la (3b) ammettono soluzioni.

L'equazione e quindi la corrispondente congruenza sono risolvibili solo se sussiste la seguente condizione [Old]:

C è divisibile per (A,B)

Questa condizione equivale ad una qualsiasi di queste due condizioni:

A e B sono primi tra loro.

A e B non sono primi tra loro, ma un loro divisore comune è anche divisore di C.

4.2 Risoluzione della equazione

$$A \cdot x - B \cdot y = 1 \tag{4}$$

Rivolgiamo ora l'attenzione all'equazione

$A \cdot x - B \cdot y = 1$ dove A e B possono essere degli interi sia positivi che negativi.

Innanzitutto perché l'equazione sia risolvibile tenendo presenti le condizioni dette sopra, occorre che

A e B siano primi tra loro e che quindi sia (A,B) = 1 per A e B dello stesso segno e (A,B) = -1 per A e B di segno opposto.

Per i valori generici di p_i e di q_i si dimostra sempre per induzione (vedi [Old]) che sussiste la

$$\text{seguinte relazione: } p_i \cdot q_{i-1} - p_{i-1} \cdot q_i = (-1)^i$$

Applicando l'algoritmo iterativo che è esposto nel riquadro, si esce dal ciclo alla iterazione per cui si ha resto $r_i = 0$; in quest'ultima iterazione che chiameremo iterazione *n-esima* si sono acquisiti i valori di p_{n-1} , q_{n-1} , p_n e q_n per cui vale la relazione $p_n \cdot q_{n-1} - p_{n-1} \cdot q_n = (-1)^n$

Ma dalla (2) si osserva che $A = p_n$ e $B = q_n$, per cui si perviene alla seguente relazione:

$$p_n \cdot q_{n-1} - p_{n-1} \cdot q_n = A \cdot q_{n-1} - B \cdot p_{n-1} = (-1)^n$$

Se n è pari si ottiene $A \cdot q_{n-1} - B \cdot p_{n-1} = 1$.

Si può allora vedere immediatamente dal confronto con la (4) che i valori q_{n-1} e p_{n-1} sono i valori interi rispettivamente di x e di y che soddisfano l'equazione. Tuttavia, per tenere conto che A e B possono essere sia positivi che negativi e quindi anche di segno opposto, gli effettivi valori soddisfacenti l'equazione sono dati dalle seguenti

$$\text{formule: } x_0 = \frac{q_{n-1}}{(A,B)} \quad \text{e} \quad y_0 = \frac{p_{n-1}}{(A,B)}$$

con (A,B) = 1 se A e B sono dello stesso segno e (A,B) = -1 se A e B sono di segno opposto.

Si può mostrare [Old] poi che anche valori di x del tipo $x_0 + B \cdot k$ e i corrispondenti valori di

y del tipo $y_0 + A \cdot k$ dove k è un intero positivo o negativo qualsiasi

soddisfano anch'essi l'equazione (4). Pertanto tutti gli infiniti valori risolutivi di x e di y si possono ottenere dalle seguenti formule:

$$x = x_0 \pm B \cdot k; \quad y = y_0 \pm A \cdot k$$

con $k = 1, 2, 3, \dots$

Qualora n fosse dispari si può procedere modificando lo sviluppo (1b) nella seguente maniera come mostrato in [Old]:

$$\frac{A}{B} = [a_1, a_2, a_3, a_4, \dots, a_{n-1}, a_n]$$

$$= [a_1, a_2, a_3, a_4, \dots, a_{n-1}, a_n - 1, 1]$$

senza alterarne il valore, riportando però ad un numero pari lo sviluppo dei quozienti parziali

4.3 Risoluzione dell'equazione

$$A \cdot x - B \cdot y = -1$$

L'algoritmo per risolvere questa equazione è analogo a quello utilizzato per il termine noto di valore +1.

Pervenendo alla n-esima iterazione alla relazione

$$p_n \cdot q_{n-1} - p_{n-1} \cdot q_n = A \cdot q_{n-1} - B \cdot p_{n-1} = (-1)^n$$

se n è di valore dispari si ottengono $x_0 = q_{n-1}$ e

$y_0 = p_{n-1}$ quali soluzioni dell'equazione.

Se n risultasse pari si procede anche qui scomponendo il quoziente parziale a_n nei due quozienti parziali a_{n-1} e 1 senza alterare il valore dello sviluppo, riportando però ad un valore dispari il numero dei quozienti parziali.

Per tenere conto che A e B possono essere di segno opposto, gli effettivi valori che soddisfano l'equazione sono dati dalle seguenti formule :

$$x_0 = \frac{q_{n-1}}{(A, B)} \quad \text{e} \quad y_0 = \frac{p_{n-1}}{(A, B)}$$

dove risulta $(A, B) = 1$ se A e B sono dello stesso segno e $(A, B) = -1$ se A e B sono di segno opposto.

4.4 Risoluzione dell'equazione

$$A \cdot x - B \cdot y = C$$

Una volta risolta la $A \cdot x - B \cdot y = 1$ con la sua soluzione particolare e è facile vedere che per l'equazione generale $A \cdot x - B \cdot y = C$ con A, B, C interi positivi o negativi, la corrispondente soluzione particolare è :

$$x_{00} = C \cdot x_0 = C \cdot \frac{q_{n-1}}{(A, B)} ; \quad y_{00} = C \cdot y_0 = C \cdot \frac{p_{n-1}}{(A, B)}$$

e tutti gli altri infiniti valori di x e di y che la soddisfano sono dati da:

$$x_k = x_{00} + B \cdot k ; \quad y_k = y_{00} + A \cdot k$$

con $k = 1, 2, 3, 4, \dots$

Se si vuole poi ricercare la soluzione x, y tale per cui x assume il *minimo valore positivo*, essa è data dai seguenti valori:

$$x_m = x_{00} - \left\lfloor \frac{x_{00}}{nx} \right\rfloor \cdot nx \quad \text{con } nx = \frac{B}{(A, B)}$$

$$y_m = y_{00} - \left\lfloor \frac{y_{00}}{ny} \right\rfloor \cdot ny \quad \text{con } ny = \frac{A}{(A, B)}$$

con $(A, B) = 1$ se A e B sono dello stesso segno e $(A, B) = -1$ se A e B sono di segno opposto.

5. Applicazione alla Crittografia

Un'importante applicazione in cui si utilizza l'equazione diofantea del tipo sopra illustrato, in particolare del tipo $A \cdot x - B \cdot y = 1$ con A e B entrambi positivi, si ha nell'ambito della crittografia a chiave pubblica e precisamente nell'algoritmo RSA, nell'algoritmo DSA (Digital Signature Algorithm), dedicato quest'ultimo espressamente alla firma elettronica, e nell'algoritmo di *El Gamal* come pure in altri algoritmi crittografici meno noti.

Considerando i due importanti metodi crittografici RSA e DSA si riportano qui solo le specifiche riguardanti alcuni loro parametri senza prendere in considerazione una loro descrizione per la quale si può rimandare a vari testi o articoli, vedi ad esempio [FeL], [G.U.], [Hel] [Men], [Sch], [Sga], i siti Internet [Men], [Oli] e in particolare il sito Internet del Liceo Classico M.Foscarini - Venezia [L.F], per semplicità e chiarezza nell'esposizione del principio di funzionamento dell'algoritmo RSA. Per quanto riguarda questo algoritmo si illustrerà in dettaglio più avanti come si può trovare la **Chiave privata** con l'ausilio delle congruenze lineari

5.1 Chiavi nell' Algoritmo RSA

Chiave pubblica: è costituita da due numeri denominati convenzionalmente uno con il simbolo n , l'altro con il simbolo e .

n è un numero composto da due numeri primi p, q grandi: $n = p \cdot q$;

e è un numero random positivo $< \Phi$ dove

$\Phi = (p-1) \cdot (q-1)$ è denominata funzione di Eulero;

e deve essere tale per cui $(e, \Phi) = 1$. Se si pone e primo non occorre naturalmente verificare la condizione che sia $(e, \Phi) = 1$.

Chiave privata: è un numero d legato ad e ed a Φ dalla seguente relazione: $e \cdot d - \Phi \cdot y = 1$ che possiamo anche porre sotto la forma

$$e \cdot d = 1 \pmod{\Phi} \quad \text{od anche} \quad d = e^{-1} \pmod{\Phi}$$

Pertanto d risulta essere l'*inverso moltiplicativo* di e modulo Φ

Per trovare il valore di d , una volta noti i valori di Φ e di e , si dovrà risolvere pertanto l'equazione diofantea. $e \cdot x - \Phi \cdot y = 1$. Il valore di x soddisfacente l'equazione sarà il valore della chiave privata d .

5.2 Algoritmo DSA

Per questo più complesso algoritmo, dedicato alla firma digitale, si accenna brevemente solo a quanto segue, facendo riferimento per la denominazione dei simboli e per maggiori dettagli a [Men].

Chiave pubblica: è costituita da quattro parametri denominati p, q, α, y dove:

q è un numero primo random costituito da 48 cifre decimali;

p è un numero primo costituito da non meno di 154 cifre decimali e tale per cui $(p-1)$ sia divisibile per q ;

$$\alpha = g^{\frac{p-1}{q}}$$

essendo g un intero positivo random $< p$;

$$y = \alpha^a \pmod{p}$$

dove a è un intero positivo random

Chiave privata: è costituita dal solo parametro a (sopra definito)

Per la generazione di questi parametri pertanto non risulta implicata nessun'operazione di risoluzione di equazioni lineari diofantee

Questo tipo di operazione viene invece impiegata nei riguardi sia della **generazione** della **Firma digitale** (*Signature generation*) sia in quella per la **verifica** della **Firma** (*Signature verification*).

Senza entrare nei particolari, nel computo dei parametri richiesti per la generazione della firma digitale, uno di essi risulta esser l'inverso moltiplicativo modulo q di un numero segreto k intero positivo scelto in modo random e minore di q ; si

deve effettuare quindi il calcolo di $k^{-1} \pmod{q}$ e quindi risolvere una equazione lineare diofantea. Analogamente anche nelle operazioni di verifica per trovare uno dei suoi parametri occorre effettuare il calcolo dell'inverso moltiplicativo di un parametro definito nelle operazioni dedicate alla generazione.

5.3 Calcolo della Chiave privata nell'algoritmo RSA

Vediamo ora più in dettaglio quali sono le operazioni necessarie per il calcolo della Chiave Privata nell'algoritmo crittografico RSA.

Per calcolare la chiave privata d , come si è già detto occorre risolvere l'equazione

$$e \cdot x - \Phi \cdot y = 1$$

Tenendo conto di quanto esposto nei paragrafi precedenti una volta conosciute le grandezze Φ

ed e il valore risolutivo $x_0 = \frac{q_{n-1}}{(e, \Phi)}$ sarà la

Chiave privata d cercata. Pertanto si ha

$$d = x_0 = \frac{q_{n-1}}{(e, \Phi)}$$

Poiché poi deve essere $(e, \Phi) = 1$ si avrà $d = q_{n-1}$.

Si mostrano qui di seguito esempi di calcolo di tale grandezza con l'ausilio di due programmi in Qbasic.

Il primo programma, riportato in **Allegato I**, utilizza l'aritmetica disponibile sul PC col Qbasic che non va oltre la doppia precisione; quindi con esso per non avere risultati approssimati si devono trattare per i tre parametri e, p, q valori positivi tali per cui la somma complessiva delle cifre che li compongono non superi il valore 16

(ad esempio e composto al massimo da 4 cifre con p e q composti da 6 cifre), i quali quindi si possono considerare solo come esempi esplicativi dell'algoritmo.

Essendo in effetti questo programma rivolto alla risoluzione dell'equazione generale

$$A \cdot x - B \cdot y = C$$

anche i valori dei coefficienti **A, B, C**, che possono essere numeri interi sia positivi che negativi, non devono avere in linea di massima ciascuno un valore assoluto maggiore 10^5 di, avvertendo che per ottenere soluzioni esatte qualsiasi risultato ottenibile nei calcoli non deve avere valore $> 10^5$ in relazione ad una qualunque delle tre opzioni offerte dal programma:

- la prima dedicata alla risoluzione di una generica equazione lineare diofantea;
- la seconda relativa espressamente al calcolo della chiave privata d una volta introdotti da input la grandezza e ed i due numeri primi p e q , da cui si può ricavare il valore di

$\Phi = (p-1) \cdot (q-1)$ e il valore di $n = p \cdot q$ che costituisce insieme ad e la **Chiave pubblica** (n, e) nell'RSA.

- la terza dedicata ad un esempio di calcolo di chiave privata d .

Il secondo programma sempre in Qbasic, riportato in **Allegato 2**, è dedicato anch'esso alla risoluzione di equazioni lineari diofantee, ma a differenza del programma in allegato 1 è in grado di elaborare numeri grandi e quindi di calcolare **effettivi** valori di chiavi private poiché le operazioni di calcolo sono programmate per una loro utilizzazione in aritmetica a precisione multipla.

Esso comunque per non appesantirlo troppo è dedicato solo alla generazione della **chiave privata**, quindi alla risoluzione dell'equazione del tipo

$$e \cdot x - \Phi \cdot y = 1 \quad \text{e presenta due opzioni:}$$

Prima opzione: dedicata al calcolo della chiave privata d introducendo dall'esterno e quindi da richiesta di INPUT tre numeri primi grandi, costi-

tuiti da stringhe di tipo numerico, riguardanti i seguenti tre parametri:

$e^{(1)}$ numero primo random ;

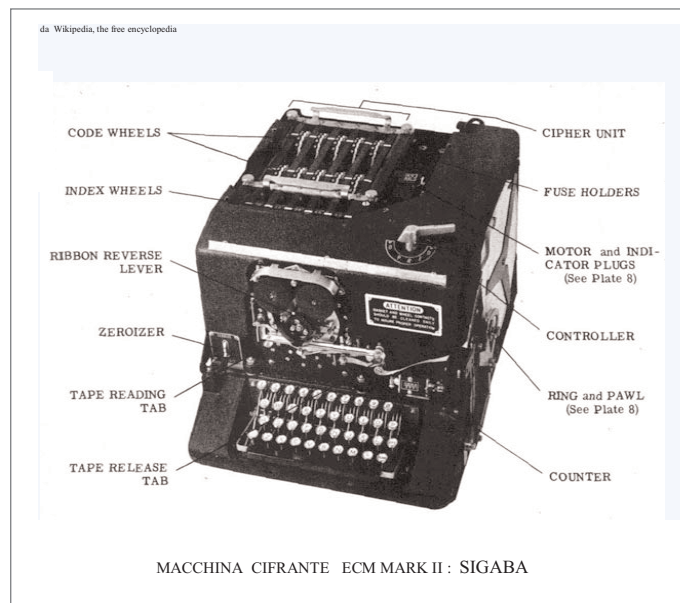
p numero primo random grande (il numero p per un suo effettivo impiego dovrebbe essere costituito da almeno 160 cifre);

q numero primo random (di almeno 140 cifre);

i numeri p e q devono essere tali per cui $n = p \cdot q$ risulti costituito da almeno 309 cifre decimali (digit) pari a 1024 bit (vedi Art.4 dell'ALLEGATO TECNICO di [G.U.]).

Si suppone che i suddetti tre parametri siano stati preliminarmente creati tramite calcoli già effettuati con opportuni programmi [vedi Nota] e tali per cui $\text{MCD}(e, \Phi) = 1$ dove $\Phi = (p-1) \cdot (q-1)$

Seconda opzione: relativa ad un esempio con i valori dei tre parametri suddetti già inseriti nel programma come dati costituiti da stringhe numeriche e quindi immediatamente disponibili.



Nota: i numeri primi grandi e, p, q possono ricavarsi in genere tramite l'ausilio di appositi pacchetti software matematici (vedi ad esempio "Mathematica", "Maple", ecc.). I numeri primi casuali riportati negli esempi e nel testo del programma riportato nell'Allegato 2 sono invece stati creati ciascuno con un tempo di calcolo di qualche minuto, utilizzando un apposito programma in Qbasic dedicato alla generazione e alla verifica di numeri primi relativamente grandi, descritto e riportato in [Teo]. Inoltre per il calcolo dei valori di Φ e del MCD si sono utilizzati opportuni programmi in aritmetica a precisione multipla realizzati dall'autore.

(1) La scelta del più opportuno valore di e richiede particolare attenzione in quanto si deve tener conto di due esigenze fra di loro contrastanti: un valore piccolo di e comporta un più veloce processo di cifratura; d'altra parte è opportuno avere grandi sia il valore di e che quello della chiave privata d [Sch] contro possibili attacchi crittoanalitici.

RIFERIMENTI

- [FeLu] P. Ferragina e F. Luccio, CRITTOGRAFIA, Capp. 8 e 9 - 2001 Bollati Boringhieri editore s.r.l , Torino
- [G.U.] DECRETO DEL PRESIDENTE DEL CONSIGLIO DEI MINISTRI 8 febbraio 1999 - GAZZETTA UFFICIALE DELLA REPUBBLICA ITALIANA, Anno 140° - Numero 87 del 15 aprile 1999
- [Hel] M. E. Hellman, The Mathematics of Public - Key Cryptography, Scientific American, v. 241 n. 81, Aug. 1979
- [L.F.] <http://www.liceofoscarini.it/studenti/crittografia/critto/rsa/metodo.html>
- [Men] A.J. Menezes , P.C. van Oorschot, S.a Vanstone - HANDBOOK of APPLIED CRYPTOGRAPHY Ch. 11 - <http://www.cacr.math.uwaterloo.ca/hac/>
- [Old] C.D. Olds , FRAZIONI CONTINUE, Zanichelli, Bologna 1970
- [Oli] A. Oliva , Cifratura RSA , <http://www.matematicamente.it/approfondimenti/index.html>
- [Sch] B. Schneier, APPLIED CRYPTOGRAPHY, Capp. 12 e 13 - 1994 John Wiley & Sons, Inc.
- [Sga] A Sgarro, CRITTOGRAFIA, Cap. 7 - Prima edizione 1986 , franco muzzio & c. editore
- [Teo] C. Teodoro, Verifica e Generazione di Numeri Primi relativamente grandi <http://www.comunicazioni.it/it/index.php?IdPag=844>

Allegato I

DEFDBL A-Z

```

PRINT "----- Programma LCONGRSA.BAS -----"
PRINT "il presente Programma prevede tre opzioni:"
PRINT "1^ OPZIONE: RISOLUZIONE DELLA EQUAZIONE DIOFANTEA  $A * x - B * y = C$ "
PRINT " se si vuole questa opzione battere un tasto qualsiasi e poi"
PRINT " sul tasto relativo alla cifra 1"
PRINT "2^ OPZIONE: CALCOLO DELLA CHIAVE PRIVATA d NELL'ALGORITMO RSA"
PRINT " se si vuole questa opzione battere un tasto qualsiasi e poi";
PRINT " sul tasto relativo alla cifra 2"
PRINT "3^ OPZIONE: ESEMPIO DI CALCOLO DELLA CHIAVE PRIVATA d NELL'ALGORITMO RSA"
PRINT " se si vuole questa opzione battere un tasto qualsiasi e poi"
PRINT " sul tasto relativo alla cifra 3"
PRINT : PRINT "battere ora un tasto qualsiasi": PRINT
DO: y$ = INKEY$: LOOP WHILE y$ = ""
DIM a(100), r(100), p(100), q(100), x(100), y(100)
5 INPUT "INTRODURRE L'OPZIONE DESIDERATA e battere poi il tasto Invio :", oz$
IF oz$ = "1" GOTO 10
IF oz$ = "2" GOTO 20
IF oz$ = "3" GOTO 30
IF oz$ <> "1" OR oz$ <> "2" OR oz$ <> "3" GOTO 5

REM ----- OPZIONE 1 -----
10 PRINT : INPUT "A"; e: INPUT "B"; f: INPUT "C"; c:
le = LEN(STR$(e)): lp = LEN(STR$(f)): lq = LEN(STR$(c)):
lt = le + lp + lq - 3
IF lt > 16 THEN PRINT "qualche valore è troppo grande: ricomincia": GOTO 10
PRINT "l'equazione da risolvere è la seguente:";

```

```
PRINT "("; e; ") * x - ("; f; ") * y =" ; c
an = 0
GOTO 40
```

```
REM ----- OPZIONE 2 -----
```

```
20 : PRINT
PRINT "ATTENZIONE: ACCERTARSI CHE I VALORI DA INTRODURRE SIANO NUMERI PRIMI"
PRINT
```

```
pare: INPUT "introduci numero primo e:", e
IF e / 2 = INT(e / 2) THEN PRINT "reintrodurre il valore di e": GOTO pare
parp: INPUT "introduci numero primo p:", np
IF np / 2 = INT(np / 2) THEN PRINT "reintrodurre il valore di p": GOTO parp
parq: INPUT "introduci numero primo q:", nq
IF nq / 2 = INT(nq / 2) THEN PRINT "reintrodurre il valore di q": GOTO parq
z2$ = "0": le = LEN(STR$(e)): lp = LEN(STR$(np)): lq = LEN(STR$(nq)):
lt = le + lp + lq - 3: PRINT "!!!!"; lt
IF lt > 16 THEN PRINT "qualche valore è troppo grande: ricomincia": GOTO 20
n = np * nq: f = (np - 1) * (nq - 1)
PRINT "i due parametri della chiave pubblica nell'algorithm RSA sono:"
PRINT "n = p * q = "; n; " e = "; e:
PRINT "-----"
PRINT "il valore della funzione di Eulero f = (p-1)*(q-1) è:"; f
PRINT : PRINT "la chiave privata d nell'algorithm RSA si calcola risolvendo";
PRINT "l'equazione lineare diofantea:";
PRINT : PRINT "          "; e; " * x - "; f; " * y = 1"
GOTO 40
```

```
REM ----- OPZIONE 3 : ESEMPIO -----
```

```
30 PRINT
PRINT "i 3 numeri primi seguenti sono posti nel listato del programma"
PRINT : e = 8627: dp = 749129: dq = 94321: f = (dp - 1) * (dq - 1):
PRINT "e ="; e; " p ="; dp; " q ="; dq
PRINT : PRINT "funzione di Eulero: f = (p-1)*(q-1) ="; f
PRINT : PRINT "la chiave privata d nell'algorithm RSA si calcola risolvendo";
PRINT "l'equazione lineare diofantea:";
PRINT : PRINT "("; e; ") * x - ("; f; ") * y = 1"
REM -----PARTE COMUNE ALLE TRE OPZIONI-----
40 IF ABS(e) = ABS(f) THEN an = 1
IF an = 1 AND c / e <> INT(c / e) THEN PRINT "l'equazione ŝ impossibile": END
IF an = 1 THEN PRINT "l'equazione diofantea "("; e; ") x - ("; f; ") y = "; c;
IF an = 1 THEN PRINT "ŝ soddisfatta per un qualsiasi valore intero di";
IF e = -f THEN PRINT " x con y = "; c / e; " - x": END
IF e = f AND c / e < 0 THEN PRINT " x con y = x +"; ABS(c / e); : END
IF e = f AND c / e > 0 THEN PRINT " x con y = x -"; ABS(c / e); : END
```

```

REM -----CALCOLO DEL MASSIMO COMUN DIVISORE -----
j = 0: e1 = e: f1 = f
inizio: j = j + 1: ' PRINT j;
  a(j) = INT(e1 / f1): rs = e1 - a(j) * f1
  IF rs = 0 THEN j1 = j: GOTO masscom
  e1 = f1: f1 = rs
GOTO inizio
masscom: mcd = f1
PRINT : PRINT "si trova: MCD("; e; ", "; f; ") ="; f1
REM -----
'PRINT "l'equazione è risolvibile ed ha la seguente soluzione;"
IF c / mcd <> INT(c / mcd) THEN PRINT "l'equazione ("; e; ") x - ("; f; ") y";
IF c / mcd <> INT(c / mcd) THEN PRINT " ="; c; " non ha soluzioni. ": END
IF c / mcd <> INT(c / mcd) AND oz$ = "2" THEN z2$ = "cvp"
IF z2$ = "cvp" THEN PRINT " cambiare i valori dei parametri": GOTO 20

IF j1 = INT(j1 / 2) * 2 GOTO term
IF a(j1) > 1 THEN a(j1) = a(j1) - 1: a(j1 + 1) = 1: j1 = j1 + 1: GOTO term
IF a(j1) = 1 THEN a(j1 - 1) = a(j1) + 1: j1 = j1 - 1
term: p(0) = 1: p(1) = a(1)
      q(0) = 0: q(1) = 1
FOR k = 2 TO j1
  p(k) = a(k) * p(k - 1) + p(k - 2): ' PRINT "p("; k; ")="; p(k),
  q(k) = a(k) * q(k - 1) + q(k - 2): ' PRINT "q("; k; ")="; q(k)
NEXT k
xo = q(j1 - 1): 'PRINT "xo="; xo,
yo = p(j1 - 1): 'PRINT "yo="; yo
REM dell'equazione diofantea ax - by = I
ss = e * xo - f * yo
IF oz$ = "1" GOTO opzione1
REM ----- OPZIONI 2 E 3 -----
PRINT : PRINT "      xo ="; xo, "yo ="; yo
PRINT "con xo ed yo soluzioni della suddetta equazione"
d = xo
IF ss <> 1 THEN PRINT "poiché risulta e * d - f * y ="; ss; "<> 1 il valore"
IF ss <> 1 THEN PRINT "trovato per d non puo considerarsi Chiave Privata": END
PRINT : 'PRINT "l'equazione è risolvibile in quanto M.C.D.(e,f) = 1"
IF ss = 1 THEN PRINT "la chiave privata cercata è d = xo ="; d:PRINT
PRINT "in quanto si ha:"
PRINT : PRINT "e*d - f*yo ="; e; "d"; "-"; f; "yo"; yo
PRINT "e*d - f*yo ="; e * d; "-"; f * yo; "="; ss
IF ss = 1 THEN END

```

```

REM ----- OPZIONE I -----
opzioneI: xoo = c * q(jl - l) / mcd
yoo = c * p(jl - l) / mcd
PRINT:"l'equazione è risolvibile ed ha la seguente soluzione:"
PRINT "xoo = c*xo/MCD = "; c; "*"; xo; "/"; fl; "="; xoo
PRINT "yoo = c*yo/MCD = "; c; "*"; yo; "/"; fl; "="; yoo
xa = ABS(xoo): ya = ABS(yoo):
IF xa > 10 ^ 16 THEN PRINT " valore di xoo arrotondato e quindi non preciso"
IF ya > 10 ^ 16 THEN PRINT " valore di yoo arrotondato e quindi non preciso"
IF xa > 10 ^ 16 OR ya > 10 ^ 16 THEN nr = 1: PRINT "pertanto non è possibile";
IF nr = 1 THEN PRINT "risolvere l'equazione con valori esatti per x ed y": END
nx = f / mcd: ny = e / mcd:
xom = xoo - INT(xoo / nx) * nx
yom = yoo - INT(yoo / ny) * ny
'PRINT "          xoo ="; xoo; ", " yoo = "; yoo
PRINT : 'PRINT "l' equazione "; e; "x -"; f; "y = "; c
PRINT "I valori minimi di x e di y che soddisfano l'equazione:"
PRINT : PRINT "("; e; ") * x -("; f; ") * y = "; c: PRINT : PRINT "sono:";
PRINT " xom = "; xom; ", yom ="; yom
PRINT : ' PRINT
PRINT"          VERIFICA          "
PRINT : PRINT "A * xoo - B * yoo = ";
PRINT "("; e; ")*( "; xoo; ") - ("; f; ")*( "; yoo; ") = "; e * xoo - f * yoo
w = e * xoo - f * yoo: IF w = c THEN PRINT "xoo e yoo sono valori VALIDI"
PRINT : PRINT "verifica con valori minimi xom e yom :"
PRINT "A * xom - B * yom = ";
PRINT "("; e; ")*( "; xom; ") - ("; f; ")*( "; yom; ") = "; e * xom - f * yom
m = xom * e - yom * f: IF m = c THEN PRINT "xom e yom sono valori VALIDI"

IF oz$ <> "I" THEN END
PRINT "-----ALTRE SOLUZIONI-----"
PRINT "altre soluzioni di valore intorno a xom ed yom sono le seguenti:"
h = -4: h0 = h
acca: h = h + 1: IF h > 3 THEN END
IF h = 0 THEN PRINT "xom ="; xom, "yom ="; yom: GOTO acca x = xom + h * f / mcd: PRINT "x(";
h; ")="; x,
y = yom + h * e / mcd: PRINT "y("; h; ")="; y
'PRINT s; "x("; h; ") - "; r; "y("; h; ")="; s * x(h) - r * y(h)
GOTO acca
DO: y$ = INKEY$: LOOP WHILE y$ = ""
CLS : END

```

Esempi di risoluzione tramite il programma in Qbasic riportato in Allegato I

Si illustra per ciascuna delle 3 opzioni un esempio di calcolo, riportando ciò che compare sullo schermo del monitor dopo aver fatto partire il programma ed aver inserito gli eventuali dati richiesti riguardanti l'opzione scelta.

1° Esempio relativo alla 1^a opzione

Eseguendo il programma riportato in allegato, una volta introdotti da input i valori di A, B , C relativi alla equazione diofantea $A \cdot x - B \cdot y = C$, sullo schermo del monitor comparirà quanto segue:

```

----- Programma LCONGRSA.BAS -----
il presente Programma prevede tre opzioni:
1^ OPZIONE: RISOLUZIONE DELLA EQUAZIONE DIOFANTEA A * x - B * y = C
  se si vuole questa opzione battere un tasto qualsiasi e poi il tasto relativo alla cifra 1
2^ OPZIONE: CALCOLO DELLA CHIAVE PRIVATA d NELL'ALGORITMO RSA
  se si vuole questa opzione battere un tasto qualsiasi e poi il tasto relativo alla cifra 2
3^ OPZIONE: ESEMPIO DI CALCOLO DELLA CHIAVE PRIVATA d NELL'ALGORITMO RSA
  se si vuole questa opzione battere un tasto qualsiasi e poi il tasto relativo alla cifra 3
  battere ora un tasto qualsiasi

INTRODURRE L'OPZIONE DESIDERATA e battere poi il tasto Invio :1
A? -39801
B? 7468
C? -56113
l'equazione da risolvere è la seguente:(-39801 ) * x - ( 7468 ) * y =-56113

si trova: MCD(-39801 , 7468 ) = 1
l'equazione è risolubile ed ha la seguente soluzione:
xoo = c*xo/MCD = -56113 * 5623 / 1 =-315523399
yoo = c*yo/MCD = -56113 * -29968 / 1 = 1681594384

I valori minimi di x e di y che soddisfano l'equazione:

(-39801 ) * x -( 7468 ) * y = -56113

sono: xom = 7069 , yom =-37667

_____ VERIFICA _____

A * xoo - B * yoo = (-39801 )*(-315523399 ) - ( 7468 )*( 1681594384 ) = -56113
xoo e yoo sono valori VALIDI

verifica con valori minimi xom e yom :
A * xom - B * yom = (-39801 )*( 7069 ) - ( 7468 )*(-37667 ) = -56113
xom e yom sono valori VALIDI
-----ALTRE SOLUZIONI-----
altre soluzioni di valore intorno a xom ed yom sono le seguenti:
x(-3)=-15335          y(-3)= 81736
x(-2)=-7867          y(-2)= 41935
x(-1)=-399          y(-1)= 2134
xom = 7069          yom =-37667
x( 1 )= 14537          y( 1 )=-77468
x( 2 )= 22005          y( 2 )=-117269
x( 3 )= 29473          y( 3 )=-157070
    
```

2 - Esempio relativo alla 2^a opzione

Siano disponibili i seguenti valori che sono tutti numeri primi (vedi Nota a pag.8):

$$e = 4259 \quad p = 91141 \quad q = 76913$$

Eseguendo il programma riportato nell'**Allegato I**, una volta introdotti da Input i

valori di **e** di **p** e di **q** viene calcolato il valore di Φ e quindi risolta l'equazione lineare $e \cdot d - \Phi \cdot y = 1$ si trova subito il valore $d = 4432327499$ quale **Chiave privata**.

Sullo schermo del monitor compare quanto segue:

```

----- Programma LCONGRSA.BAS -----
il presente Programma prevede tre opzioni:
1^ OPZIONE: RISOLUZIONE DELLA EQUAZIONE DIOFANTEA A * x - B * y = C
  se si vuole questa opzione battere un tasto qualsiasi e poi il tasto relativo alla cifra 1
2^ OPZIONE: CALCOLO DELLA CHAVE PRIVATA d NELL'ALGORITMO RSA
  se si vuole questa opzione battere un tasto qualsiasi e poi il tasto relativo alla cifra 2
3^ OPZIONE: ESEMPIO DI CALCOLO DELLA CHAVE PRIVATA d NELL'ALGORITMO RSA
  se si vuole questa opzione battere un tasto qualsiasi e poi il tasto relativo alla cifra 3

battere ora un tasto qualsiasi

INTRODURRE L'OPZIONE DESIDERATA e battere poi il tasto Invio : 2

ATTENZIONE !! ACCERTARSI CHE I VALORI DA INTRODURRE SIANO NUMERI PRIMI

introduci numero primo e: 4259
introduci numero primo p: 91141
introduci numero primo q: 76913

i due parametri della chiave pubblica nell' algoritmo RSA sono:
n = p * q = 7009927733   e = 4259
-----
il valore della funzione di Eulero f = (p-1)*(q-1) è: 7009759680

la chiave privata d nell' algoritmo RSA si calcola risolvendo
l'equazione lineare diofantea:

      4259 * x - 7009759680 * y = 1

si trova: MCD ( 4259, 7009759680 ) = 1

      xo = 4432327499   yo = 2693
con xo e yo soluzioni della suddetta equazione

la chiave privata cercata è: d = xo = 4432327499

in quanto si ha:

e*d - f*yo = 4259 * 4432327499 - 7009759680 * 2693
e*d - f*yo = 18877282818241 - 18877282818240 = 1

```

3 - Esempio relativo alla 3^a opzione

In questo esempio sono già noti e disponibili nel listato del programma i seguenti valori di e e di p e di q (vedi Nota a pag. 8) :

$$e = 8627$$

$$p = 749129$$

$$q = 94321 \quad (\text{tutti e tre numeri primi})$$

Sullo schermo del monitor compare quanto segue:

```

----- Programma LCONGRSA.BAS-----
il presente Programma prevede tre opzioni:
1^ OPZIONE: RISOLUZIONE DELLA EQUAZIONE DIOFANTEA  $A * x - B * y = C$ 
  se si vuole questa opzione battere un tasto qualsiasi e poi il tasto relativo alla cifra 1
2^ OPZIONE: CALCOLO DELLA CHAVE PRIVATA d NELL'ALGORITMO RSA
  se si vuole questa opzione battere un tasto qualsiasi e poi il tasto relativo alla cifra 2
3^ OPZIONE:ESEMPIO DI CALCOLO DELLA CHAVE PRIVATA d NELL'ALGORITMO RSA
  se si vuole questa opzione battere un tasto qualsiasi e poi il tasto relativo alla cifra 3

battere ora un tasto qualsiasi

INTRODURRE L'OPZIONE DESIDERATA e battere poi il tasto Invio :3

i 3 numeri primi seguenti sono già posti nel listato del programma

e = 8627  p = 749129  q = 94321

funzione di Eulero:  $f = (p-1)*(q-1) = 70657752960$ 

la chiave privata d nell'algorithmo RSA si calcola risolvendo
l'equazione lineare diofantea:

 $(8627) * x - (70657752960) * y = 1$ 

si trova:  $MCD(8627, 70657752960) = 1$ 

   $x_0 = 11843179643$    $y_0 = 1446$ 
con  $x_0$  ed  $y_0$  soluzioni della suddetta equazione

la chiave privata cercata è:  $d = x_0 = 11843179643$ 

in quanto si ha:

 $e*d - f*y_0 = 8627 * 11843179643 - 70657752960 * 1446$ 
 $e*d - f*y_0 = 102171110780161 - 102171110780160 = 1$ 

```

Allegato 2

```

REM ----- PROGRAMMA "INVERZZ.BAS " -----

REM Il programma è dedicato al calcolo di x ed y interi soddisfacenti la
REM equazione diofantea  $e*x - F*y = 1$ , ovvero la congruenza:  $e*x = 1 \pmod{F}$ 
REM con  $e =$  parametro costituente insieme a  $n = p * q$  la chiave
REM pubblica nell' algoritmo crittografico RSA ed  $F$  è la Funzione
REM di Eulero definita da  $F = (p-1)*(q-1)$ , dove  $e, p, q$  sono
REM tre numeri primi grandi random.
REM Il valore intero di x trovato risulta essere l'inverso
REM moltiplicativo di e, quindi è la chiave privata  $d = x = e^{-1} \pmod{F}$ 
REM nell'RSA.
REM I valori di  $e$  di  $p$  e di  $q$  possono essere anche formati anche
REM da DECINE o CENTINAIA di cifre; dovranno pertanto essere
REM trovati previamente con apposito programma.
REM battere un tasto qualsiasi
CLS
REM -----
PRINT
PRINT "===== RISOLUZIONE DELL'EQUAZIONE  $e * x - F * y = 1$  =====";
PRINT "======"
PRINT : PRINT "(e = CHIAVE PUBBLICA; F = FUNZIONE DI EULERO);";
PRINT " d = x CHIAVE PRIVATA nell'RSA)"
PRINT : PRINT "Sono previste due opzioni": PRINT
PRINT "la PRIMA OPZIONE è relativa ad introdurre come INPUT tre numeri";
PRINT " primi random : e, p, q": PRINT
PRINT "la SECONDA OPZIONE è relativa ad un ESEMPIO di calcolo con e, p, q";
PRINT " già inseriti"
PRINT "-----"
PRINT "Se si vuole la prima opzione battere prima su un tasto qualsiasi";
PRINT "e poi sul tasto relativo alla cifra 1": PRINT
PRINT "Se si vuole la seconda opzione battere su prima un tasto qualsiasi";
PRINT "e poi sul tasto relativo alla cifra 2"

DO: y$ = INKEY$: LOOP WHILE y$ = ""
CLS : DEFDBL A-Z: PRINT

DIM ap(200), aq(200)
DIM a(200), a1(200), b(200), b1(200), r(200), q(200)
DIM pt(200), p1(200), p0(200), qt(200), q1(200), q0(200)
DIM s(500), s1(500), s2(500)
p0(0) = 0: q0(0) = 1: p1(0) = 1: q1(0) = 0
g = 6: pr = 10 ^ g
oz: INPUT "quale opzione"; oz$
IF oz$ = "1" GOTO iniz
IF oz$ = "2" GOTO tempo1
PRINT " ribatti": GOTO oz
iniz: PRINT "-----ATTENZIONE !!!!!!!-----"

```

```

PRINT "I TRE NUMERI DA INTRODURRE DEVONO ESSERE PRIMI"
PRINT: INPUT "introdurre il parametro e:", dn$: ln = LEN(dn$):
PRINT "cifre di e: "; ln
PRINT
INPUT "introdurre il numero primo p:", dp$: lp = LEN(dp$)
PRINT "cifre di p: "; lp
PRINT
INPUT "introdurre il numero primo q:", dq$: lq = LEN(dq$)
PRINT "cifre di q: "; lq
tempo1: t1 = TIMER: ' g = 5: pr = 10 ^ g
IF oz$ = "1" GOTO 111
333 REM ===== esempio =====

dn1$= "76510020102356880421350797640553701976467646374464641220078978722"
dn2$ = "888897102002419"
dn$ = dn1$ + dn2$
ld = LEN(dn$): ' PRINT dn$:
PRINT "e = " + dn$, "cifre di e: "; ld

dp1$ = "4900231657891653479356441986527365947689785649867625649716597289"
dp2$ = "1010201235656555564373567897346132043725373865670002344537031024"
dp3$ = "43730010255997115760932480123703123"
dp$ = dp1$ + dp2$ + dp3$
lp = LEN(dp$):
PRINT "p = " + dp$, "cifre di p: "; lp

dq1$= "71643001020050405287995646528007777798442222643500002132652407337"
dq2$= "13983429277861982546792312402466508770707088985552467921534565571"
dq3$ = "240326478700269"
dq$ = dq1$ + dq2$ + dq3$
lq = LEN(dq$):
PRINT "q = " + dq$, "cifre di q: "; lq
111 REM ----- MESSA in FORMA DI e -----
ln = LEN(dn$) + 1:
cn = INT((ln - 1) / g): IF cn = (ln - 1) / g THEN cn = cn - 1
cr = INT((lr - 1) / g): IF cr = (lr - 1) / g THEN cr = cr - 1
IF cn = 0 THEN a(0) = VAL(dn$): PRINT a(0): ' GOTO divis
FOR k = 0 TO cn - 1: c = ln - (k + 1) * g: a$ = MID$(dn$, c, g)
a(k) = VAL(a$)
NEXT k
IF c > 1 THEN a$ = LEFT$(dn$, c - 1): a(cn) = VAL(a$)
REM ----- messa in forma del numero primo p -----
lp = LEN(dp$) + 1: ' PRINT "cifre di p: "; lp - 1
cp = INT((lp - 1) / g): IF cp = (lp - 1) / g THEN cp = cp - 1
IF cp = 0 THEN ap(0) = VAL(dp$): PRINT ap(0): GOTO divp
FOR k = 0 TO cp - 1: c = lp - (k + 1) * g: a$ = MID$(dp$, c, g)
ap(k) = VAL(a$)
NEXT k
IF c > 1 THEN a$ = LEFT$(dp$, c - 1): ap(cp) = VAL(a$):

```

```

divp: ' PRINT "p ="; ap(cp);
'FOR k = cp - 1 TO 0 STEP -1: PRINT ap(k); : NEXT k
'PRINT "p = " + dp$, "cifre di p:"; lp - 1
REM ----- messa in forma del numero primo q -----
lq = LEN(dq$) + 1: ' PRINT "cifre di q:"; lq - 1
cq = INT((lq - 1) / g): IF cq = (lq - 1) / g THEN cq = cq - 1
IF cq = 0 THEN aq(0) = VAL(dq$): PRINT aq(0): GOTO divq
FOR k = 0 TO cq - 1: c = lq - (k + 1) * g: a$ = MID$(dq$, c, g)
aq(k) = VAL(a$)
NEXT k
IF c > 1 THEN a$ = LEFT$(dq$, c - 1): aq(cq) = VAL(a$):
divq:
'PRINT "q = " + dq$, "cifre di q:"; lq - 1
REM ----- calcolo della funzione di EULERO : F=(p-1)*(q-1)-----
ap(0) = ap(0) - 1: aq(0) = aq(0) - 1
w = cp + cq + 1:
FOR k = 0 TO cq: r = 0
  FOR h = 0 TO cp
    x = b(h + k) + ap(h) * aq(k) + r: r = INT(x / pr): b(h + k) = x - r * pr
  NEXT h
  b(h + k) = r
NEXT k
IF b(w) = 0 THEN w = w - 1
PRINT "-----";
PRINT "-----"
PRINT "Funzione di Eulero:";
PRINT " F = (p-1)*(q-1) "; : ' FOR i = w TO 0 STEP -1: PRINT b(i); : NEXT i
F$ = STR$(b(w)): fw = LEN(F$) - 1: cf = w * g + fw
PRINT : F$ = " ": ab$ = STR$(b(w)): lx = LEN(ab$)
IF g = 7 THEN w$ = " "
PRINT "F   = "; SPC(g - lx); ab$; SPC(1);
FOR k = w - 1 TO 0 STEP -1
  x$ = "": c$ = STR$(b(k)): la = LEN(c$) - 1: r$ = RIGHT$(c$, la)
  s = g - la: IF s = 0 THEN b$ = x$ + r$: GOTO X99
  z$ = "0": x$ = STRING$(s, z$): b$ = x$ + r$
X99: PRINT b$; SPC(1);
NEXT k
PRINT : PRINT "=====";
PRINT
"-----"
cibi: cb = w
REM n^ celle del DIVIDENDO a(k) : cn
REM n^ celle del DIVISORE b(h) : cb
cn1 = cn: cb1 = cb: i = 0
FOR k = 0 TO cn1: a1(k) = a(k): NEXT k
FOR h = 0 TO cb1: b1(h) = b(h): NEXT h

REM ----- DIVISIONE -----
REM PRINT " i", " r", " qz", " p", " q"

```

```

REM -----INIZIO CICLO per il calcolo del M.C.D. -----
mcd: i = i + 1: 'PRINT i,
ERASE q: cc = 0
FOR k = cn TO 0 STEP -1: r(k) = a(k): NEXT k
  d = b(cb) + 1: ' PRINT "b(cb)="; b(cb), "cb="; cb
FOR h = cn TO cb + 1 STEP -1: a = r(h): sv = cb
  IF a < d THEN a = r(h) * pr + r(h - 1): sv = cb + 1
  q = INT(a / d): q(h - sv) = q(h - sv) + q
  FOR k = 0 TO cb
    x = q * b(k) + r: r = INT(x / pr): z = x - r * pr
    s = k + h - sv
    IF z > r(s) THEN r(s) = pr + r(s): r(s + 1) = r(s + 1) - 1
    r(s) = r(s) - z
  NEXT k
  r(h) = r(h) - r
  r = 0
  IF r(h) <> 0 THEN h = h + 1
NEXT h
REM ----- controllo sul resto r della divisione -----
REM ----- r puo' essere ancora > di b : si va a "continua"
REM ----- se r < b, r risulta gia' essere l'effettivo resto: si va a "resto"
trovar: FOR j = cb TO 0 STEP -1
  IF r(j) < b(j) GOTO resto
  IF r(j) > b(j) GOTO continua
NEXT j
REM RESTO = 0
GOTO gcd
continua: r = 0: d = b(cb) + 1
  IF r(cb) = b(cb) THEN d = b(cb)
  q = INT(r(cb) / d): q(0) = q(0) + q
FOR j = 0 TO cb
  x = q * b(j) + r: r = INT(x / pr): z = x - r * pr
  IF z > r(j) THEN r(j) = pr + r(j): r(j + 1) = r(j + 1) - 1
  r(j) = r(j) - z
NEXT j
IF r(cb) > 0 GOTO trovar
resto: FOR j = cb TO 0 STEP -1: IF r(j) > 0 THEN cr = j: j = 0: GOTO stampar
NEXT j
stampar: REM PRINT "RESTO: "; : FOR k = cr TO 0 STEP -1: PRINT r(k); : NEXT k
cc = ABS(cn - cb): IF q(cc) = 0 THEN cc = cc - 1
rs = 0
REM ----- CALCOLO FINALE del quoziente -----
FOR k = 0 TO cc: x = q(k) + rs: rs = INT(x / pr): q(k) = x - rs * pr: NEXT k
IF rs > 0 THEN cc = cc + 1: q(cc) = rs
REM -----
PRINT "=====
'DO: y$ = INKEY$: LOOP WHILE y$ = ""

```

```

facile: REM -----
REM ----- calcolo di  $pt(k) = q(k) * p1(k) + p0(k)$  -----
REM ----- calcolo di  $s(j) = q(k) * p1(k)$  -----
  FOR k = 0 TO cp1: r = 0
  FOR h = 0 TO cc
     $x = s(h + k) + q(h) * p1(k) + r$ :  $r = INT(x / pr)$ :  $s(h + k) = x - r * pr$ 
  NEXT h
   $s(h + k) = r$ 
NEXT k
cs = cc + cp1 + 1
FOR h = cs TO 0 STEP -1
  IF  $s(h) <> 0$  THEN cpt = h: GOTO piti
NEXT h

piti: REM ----- calcolo di  $pt(j) = s(j) + p0(j)$  -----
IF cpt < cp0 THEN cpt = cp0
r = 0
FOR j = 0 TO cpt:
   $x = s(j) + p0(j) + r$ :  $r = INT(x / pr)$ :  $pt(j) = x - r * pr$ 
NEXT j
IF r > 0 THEN cpt = cpt + 1:  $pt(cpt) = r$ 

ERASE s: : cs = 0: r = 0
'DO: y$ = INKEY$: LOOP WHILE y$ = ""
REM -----
REM ----- calcolo di  $qt(k) = q(k) * q1(k) + q0(k)$ 
REM ----- calcolo di  $s(j) = q(k) * q1(k)$  -----
  FOR k = 0 TO cq1: r = 0
  FOR h = 0 TO cc
     $x = s(h + k) + q(h) * q1(k) + r$ :  $r = INT(x / pr)$ :  $s(h + k) = x - r * pr$ 
  NEXT h
   $s(h + k) = r$ 
NEXT k
cs = cc + cq1 + 1: ' PRINT "cs="; cs
FOR h = cs TO 0 STEP -1
  IF  $s(h) <> 0$  THEN cq1 = h: GOTO ciquiti
NEXT h

REM ----- calcolo di  $qt(j) = s(j) + q0(j)$  -----
ciquiti: ' cq1 = cs:
IF cq1 < cq0 THEN cq1 = cq0
r = 0
FOR j = 0 TO cq1
   $x = s(j) + q0(j) + r$ :  $r = INT(x / pr)$ :  $qt(j) = x - r * pr$ 
NEXT j
IF r > 0 THEN cq1 = cq1 + 1:  $qt(cq1) = r$ 
ERASE s: cs = 0: r = 0
'DO: y$ = INKEY$: LOOP WHILE y$ = ""
IF fl = 1 GOTO term
REM ----- sostituzione di  $a(h) <--- b(h)$  e di  $b(h) <--- r(h)$  -----

```

```

FOR h = 0 TO cb: a(h) = b(h): NEXT h
FOR h = 0 TO cb: b(h) = r(h): NEXT h
cn = cb: cb = cr
REM ---- sostituzione: p0(k)<--- p1(k), p1(k)<--- pt(k) -----
FOR k = 0 TO cp1: p0(k) = p1(k): NEXT k
ERASE p1
FOR k = 0 TO cpt: p1(k) = pt(k): NEXT k
cp0 = cp1: cp1 = cpt
REM ---- sostituzione: q0(h)<--- q1(h), q1(h)<--- qt(h) -----
FOR h = 0 TO cq1: q0(h) = q1(h): NEXT h
ERASE q1
FOR k = 0 TO cq1: q1(k) = qt(k): NEXT k
cq0 = cq1: cq1 = cq1
GOTO mcd
gcd: 'DO: y$ = INKEY$: LOOP WHILE y$ = ""

PRINT "m.c.d.(e,F) ="; : FOR k = cb TO 0 STEP -1: PRINT b(k); : NEXT k
fl = 0
IF cb > 0 THEN PRINT "non essendo MCD(e,F)=1 il calcolo";
IF cb > 0 THEN PRINT "della chiave d è IMPOSSIBILE": GOTO tempo
IF b(0) <> 1 THEN PRINT "IMPOSSIBILE": GOTO tempo
PRINT "      numero di iterazioni :"; i
IF i = INT(i / 2) * 2 GOTO term
'IF cc = 0 AND q(0) = 1 THEN PRINT "cc="; cc, "q(0)="; q(0);: GOTO 100
REM ----- numero di iterazioni i è dispari e q(0)>1 (a>1) -----
FOR k = cc TO 0 STEP -1
  'IF q(k) > 0 THEN PRINT "q>1", "q(0)="; q(0):
NEXT k
REM -----calcolo di q(k)=q(k)-1 -----
'PRINT : PRINT "pt:"; : FOR k = cpt TO 0 STEP -1: PRINT pt(k); : NEXT k
'PRINT : PRINT "qt:"; : FOR k = cq1 TO 0 STEP -1: PRINT qt(k); : NEXT k
fl = 1: PRINT : ' PRINT "fl"; fl
GOTO facile
REM -----
100 REM -----numero di iterazioni i è dispari e q(0)=1 (a=1) -----
REM -----
term: PRINT : F$ = "": ab$ = STR$(qt(cq1)): lx = LEN(ab$)
IF g = 7 THEN w$ = " "
PRINT "d = " + w$; SPC(g - lx); ab$; SPC(1);
'PRINT "e * x = " ; SPC(g - lx); ab$; SPC(1);
FOR k = cq1 - 1 TO 0 STEP -1
  x$ = "": c$ = STR$(qt(k)): la = LEN(c$) - 1: r$ = RIGHT$(c$, la)
  s = g - la: IF s = 0 THEN b$ = x$ + r$: GOTO X91
  z$ = "0": x$ = STRING$(s, z$): b$ = x$ + r$
X91: PRINT b$; SPC(1);
NEXT k
PRINT
xm = qt(cq1): cx = LEN(STR$(xm)) - 1 + g * (cq1)
PRINT: PRINT "il numero d Š l'inverso moltiplicativo di e (mod F)";

```

```

PRINT "tale numero pu• considerarsi CHIAVE PRIVATA nell'algorithmo RSA se i ";
PRINT "numeri p , q"; "risultano essere effettivamente primi;"
PRINT "d è composto da"; cx; "cifre"
IF pt(cpt) = 0 THEN cpt = cpt - 1
PRINT
Print "----- VERIFICA -----"
ERASE s: cs = 0
FOR k = 0 TO cqt: r = 0
  FOR h = 0 TO cn1
    x = s(h + k) + a1(h) * qt(k) + r: r = INT(x / pr): s(h + k) = x - r * pr
  NEXT h
s(h + k) = r
NEXT k
w = cn1 + cqt + 1
IF s(w) = 0 THEN w = w - 1
FOR i = w TO 0 STEP -1: s1(i) = s(i):
NEXT i
PRINT : F$ = "": ab$ = STR$(s1(w)): lx = LEN(ab$)
IF g = 7 THEN w$ = " "
PRINT "e * d =" + w$: SPC(g - lx); ab$: SPC(1);
FOR k = w - 1 TO 0 STEP -1
  x$ = "": c$ = STR$(s1(k)): la = LEN(c$) - 1: r$ = RIGHT$(c$, la)
  s = g - la: IF s = 0 THEN b$ = x$ + r$: GOTO X90
  z$ = "0": x$ = STRING$(s, z$): b$ = x$ + r$
X90: PRINT b$: SPC(1);
NEXT k
ERASE s: cs = 0
FOR k = 0 TO cpt: r = 0
  FOR h = 0 TO cb1
    x = s(h + k) + b1(h) * pt(k) + r: r = INT(x / pr): s(h + k) = x - r * pr
  NEXT h
s(h + k) = r
NEXT k
w = cb1 + cpt + 1
IF s(w) = 0 THEN w = w - 1
FOR i = w TO 0 STEP -1: s2(i) = s(i): NEXT i
PRINT : F$ = "": ab$ = STR$(s2(w)): lx = LEN(ab$)
IF g = 7 THEN w$ = " "
PRINT : PRINT "F * y =" + w$: SPC(g - lx); ab$: SPC(1);
FOR k = w - 1 TO 0 STEP -1
  x$ = "": c$ = STR$(s2(k)): la = LEN(c$) - 1: r$ = RIGHT$(c$, la)
  s = g - la: IF s = 0 THEN b$ = x$ + r$: GOTO X95
  z$ = "0": x$ = STRING$(s, z$): b$ = x$ + r$
X95: PRINT b$: SPC(1);
NEXT k
PRINT:REM ----- algoritmo di sottrazione: d(k) = na(k) - nb(h) -----
DIM d(100)
r = 0: w = g1
FOR k = 0 TO w:

```

```
d(k) = s1(k) - s2(k) - r: r = 0
IF d(k) < 0 THEN d(k) = pr + d(k): r = 1
NEXT k
FOR k = w TO 0 STEP -1: IF d(k) > 0 THEN k1 = k: k = 0
NEXT k
PRINT : PRINT "e * d - F * y =";
FOR h = k1 TO 0 STEP -1: PRINT d(h); : NEXT h
tempo: t2 = TIMER - t1: : PRINT "      tempo impiegato:"; t2; "secondi"
END
```

Esempi di risoluzione tramite il programma in Qbasic riportato in Allegato 2

Si riporta per la prima opzione tre esempi di calcolo mostrando ciò che compare sullo schermo del monitor dopo aver fatto partire il programma ed aver inserito gli eventuali dati richiesti riguardanti l'opzione scelta.

1° esempio relativo alla prima opzione (dati introdotti da input)

In questo esempio si osservi che non è possibile effettuare il calcolo della chiave privata d essendo $\text{MCD}(e,F) \neq 1$

```
===== RISOLUZIONE DELL'EQUAZIONE e * x - F * y = 1 =====
(e = CHIAVE PUBBLICA; F = FUNZIONE DI EULERO); d = x CHIAVE PRIVATA nell'RSA)
Sono previste due opzioni:
la PRIMA OPZIONE è relativa ad introdurre come INPUT tre numeri primi: e, p, q
la SECONDA OPZIONE è relativa ad un ESEMPIO di calcolo con e, p, q già inseriti
-----
Se si vuole la prima opzione battere prima su un tasto qualsiasi
e poi sul tasto relativo alla cifra 1

Se si vuole la seconda opzione battere prima su un tasto qualsiasi
e poi sul tasto relativo alla cifra 2

quale opzione? 1
introdurre il parametro e:34345664764764564635345455557
cifre di e: 29
introdurre il numero primo p:445364774647576346476345636463335777
cifre di p: 36
introdurre il numero primo q:3342536463543535554111
cifre di q: 22
-----
Funzione di Eulero: F = (p-1)*(q-1)
F = 1488 647998 837373 502242 652546 320192 065415 380119 707146 839360
=====
MCD(e,F) = 57 ; non essendo MCD(e,F) = 1 il calcolo della chiave d è IMPOSSIBILE
tempo impiegato: 0 secondi
```

2° esempio relativo alla prima opzione (dati introdotti da input)

Siano noti i seguenti valori (vedi Nota a pag. 8):

e: 82132240553708956102000023104321022310483

e è un primo di 41 cifre

p:5641349579891310252345676789492265610000570123746761347376644766676677102001220
100012342044599986754375491119527382222310519597356425008731040026520870011235561
p è un primo di 160

q: 4992890145577779494113201024761641002021230405513312050410123411002435674646789
9110205646789821050746124545449000012457877778744100222354444559445107

q è un primo di 149 cifre

Prima di effettuare il calcolo della chiave privata d si è appurato con un programma ad hoc che i

valori numerici random dei due primi p e q introdotti fossero tali da avere $\text{MCD}\left(\frac{p-1}{2}, \frac{q-1}{2}\right) = 1$

come suggerito in [FeL].

Eseguito il programma si ottiene sul monitor in due successive stampe il seguente risultato:

3° esempio relativo alla terza opzione

In questo esempio sono già noti e disponibili nel listato del programma i seguenti valori di e e di p e di q (vedi Nota a pag. 8), tutti e tre numeri primi. Eseguendo il programma riportato in Allegato 2 con la seconda opzione, si ottengono sul monitor in due successive stampe il seguente risultato:

PRIMA STAMPA

===== RISOLUZIONE DELL'EQUAZIONE $e * x - F * y = 1$ =====

(e = CHIAVE PUBBLICA; F = FUNZIONE DI EULERO); $d = x$ CHIAVE PRIVATA nell'RSA)

Sono previste due opzioni:

la PRIMA OPZIONE è relativa ad introdurre come INPUT tre numeri primi: e , p , q

la SECONDA OPZIONE è relativa ad un ESEMPIO di calcolo con e , p , q già inseriti

Se si vuole la prima opzione battere prima su un tasto qualsiasi
e poi sul tasto relativo alla cifra 1

Se si vuole la seconda opzione battere prima su un tasto qualsiasi
e poi sul tasto relativo alla cifra 2

SECONDA STAMPA

quale opzione? 2

$e = 7651002010235688042135079764055370197646764637446464122007897872288889710200$
2419 cifre di e : 80

$p = 4900231657891653479356441986527365947689785649867625649716597289101020123565$
65555643735678973461320437253738656700023445370310244373001025599711576093248012
3703123 cifre di p : 163

$q = 7164300102005040528799564652800777779844222264350000213265240733713983429277$
86198254679231240246650877070708898555246792153456557124032647870026501
cifre di q : 147

Funzione di Eulero: $F = (p-1)*(q-1)$

$F = 3510 673016 648150 188589 815354 294960 364829 379828 934780 692675$
649149 203794 312596 073682 418595 512607 278069 436264 505463 086559 110590
147065 663138 994246 110444 204434 311258 008903 741150 656785 240207 672273
745792 509548 176639 382493 416426 07070 4156153 758510 874259 168647 811905
779005 073923 320948 820322 188525 867662 082168 928272 733000

=====

MCD(e,F) = 1 numero di iterazioni : 173

$d = 1770 542938 088332 691539 915756 552985 275568 854717 064335 156984$
408898 085423 222951 630074 328529 348537 935353 104853 843953 635749 308032
508416 287076 845932 392184 500239 609714 826431 558495 730207 187061 990150
890257 169651 990161 646303 783211 748903 489225 944687 293544 458564 286659
960581 258884 791699 933449 498292 757670 601000 984425 570179

il numero d è l'inverso moltiplicativo di e (mod F)

tale numero può considerarsi CHIAVE PRIVATA nell'algorithmo RSA se i numeri p , q

risultano essere effettivamente primi;

d è composto da 310 cifre

----- VERIFICA -----

$e * d = 135464 275785 224347 791147 363260 580676 279353 640654 541253 286817$
317765 494145 485869 697968 285019 328449 706388 645440 419837 471587 352852
918406 366907 008880 656765 171296 276766 894916 940723 120588 804140 997017
134006 594205 450206 344836 772849 040402 115721 141423 622085 813399 036042
358851 264885 276440 176757 301120 527366 275228 084358 253866 310978 271398
701813 299939 411844 871647 414268 158158 722371 018951 526192 046712 263001

$F * y = 135464 275785 224347 791147 363260 580676 279353 640654 541253 286817$
317765 494145 485869 697968 285019 328449 706388 645440 419837 471587 352852
918406 366907 008880 656765 171296 276766 894916 940723 120588 804140 997017
134006 594205 450206 344836 772849 040402 115721 141423 622085 813399 036042
358851 264885 276440 176757 301120 527366 275228 084358 253866 310978 271398
701813 299939 411844 871647 414268 158158 722371 018951 526192 046712 263000

$e * d - F * y = 1$ tempo impiegato: .05 secondi



Cristiano Teodoro, si è laureato in Ingegneria Elettronica presso l'Università "La Sapienza" di Roma. Ha percorso la sua carriera professionale presso l'Istituto Superiore delle Poste e delle Telecomunicazioni (ISPT) e quindi presso l'attuale ISCTI. Ha svolto incarichi in diversi campi ed attività delle telecomunicazioni fra cui in particolare quelli di normazione, di omologazione e di collaudo di apparati terminali telegrafici (telescriventi) telefonici e dati (modem), di apparati di multiplexione e trasmissione a lunga distanza su portante fisico di segnali Telefonici e Dati relativi alla multiplexione PCM, alla Gerarchia Plesiocrona (PDH) e alla gerarchia sincrona (SDH).

Ha svolto inoltre una intensa attività Normativa e di Standardizzazione sia in ambito Nazionale che Internazionale quale Relatore Nazionale di due Commissioni dell'UIT - T e come membro di diverse Commissioni dello stesso Organismo Internazionale.

E' stato docente della materia "Esercitazioni di Telegrafia e Telefonia" presso e la Scuola Superiore di Specializzazione in Telecomunicazioni (SSST) dell'ISCTI.

cristianoteodoro@virgilio.it