Alma Mater Studiorum · Università di Bologna

Scuola di Scienze

Corso di Laurea in Matematica

Attacchi a RSA e reticoli

Tesi di Laurea in Algoritmi di Teoria dei Numeri e Crittografia

Relatore:
Chiar.mo Prof.
Davide Aliffi

Presentata da:
Riccardo Palmieri

II Sessione
Anno Accademico 2012-2013

Lunga vita e prosperità.

Indice

Capitolo 1

Computabilità,Complessità e Intrattabilità

Da un punto di vista moderno i problemi matematici si suddividono in due grandi categorie, i problemi risolvibili e quelli non risolvibili. I primi si suddividono in altre due sottocategorie: i problemi trattabili e i problemi intrattabili.
I problemi trattabili sono quelli a cui si può rispondere in un tempo polinomiale con una macchina di Turing deterministica, che può oggi essere rappresentata anche da un normale calcolatore; i problemi intrattabili sono chiaramente l'opposto di questi ultimi.
Queste ultime due sottocategorie non sono ancora separate chiaramente; è ancora non risolto il problema di determinare se le due classi di problemi, $\mathcal{P}$ e $\mathcal{NP}$ coincidono.

1.1 Problemi intrattabili di teoria dei numeri

1.1.1 Fattorizzazione (IFP e PFP)

Il problema della fattorizzazione intera (IFP) può essere facilmente definito come segue:

Definizione 1.1. Dato un numero positivo N composto maggiore di 1, cercare un fattore proprio f con $1 < f < N$.

$$IFP : \left\{N \in \mathbb{Z}^+_{>1}\right\} \longrightarrow \{f|N \ \ con \ \ 1 < f < N\}$$

Questo problema è ritenuto a tutt'oggi difficile da risolvere, un altro problema strettamente vicino a questo è il PFP, cioè il problema della fattorizzazione in primi.

Definizione 1.2. Dato un numero positivo N maggiore di 1, trovare i fattori primi di N.

$$PFP : \left\{\mathbb{Z}^+ \setminus 0\right\} \longrightarrow \left\{\mathbb{Z}^+ \setminus 0\right\}$$

Il teorema fondamentale dell'aritmetica ci garantisce l'esistenza e l'unicità di questa fattorizzazione, interessante però è il fatto che nella prima dimostrazione di questo teorema, fatta da Euclide, non ci sia alcun algoritmo efficiente per trovarla. Ovviamente se ci fosse un algoritmo per risolvere la PFP avremmo di conseguenza anche risolto l'altro problema, cioè l'IFP, infatti questi due problemi si possono considerare computazionalmente equivalenti:

$$IFP \overset{\mathcal{P}}{\Longleftrightarrow} PFP$$

Come già è stato detto non si conoscono algoritmi efficienti di fattorizzazione, il più performante è l'NFS (Number Field Sieve) che ha una complessità di $\mathcal{O}(exp(c(\log N)^{1/3}(\log\log n)^{2/3}))$ dove $c = (64/9)^{1/3}$.

1.1.2 Logaritmo discreto (DLP)

Definizione 1.3. Siano $x, y, N, k \in \mathbb{Z}^+$. Il DLP per il gruppo moltiplicativo $\mathbb{Z}^*_N$ può essere definito come segue:

$$DLP : \left\{n \in \mathbb{Z}^+_{>1}, x \in \mathbb{Z}^+, y \equiv x^k \ (mod \ N)\right\} \longrightarrow \{k\}$$

Il DLP si suppone abbia lo stesso grado di difficoltà dell'IFP, in quanto i metodi per quest'ultimo sono applicabili anche per il logaritmo discreto:

$$IFP \overset{\mathcal{P}}{\Longrightarrow} DLP$$

1.1.3 Vettore di norma minima (SVP)

Consiste nel dover trovare il vettore di norma minima in un reticolo, che può essere visto come sottogruppo di $\mathbb{R}^n$, e sarà ampiamente discusso nel terzo capitolo.

Capitolo 2

Crittografia ed RSA

2.1 Introduzione alla crittografia

La crittografia consiste nello studio di metodi per criptare un messaggio in modo tale che solo gli interessati possano decriptare e leggere il contenuto di questo.

$$Crittografia := Cifratura \oplus Decifrazione$$

Ovviamente parallelamente a questa disciplina si è sviluppata la criptoanalisi, cioè lo studio dei vari attacchi sui metodi di criptaggio, con il termine crittologia si intende l'insieme di queste ultime due.

Per facilitare l'esposizione si utilizzerà un linguaggio convenzionale: il mittente del messaggio sarà Bob, il ricevente Alice e il nemico, cioè colui che vuole intercettare sarà Eva.

Lo scopo della moderna crittografia è quello di trovare metodi matematici per garantire quattro proprietà della comunicazione tra Alice e Bob:

1. Confidenzialità e privacy: per impedire ad Eva di comprendere il messaggio.

2. Integrità: per impedire ad Eva di modificare il messaggio.

3. Autentificazione del mittente: per garantire il mittente del messaggio.

4. Non Ripudiabilità: per evitare che Bob, in un secondo momento, non riconosca il messaggio.

Definizione 2.1. Un sistema crittografico convenzionale S è formalmente definito come segue:

$$S = (M, C, K, E, D)$$

dove:

1. M è lo spazio dei testi in chiaro.

2. C è lo spazio dei testi cifrati.

3. K è lo spazio delle chiavi.

4. E è la funzione di cifratura.

5. D è la funzione di decifrazione.

quindi

$$E_K : M \to C$$

e

$$D_K : C \to M$$

queste ultime due devono soddisfare la seguente relazione:

$$D_k(c) = D_k(E_k(m)) = m, \ \forall m \in M, c \in C$$

La sicurezza di un sistema crittografico è ovviamnete importante, proprio per questo sono stati individuati diversi livelli di sicurezza:

1. *Incondizionatamente sicuro*: un sistema è incondizionatamente sicuro se un cirttoanalista non può determinare il testo in chiaro indipendentemente dal numero di testi cifrati di cui è in possesso, dal tempo e dalle risorse di calcolo che ha a disposizione.

2. *Computazionalmente sicuro*: un sistema è computazionalmente sicuro se il crittoanalista non può decifrare il testo in un tempo polinomiale; questo particolare tipo di sicurezza si divide in:

   (a) Sicurezza provata: si dimostra che il criptosistema è essenzialmente difficile tanto quanto un problema matematico conosciuto e supposto difficile, come l'IFP e il DLP.

   (b) Sicurezza pratica/congetturata: il criptosistema si *suppone* difficile come un problema matematico difficile come l'IFP o il DLP.

Ci sono anche diversi tipi di attacchi differenti:

1. Attacco chiphertext-only
2. Attacco Known-plaintext
3. Attacco Chosen-plaintext
4. Attacco Choesn-ciphertext

2.2 Crittografia a chiave pubblica ed RSA

2.2.1 Crittografia a chiave pubblica

Un sistema crittografico a chiave pubblica, definito anche asimmetrico, funziona in modo analogo a quello simmetrico, con la differenza che sono necessarie due chiavi, una pubblica e_k per criptare il messaggio e una privata d_k per decriptare.
Affinchè questi sistemi funzionino è necessario di una funzione univoca con trapdoor:

Definizione 2.2. Siano S e T due insiemi finiti. Una funzione unidirezionale

$$f : S \to T$$

è una funzione invertibile che soddisfa le seguenti proprietà:

1. f è facile da calcolare, cioè, dato $x \in S$, $y = f(x)$ è facilmente ottenibile.
2. f^{-1} è difficile da calcolare: dato $y \in T$, non si conoscono algoritmi efficienti per calcolare f^y.
3. f^{-1} è facilmente calcolabile quando si è a conoscenza di una trapdoor, che è una informazione segreta associata alla funzione.

2.2.2 RSA

Definizione 2.3. Il sistema di crittografia a chiave pubblica RSA può essere definito come segue:

$$RSA = (M, C, K, e, d, N, E, D)$$

dove:

1. M è lo spazio dei messaggi in chiaro.
2. C è lo spazio dei messaggi cifrati.
3. K è lo spazio delle chiavi.
4. $N = pq$ è il modulo con p,q primi, solitamente questi ultimi hanno almeno 100 cifre.
5. $\{e, N\}, \{d, N\} \in K$ con $e \neq d$ sono le chiavi di crifratura e decifrazione, che soddisfano rispettivamente

   $$ed \equiv 1 \pmod{\phi(N)}$$

   dove $\phi(N) = (p-1)(q-1)$ è la funzione di Eulero,
   definita da $\phi(N) = \#(\mathbb{Z}_N^*)$, che è quindi il numero degli elementi invertibili nel gruppo moltiplicativo $\mathbb{Z}_N^*$
6. E è la funzione di criptaggio:

   $$E_{e,N} : M \to C$$

che opera nel seguente modo:

$$c \equiv m^e \pmod{N}$$

7. D è la funzione di decriptaggio:

$$D_{d,N} : C \to M$$

che opera nel seguente modo:

$$m \equiv c^d \equiv (m^e)^d \pmod{N}$$

Teorema 2.1 (Correttezza di RSA)**.** *Siano m,c,N,e,d come sopra, allora:*

$$(m^e)^d \equiv m \pmod{N}$$

Dimostrazione.

$$\begin{aligned} c^d &\equiv (m^e)^d (mod N) \\ &\equiv m^{1+k\phi(N)} (mod N) \\ &\equiv m \cdot m^{k\phi(N)} (mod N) \\ &\equiv m \cdot (m^{\phi(N)})^k (mod N) \\ &\equiv m \cdot (1)^k (mod N) \\ &\equiv m \end{aligned}$$

□

2.2.3 La congettura dell'RSA

Data la chiave pubblica (e, N) e il testo cifrato c, si suppone che trovare il messaggi o in chiaro m sia difficile quanto fattorizzare N:

$$IFP(N) \overset{\mathcal{P}}{\Longleftrightarrow} RSA(M)$$

In realtà, per quanto sino ad ora dimostrato si sa per certo che:

$$IFP(N) \overset{\surd}{\Longrightarrow} RSA(M)$$
$$IFP(N) \overset{?}{\Longleftarrow} RSA(M)$$

Questo sta a significare che se uno potesse fattorizzare in un tempo polinomiale N si potrebbe ricavare anche M a partire da C sempre in un tempo polinomiale.
In altri termini, si suppone che RSA sia un problema difficile da risolvere come l'IFP.
Considerando ciò, quanto può essere difficile ricavare M a partire da C? Il metodo di fattorizzazione più efficiente a oggi conosciuto è l'NFS (Number Field Sieve) che gira in:

$$\mathcal{O}(\exp(c(\log N)^{1/3}(\log\log N)^{2/3}))$$

dove $c = (64/9)^{1/3}$ ma non si sa se esistano algoritmi polinomiali. Quindi l'RSA non si sa se sia un problema difficile, ma si assume che lo sia.

Capitolo 3

Reticoli

3.1 Introduzione ai reticoli

Un reticolo è un sottogruppo additivo di $\mathbb{R}^n$, in particolare ogni sottogruppo di $\mathbb{Z}^n$ lo è, ed è definito *reticolo intero*.
Un'analoga definizione di reticolo consiste in tutte le combinazioni lineari intere di un insieme di vettori (b_i) linearmente indipendenti:

$$L = \left\{ \sum_{i=1}^{n} n_i b_i | n_i \in \mathbb{Z} \right\}$$

L'insieme di questi vettori linearmente indipendenti è detto *base* del reticolo.
Tutte le basi hanno lo stesso numero di elementi, che identifica la dimensione di quest'ultimo, indicato con $dim(L)$.
Nel caso in cui la dimensione sia maggiore di 2, ci sono infinite basi, collegate tra loro da una matrice intera con determinante ± 1 ed hanno quindi lo stesso determinante di Gram [1], $det_{1 \leq i,j \leq d} \langle b_i, b_j \rangle$.
Il volume, indicato con $vol(L)$ è definito come la radice quadrata del determinante di Gram, quindi non è altro che il volume del parallelepipedo generato

[1] In algebra lineare, a Matrice di Gram di un insieme di vettori dato $v_1, \ldots, v_n$ in uno spazio con prodotto interno è la matrice Hermitiana i cui elementi sono: $G_{ij} = \langle v_j, v_i \rangle$

dai b_i.

Nel caso di un reticolo con dimensione massima, $dim(L) = n$, il volume è uguale al valore assoluto del determinante di una qualsiasi sua base, inolte, se il reticolo è intero, ossia $L \in \mathbb{Z}^n$, il volume è anche uguale all'indice del reticolo su $\mathbb{Z}^n$, $[\mathbb{Z}^n : L]$.

Dato che il reticolo è discreto, presenta un vettore di lunghezza minima diverso da zero: la norma Euclidea di questo vettore è detta il *primo minimo*, denotato da $\lambda_1(L)$ o $||L||$. In generale, viene definito l'*i-esmo minimo*$\lambda_i(L)$ di Minkowski il minimo del $max_{1\leq j\leq i}||v_j||$ con $v_1, \ldots, v_i \in L$ linearmente indipendenti.

Esistono ovviamente $v_1, \ldots, v_d$ linearmente indipendenti che raggiungono il minimo, che è indicato con $\lambda_i(L)$.

Sorprendentemente con dimensioni maggiori di 4 non è detto che questi vettori formino una base, e con dimensioni maggiori di 5 può non esistere una base di vettori che raggiunge il minimo.

Il teorema del corpo convesso di Minkowski garantisce l'esistenza del vettore di lunghezza minima in un reticolo: una più attenta applicazione mostra che ogni reticolo d-dimensionale L soddisfa $||L|| \leq \sqrt{d}vol(L)^{1/d}$, quindi il valore $\lambda_1(L)/vol(L)^{1/d}$ è limitato, da cui viene la costante di Hermite $\lambda_1(L)^2/vol(L)^{2/d}$ indicata con γ_d.

Il valore di questa costante è conosciuto solo per $d < 8$, la stima migliore conosciuta sino ad oggi per questa costante è la seguente:

$$\frac{d}{2\pi e} + \frac{\log(\pi d)}{2\pi e} + o(1) \leq \gamma_d \leq \frac{1.744d}{2\pi e}(1 + o(1))$$

3.2 Problemi algoritmici

In questo paragrafo useremo reticoli razionali, ossia $l \subseteq \mathbb{Q}$ di dimensione d.

Il problema più noto è l'SVP(*shortest vector problem*): data una base del reticolo L, trovare $\mathbf{u} \in L$ tale che $||\mathbf{u}|| = ||L||$ (si ricorda che $||L|| = \lambda_1(L)$).

Si può inoltre approssimare il quesito richiedendo un vettore di norma non

nulla limitato da un fattore di approssimazione: $||\mathbf{v}|| \leq f(d)||L||$.
Altro problema è il CVP (*closest vector problem*), detto anche il problema del punto più vicino nel reticolo, che è una versione non omogenea dell'SVP: data una base del reticolo L e un vettore $\mathbf{v} \in \mathbb{R}^n$ trovare un vettore del reticolo che minimizza la distanza da $\mathbf{v}$. Anche qui si può richiedere un vettore $\mathbf{u} \in L$ tale che per ogni $\mathbf{w} \in L, ||\mathbf{u} - \mathbf{v}|| \leq f(d)||\mathbf{w} - \mathbf{v}||$.

3.3 Algoritmo LLL

Di tutte le basi di un reticolo intero, quelle con un vettore di norma minima, sono dette *ridotte*. Dato che tutte le basi hanno lo stesso determinante, a meno del segno, significa che queste ultime sono più vicine ad essere ortogonali.
Il concetto di base ridotta è piuttosto vecchio, ma non ci sono degli algoritmi veramente soddisfacenti per trovarle, quello più perfomante è stato trovato nel 1982 da K.Lenstra, H.W.Lenstra e L.Lovasz con la nozione di riduzione LLL con relativo algoritmo.

Vediamo ora come si può applicare il metodo di riduzione delle basi nel caso bidimensionale:

Esempio 3.1. Siano (v_1, v_2) una base di un reticolo bidimensionale. Il nostro obiettivo è quello di trovare una base ridotta.
Se $||v_1|| > ||v_2||$, li scambiamo in modo che il primo vettore abbia norma minima.
Idealmente vorremmo sostituire v_2 con un vettore v_2^* perpendicolare al primo. Attraverso il procedimento di Gram-Schmidt, otteniamo il vettore:

$$v_2^* = v_2 \frac{v_1 \cdot v_2}{v_1 \cdot v_1} v_1$$

perpendicolare a v_1.
Tuttavia non è detto che questo vettore appartenga al reticolo, quindi sia t l'intero più vicino a $\frac{v_1 \cdot v_2}{v_1 \cdot v_1}$, (per definizione sia 0 l'intero più vicino a $\pm\frac{1}{2}$, 1 a

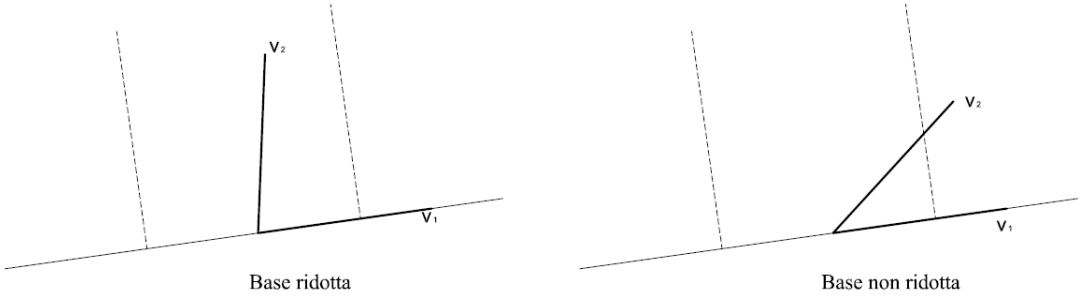
$\pm\frac{3}{2}$, e così via), quindi sostituiamo la nostra base con:

$$\{v_1, v_2 - tv_1\}$$

e ripetiamo il processo finchè:

$$||v_1|| < ||v_2||, \quad e \quad -\frac{1}{2} \leq \frac{v_1 \cdot v_2}{v_1 \cdot v_1} \leq \frac{1}{2}$$

il processo termina esattamente quando ho una base ridotta.



Definizione 3.1. Una la base $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$, si dice LLL ridotta se

$$|\mu_{i,j}| \leq \frac{1}{2} \quad per\ 1 \leq j < i \leq n$$

e

$$||b_i^* + \mu_{i,i-1} b_{i-1}^*||^2 \geq \frac{3}{4}||b_{i-1}^*|| \quad per\ 1 < i \leq n$$

o equivalentemente

$$||b_i^*||^2 \geq \left(\frac{3}{4} - \mu_{i,i-1}^2\right) ||b_{i-1}^*||^2$$

dove i $\mu_{i,j} = b_i^* \cdot b_j^* / b_j^* \cdot b_j^*$ sono i coefficienti di Gram-Schimdt e quindi i b_i^* sono le proizioni sul complemento ortogonale di Span$(v_1, \ldots, v_{i-1})$.

Da cui abbiamo il seguente teorema:

Teorema 3.1. *Sia* $B = \{\boldsymbol{b}_1 \boldsymbol{b}_2, \ldots, \boldsymbol{b}_n\}$ *una base LLL ridotta,*$|det(B)| = d(L)$ *allora:*

1. $$d(L) \leq \prod_{i=1}^{n} ||b_i^*|| \leq 2^{n(n-1)/4} d(L)$$

2. $$||b_j^*|| \leq 2^{(i-1)/2} ||b_i^*||, \quad se 1 \leq j \leq i \leq n$$

3. $$||b_1|| \leq 2^{(n-1)/4} d(L)^{1/n}$$

   Per ogni $\boldsymbol{x} \in L$ *con* $\boldsymbol{x} \neq \boldsymbol{0}$ *abbiamo:*

4. $$||b_1|| \leq 2^{(n-1)/2} ||x||$$

5. *Più in generale per ogni insieme di vettori linearmente indipendenti* $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t \in L$ *abbiamo:*

   $$|b_j| \leq 2^{(n-1)/2} max(|\boldsymbol{x}_1|, \ldots, |\boldsymbol{x}_t|) \quad per 1 \leq j \leq t$$

Prima di dimostrarlo premettiamo un corollario:

Corollario 1 (Disuguaglianza di Hadamard)**.** Sia L un reticolo di determinante d(L), q una forma quadratica, $(b_i)_{1 \leq i \leq n}$ una $\mathbb{Z}$ base di L, e per ogni $x \in L$ si scriva $|x|$ per $q(x)^{1/2}$, allora:

$$d(L) \leq \prod_{i=1}^{n} |b_1|$$

Equivalentemente, se B è una matrice $n \times n$,

$$|det(B)| \leq \prod_{1 \leq i \leq n} \left(\sum_{1 \leq j \leq n} |b_{i,j}|^2 \right)^{1/2}$$

Dimostriamo il corollario,

Dimostrazione. Se poniamo $B_i = |b_i^*|^2$, l'ortogonalità di b_i^* implica che

$$q(b_i) = |b_i|^2 = B_i + \sum_{1 \leq j < i} \mu_{i,j}^2 B_j$$

da cui $d(L) = \prod_{1 \leq i \leq n} B_i \leq \prod_{1 \leq i \leq n} |b_i|^2$ □

Dimostriamo ora il teorema:

Dimostrazione. La prima disuguaglianza del punto 1, ponendo $B_i = |b_i|^2$ è esattamente il corollario. Poichè i b_i sono LLL ridotti, abbiamo $B_i \geq (3/4 - \mu_{i,i-1}^2)B_i - 1 \geq B_{i-1}/2$ poichè $\mu_{i,i-1} \leq 1/2$. Per induzione si vede che $B_j \leq 2^{i-j} B_i$ per $i \geq j$, da cui

$$b_i^2 \leq \frac{2^{i-1}+1}{2} B_i,$$

e ciò implica il primo punto del teorema.
Combinando le due disuguaglianze che abbiamo appena ottenuto, si ha che per ogni $j < i$,$b_j^2 \leq (2^{i-2} + 2^{i-j-1})B_i)$ che implica il punto 2. Se imponiamo $j = 1$ in quest'ultimo e facciamo il prodotto per i che va da 1 a n, otteniamo $(b_i^2)^n \leq 2^{n(n-1)/2} \prod_{1 \leq j \leq i} r_j b_j = \sum_{1 \leq j \leq i} s_j b_j^*$ con $r_i \neq 0$, $r_j \in \mathbb{Z}$ e $s_j \in \mathbb{R}$. É chiaro dalla definizione dei b_j^* che $r_i = s_i$, da cui

$$|x|^2 \geq s_i^2 B_i = r_i^2 B_i \geq B_i$$

poichè r_i è un intero diverso da zero, e poichè vale il punto 2 sappiamo che $B_i \geq 2^{1-i}|b_1|^2 \geq 2^{1-n}|b_1|^2$, e quindi abbiamo dimostrato il quarto punto, il quinto è invece solo una generalizzazione di quest'ultimo. □

Vediamo ora l'algoritmo LLL:

Algoritmo 1 (Algoritmo LLL)**.** Data una base $b_1, b_2, \ldots, b_n$ di un reticolo (L,q), questo algoritmo trasforma i vettori b_i in modo tale che al termine, questi formino una base LLL. In aggiunta l'algoritmo fornisce una matrice H con le coordinate della base LLL ridotta nei termini della precedente. Indicheremo con H_i le colonne di H.

1. Inizializzazione: Si pone $k \leftarrow 2$, $k_{max} \leftarrow 1, b_1^* \leftarrow b_1, B_1 \leftarrow b_1 \cdot b_1$ e $H \leftarrow I$.

2. Gram-Schmidt incrementale: Se $k \leq k_{max}$ vai allo step 3, altrimenti si pone $k_max \leftarrow k, b_k \leftarrow b_k$ poi per $j = 1, \ldots, k-1$ si pone $\mu_{k,j} \leftarrow b_k \cdot b_j^*/B_j$ e $b_k^* \leftarrow b_k^* - \mu_{k,j} b_j^*$. Infine, $B_K \leftarrow b_k^* \cdot b_k^*$. Se $B_k = 0$ si termina con un messaggio di errore dicendo che i b_i non formano una base LLL ridotta e termina l'algoritmo.[2]

3. Test delle condizioni LLL: Esegui il sottoalgoritmo RED(k,k-1) riportato qua sotto. Se $B_k < (0.75 - \mu_{k,k-1}^2)$ esegui il sottoalgoritmo SWAP(k) qua sotto, poni $k \leftarrow max(2, k-1)$ e vai allo step 3, altrimenti per $l = k-2, k-3, \ldots, 1$ esegui il sottoalgoritmo RED(k,l) e quindi $k \leftarrow k+1$

4. Controllo finale: Se $k \leq n$ vai allo step 2, altrimenti dai come output i vettori b_i che ora formano una base LLL ridotta, la matrice di trasformazione $H \in GL_n(\mathbb{Z})$ e termina l'algoritmo.

Algoritmo 2 (RED(k,l))**.** Se $\mu_{k,l} \leq 0.5$ termina il sottoalgoritmo, altrimenti, sia q l'intero più vicino a $\mu_{k,l}$ i.e.

$$q \leftarrow \lfloor \mu_{k,l} \rceil = \lfloor 0.5 + \mu_{k,l} \rceil.$$

Poni $b_k \leftarrow b_k - q b_l, H_k \leftarrow H_k - q H_l, \mu_{k,l} - q$, *per* $1 \leq i \leq l-1$, sia $\mu_{k,l} \leftarrow \mu_{k,i} - q\mu_{l,i}$ e termina l'algoritmo.

Algoritmo 3 (SWAP(k))**.** Scambia i vettori b_k e b_{k-1}, H_k e H_{k-1} e se $k > 2$, per ogni j tale che $i \leq j \leq k-2$ scambia $\mu_{k,j}$ con $\mu_{k-1,j}$. Poi, si pongano, $\mu \leftarrow \mu_{k,k-1}, B \leftarrow B_k + \mu^2 B_{k-1}, \mu_{k,k-1} \leftarrow \mu B_{k-1}/B, b \leftarrow b_{k-1}^*, b_{k-1}^* \leftarrow b_k^* + \mu b, b_k^* \leftarrow -\mu_{k,k-1} b_k^* + (B_k/B)b, B_k \leftarrow B_{k-1} B_k/B$ *e* $B_{k-1} \leftarrow B$. Infine per $i = k+1, k+2, \ldots, k_{max}$ si ponga $t \leftarrow \mu_{i,k}, \mu_{i,k} \leftarrow \mu_{i,k-1} - \mu t, \mu_{i,k-1} \leftarrow t + \mu_{k,k-1}\mu_{i,k}$ e finisci l'algoritmo.

[2]Quindi l'algoritmo LLL può fallire, vedi Henri Cohen, A Course in Computational Algebraic Number Theory, pagina 87

Corollario 2. L'algoritmo LLL termina in tempo polinomiale.

Dimostrazione. Si vede che all'inizio dello step 4 le condizioni LLL sono valide per $i \leq k-1$. Quindi, se $k > n$ abbiamo ottenuto una famiglia LLL ridotta, ed è chiaro che le operazioni effettuate sui b_i hanno determinante ± 1 e quindi questa famiglia è anche una base di L e l'algoritmo è corretto.
Dobbiamo quindi mostrare che l'algoritmo effettivamente giunge al termine. Se poniamo per $0 \leq i \leq n$

$$d_i \det((b_r \cdot b_s)_{1 \leq r,s \leq i})$$

si può facilmente vedere che

$$d_i = \prod_{1 \leq j \leq i} B_j$$

dove come al solito $B_i = |b_i^*|^2$ e in particolare $d_i > 0$ e quindi $d_0 = 1$ e $d_n = d(L)^2$. Sia

$$D = \prod_{1 \leq i \leq n-1} d_i$$

questo può cambiare solo se cambia B_i, e ciò succede solo nell'algoritmo SWAP. Nel sottoalgoritmo i d_i vengono cambiati per $i < k-1$ e per $i > k$, e dallo step 3, d_{k-1} viene moltiplicato al massimo da un fattore di $3/4$, quindi anche D può essere al massimo ridotto dello stesso valore.
Sia L_i, il reticolo di dimensione i generato dai b_j con $j \leq i$ e sia s_i il valore più piccolo diverso da zero della forma quadratica q in L_i, usando la costante di Hermite otteniamo:

$$d_i \geq s_i^i \gamma_i^{-i} \geq s_n^i \gamma_i^{-i}$$

e poichè s_n è il più piccolo valore diverso da zero di q(x) questa espressione dipende solo dalle i e non dai b_j, ne segue che i d_i sono limitati inferiormente da una costante positiva che dipende solo da i ed L. Poichè D è limitato da una costante positiva dipendente solo da L, si vede che il numero di iterazioni dell'algoritmo SWAP deve essere finito, giacchè questa è l'unica parte in cui k decresce, l'algoritmo deve essere finito e ciò prova la sua validità. □

Enunciamo solo un lemma che ci sarà poi in seguito utile per un eventuale attacco all'RSA:

Lemma 1. Se un elemento $\mathbf{s}$ di un reticolo L soddisfa le segunte proprietà:

$$|\mathbf{s}| < D^{1/n} 2^{-(n-1)/4}$$

allora $\mathbf{s}$ giace sull'iperpiano generato da $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n-1}$, dove $D = \prod_{1 \leq i \leq n-1} d_i$ e i b_i sono i vettori della base ridotta.

Capitolo 4

Il teorema di Coppersmith

4.1 Introduzione

Per migliorare l'efficienza della funzione di cifratura dell'RSA:

$$C \equiv M^e \ (mod\ N)$$

è vantaggioso utilizzare esponenti pubblici (e) piccoli come $e = 3 = (11)_2$ o $e = 17 = (10001)_2$.
Alcuni attacchi a questo criptosistema si basano appunto su questo fatto, uno di questi è l'attacco ideato da Coppersmith.

4.2 Trovare le radici di un'equazione polinomiale di grado basso

4.2.1 Equazioni modulari in una variabile

Il problema di risolvere equazioni polinomiali in una variabile modulo un qualche intero N, di fattorizzazione ignota sembra essere difficile. Coppersmith invece usando l'algoritmo LLL dimostrò che trovare delle radici piccole, caso di interesse per l'RSA, è piuttosto facile.

Sia N un numero composto intero la cui fattorizzazione non è conosciuta, sia

$$p(x) = x^{\delta} + p_{\delta-1}x^{\delta-1} + \cdots + p_2x^2 + p_1x + p_0$$

un polinomio intero di grado δ nella singola variabile x. Supponiamo esista una soluzione intera x_0 a

$$p(x_0) = 0 \ (mod\ N)$$

che soddisfa

$$|x_0| < N^{1/\delta}$$

Coppersmith è riuscito a trovare la soluzione x_0 in tempo polinomiale in $(\log N, 2^{\delta})$. Per farlo utilizzeremo i coefficienti del polinomio p per costruire una matrice M, le cui righe sono la base di un reticolo intero. Considereremo il vettore colonna $\mathbf{r}$ i cui elementi sono potenze della soluzione desiderata: x_0^i. Il vettore $\mathbf{s} = \mathbf{r}M$ sarà un elemento relativamente corto del reticolo. Applicando il metodo di riduzione delle basi troviamo l'iperpiano contenente tutti i vettori di norma piccola[1], l'equazione di questo iperpiano si trasla con una relazione lineare sugli elementi di $\mathbf{r}$, ed in seguito ad un'equazione lineare $c(x_0)$ su $\mathbb{Z}$, equazione che si risolverà direttamente.
Una importante applicazione è, come già spiegato, l'attacco all'RSA basato su questo metodo, quando la maggior parte del messaggio è fissata o stereotipata. Supponiamo che il testo in chiaro m consista di due parti, una parte conosciuta B e una non conosciuta x, cioè $m = B+x$. Supponendo che questo messaggio sia criptato con esponente 3, $c = m^3 = (B + x)^3 \ (mod\ N)$. Se si conosce B, c e N, possiamo applicare il risultato appena esposto all'equazione modulare polinomiale seguente:

$$p(x) = (B + x)^3 - c = 0 \ \ (mod\ N)$$

e ricavare x nel caso in cui $|x| < N^{1/3}$, in modo tale che x abbia meno di un terzo dei bit del messaggio e che siano tutti consecutivi.

[1] vedi Lemma 1 del capitolo sui Reticoli

Altra applicazione, sempre come attacco all'RSA con esponente piccolo è quando siamo alla presenza di padding casuale, supponiamo che a un messaggio m sia aggiunto un valore casuale r_1 prima di criptarlo con esponente 3, dando così il seguente testo cifrato:

$$c_1 = (m + r_1)^3 \pmod{N}$$

e anche che questa azione venga reiterata una seconda volta con un valore, sempre casuale, r_2:

$$c_2 = (m + r_2)^3 \pmod{N}$$

Si mostrerà come ricavare m da $c_1, c_2.N$, nel caso in cui gli r_i abbiano un numero di bit inferiore ad $\frac{1}{9} \log N$.

4.2.2 Costruzione della matrice M

Sia N un numero intero grande di fattorizzazione ignota e sia dato il seguente polinomio monico:

$$p(x) = x^{\delta} + p_{\delta-1}x^{\delta-1} + \cdots + p_2x^2 + p_1x + p_0 \pmod{N}$$

che assumiamo essere monico, cioè, $p_\delta = 1$ Supponiamo esista un intero x_0 che soddisfi

$$p(x_0) = 0 \pmod{N}$$

con

$$|x_0| < \frac{N^{1/\delta} - \varepsilon}{2} = \mathcal{O}(\sqrt[\delta]{N})$$

per un qualche $\varepsilon > 0$.
Iniziamo scegliendo un intero

$$h \geq \max\left(\frac{\delta - 1 + \varepsilon\delta}{\varepsilon\delta^2}, \frac{7}{\delta}\right)$$

Da qui segue che

$$\frac{h-1}{h\delta - 1} \geq \frac{1}{\delta} - \varepsilon$$

e anche che $h\delta \geq 7$.
Per ogni coppia di interi i, j con $0 \leq i < \delta, 1 \leq j < h$, definiamo il polinomio

$$q_{ij}(x) = x^i p(x)^j$$

Per la soluzione desiderata x_0 sappiamo che $p(x_0) = y_0N$ per qualche intero y_0, tale che

$$q_{ij}(x_0) = 0 \pmod{N^j}$$

Costruiremo una matrice razionale M di dimensione $(2h\delta - \delta) \times (2h\delta - \delta)$, usando i coefficienti dei polinomi $q_{ij}(x)$, in maniera tale che una combinazione lineare delle righe della matrice M dia le potenze di x e y_0 darà un vettore

di norma Euclidea relativamente piccola, Una volta moltiplicata la matrice in modo da renderla intera, possiamo applicare l'algoritmo di riduzione.
La matrice M è divisa in due blocchi, il blocco superiore destro, $(h\delta) \times (h\delta - \delta)$, presenta delle righe indicizzate da un intero g con $0 \leq g \leq h\delta$, e delle colonne da $\gamma(i,j) = h\delta + i + (j-1)\delta$ con $0 \leq i < \delta$ e $1 \leq j < h$, tali che $h\delta \leq \gamma(i,j) < 2h\delta - \delta$. L'elemento $(g, \gamma(i,j)$ è il coefficiente di x^g del polinomio $q_{i,j}(x)$.
Il blocco in basso a destra, $(h\delta - \delta) \times (h\delta - \delta)$, è diagonale, con il valore N^j in ogni colonna $\gamma(i,j)$.
Il blocco superiore sinistro, $(h\delta) \times (h\delta)$, è diagonale, il suo valore nella riga g è l'approssimazione razionale di $X^{-g}/\sqrt{h\delta}$ dove $X = \frac{1}{2}N^{(1/\delta)-\varepsilon}$ è un limite superiore alla soluzione $|x|$ di nostro interesse.
Il blocco inferiore sinistro, $(h\delta - \delta) \times (h\delta)$, è di zeri.
Per semplificare la comprensione, riportiamo al matrice M nel caso di $h = 3, \delta = 2$, assumiamo $p(x) = x^2 + ax + b$ e $p(X)^2 = x^4 + cx^3 + dx^2 + ex + f$, e pre semplificare la notazione scriviamo τ al posto di $1/\sqrt{h\delta}$.

$$M = \begin{pmatrix} \tau & 0 & 0 & 0 & 0 & 0 & b & 0 & f & 0 \\ 0 & \tau X^{-1} & 0 & 0 & 0 & 0 & a & b & e & f \\ 0 & 0 & \tau X^{-2} & 0 & 0 & 0 & 1 & a & d & e \\ 0 & 0 & 0 & \tau X^{-3} & 0 & 0 & 0 & 1 & c & d \\ 0 & 0 & 0 & 0 & \tau X^{-4} & 0 & 0 & 0 & 1 & c \\ 0 & 0 & 0 & 0 & 0 & \tau X^{-5} & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & N & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N^2 \end{pmatrix}$$

Le colonne di M generano un reticolo, del quale a noi interessa il vettore $\mathbf{s}$, relativo alla soluzione x_0. Si consideri un vettore $\mathbf{r}$ i cui elementi sono potenze di x_0:

$$r_g = x_0^g$$

e

$$r_{\gamma(i,j)} = -x_0^i y_0^j$$

$$\mathbf{r} = (1, x_0, x_0^2, \ldots, x_0^{h\delta-1}, -y_0, -x_0^{\delta-1}y_0, -y_0^2, -x_0y_0^2, \ldots, -x_0^{\delta-1}y_0^{h-1})$$

Il prodotto $\mathbf{s} = \mathbf{r}M$ è un vettore riga tale che:

$$s_g = \frac{(x_0/X)^g}{\sqrt{h\delta}}$$

e

$$s_{\gamma(i,j)} = q_{ij}(x_0) - x_0^i y_0^j N^j = 0$$

Una stima della norma Euclidea si $\mathbf{s}$ è la seguente:

$$|\mathbf{s}| = \left[\sum_g s_g^2\right]^{1/2} < \left[\sum_g \left(\frac{1}{\sqrt{h\delta}}\right)^2\right]^{1/2} = 1$$

Possiamo restringere la nostra attenzione ad un sottoreticolo $\hat{M}$ di M, questo perchè gli $h\delta - \delta$ elementi a destra di $\mathbf{s}$ sono zeri.
$\hat{M}$ è formato semplicemente dai punti che hanno le $h\delta - \delta$ coordinate finali uguali a 0, quindi:
$\hat{M} = M \cap (\mathbb{R}^{h\delta} \times \{0\}^{h\delta-\delta})$.
Per fare ciò, computazionalmente, traiamo vantaggio dal fatto che $p(x)$ e quindi $q_{ij}(x)$ siano monici, quindi le $h\delta-\delta$ righe a destra del blocco superiore di M formano una matrice triangolare superiore con degli uno sulla diagonale. Questo implica che possiamo effettuare delle operazioni elementari su M per ottenere una matrice a blocchi $\tilde{M}$ il cui blocco $(h\delta - \delta) \times (h\delta - \delta)$ inferiore destro è l'identità e il cui blocco $(h\delta) \times (h\delta - \delta)$ superiore destro è di zeri.
Il blocco superiore sinistro $(h\delta) \times (h\delta)$ $\hat{M}$ di $\tilde{M}$ rappresenta il sottoreticolo desiderato: un reticolo $h\delta$-dimensionale del quale $\mathbf{s}$ è il vettore di norma minima.

4.2.3 Analisi del determinante

M è una matrice triangolare superiore, quindi il suo determinante è il prodotto degli elementi diagonali:

$$\begin{aligned}\det(M) &= \prod_g \frac{1}{X^g\sqrt{h\delta}} \prod_j N^j \\ &= \frac{N^{\delta h(h-1)/2} X^{-(h\delta)(h\delta-1)/2\,h\delta}}{\sqrt{h\delta}} \\ &= \left[N^{(h-1)/2} X^{-(h\delta-1)/2} (h\delta)^{-1/2}\right]^{h\delta}\end{aligned}$$

Per costruzione

$$\det(M) = \det(\tilde{M}) = \det(\hat{M}) \times \det(I) = \det(\hat{M})$$

Usiamo quindi ora il Lemma 1 sulla matrice $\hat{M}$, la cui dimensione è $n = h\delta$.
Dato che sappiamo

$$||s|| < 1$$

la condizione richiesta è

$$1 \leq |\det(\hat{M})|^{1/h\delta}\ 2^{-(h\delta-1)/4}.$$

Poichè

$$\det(\hat{M}) = (N^{(h-1)/2} X^{-(h\delta-1)/2} (h\delta)^{-1/2})^{h\delta}$$

si ha

$$1 \leq N^{(h-1)/2} X^{-(h\delta-1)/2} (h\delta)^{-1/2} 2^{-1/2}$$

$$X \leq N^{(h-1)/(h\delta-1)} (h\delta)^{-1/(h\delta-1)} 2^{-1/2}$$

Quindi le ipotesi del Lemma 1 sono soddisfatte se

$$X \leq N^{(h-1)/(h\delta-1)} (h\delta)^{-1/(h\delta-1)} 2^{-1/2}$$

Con la nostra scelta di h abbiamo $h\delta \geq 7$, quindi

$$(h\delta)^{-1/(h\delta-1)} > 2^{-1/2}$$

Sempre dalla nostra scelta di h sappiamo che

$$\frac{h-1}{h\delta-1} \geq \frac{1}{\delta} - \varepsilon$$

Quindi se scegliamo

$$X \leq \frac{1}{2} N^{(1/\delta)-\varepsilon}$$

avremo

$$|\mathbf{s}| < 1 \leq \det(\hat{M})^{1/n} 2^{-(n-1)/4}$$

come richiesto.

4.2.4 Trovare la soluzione desiderata

Applicando il teorema di riduzione delle basi alle righe della matrice $\hat{M}$ si produce una base $\mathbf{b}_1^*, \mathbf{b}_2^*, \ldots, \mathbf{b}_n^*$ che soddisfa:

$$||\mathbf{b}_n^*|| \geq \det(\hat{M})^{1/n} 2^{-(n-1)/4}$$

dove, come prima $n = h\delta = dim(\hat{M})$. Grazie ai calcoli della precedente sezione abbiamo quindi che

$$||\mathbf{b}_n^*|| \geq 1.$$

Grazie al Lemma 1, tutti i vettori del reticolo generato dalle righe di $\hat{M}$ con norma minore di 1 devono giacere nell'iperpiano generato da $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n-1}$. In termini della matrice più grande M e dei vettori $\mathbf{r}$,$\mathbf{s}$ con $\mathbf{r}M = \mathbf{s}$, c'è uno spazio $h\delta$-dimensionale di vettori $\mathbf{r}$ tale che $\mathbf{r}M = \mathbf{s}$ di zeri negli $(h\delta - \delta)$ elementi a destra. Dal Lemma 1, i vettori interi $\mathbf{r}$ che soddisfano $||\mathbf{s}|| < 1$ devono giacere in uno spazio di dimensione inferiore di una unità, cioè $h\delta - 1$. Questo determina una equazione lineare sugli elementi r_g con $0 \leq g < h\delta$,

grazie alla quale otteniamo i coefficienti, non tutti nulli, c_g, tali che:
Per ogni vettore intero $\mathbf{r} = (r_g, r_{\gamma(i,j)})$ tale che gli elementi a destra di $\mathbf{s} = \mathbf{r}M$ sono 0 e $||s|| < 1$, abbiamo:

$$\sum c_g r_g = 0$$

Questo vale per tutti i vettori $\mathbf{s}$ di norma piccola nel reticolo con gli elementi a destra nulli, in particolare anche per i vettori ottenuti da $\mathbf{r}$ dove:

$$r_g = x_0^g, \qquad r_{\gamma(i,j)} = -x_0^i y_0^j$$

Quindi otteniamo un polinomio $C(x)$ con coefficienti c_g che viene soddisfatto dalla soluzione x_0

$$C(x_0) = \sum c_g x_0^g = 0$$

Notiamo che l'equazione qua sopra è in $\mathbb{Z}$ (non modulo N), quindi possiamo risolverla facilmente usando tecniche tradizionali ed ottenere così x_0.

Nota 1. Se il polinomio ha più soluzioni piccole x_0, la procedura le trova tutte simultaneamente e tutte sono soluzioni del polinomio

$$C(x_0) = \sum c_g x_0^g = 0$$

Abbiamo visto:

Teorema 4.1. *Sia p(x) un polinomio di grado δ in una variabile modulo un intero N di fattorizzazione ignota. Sia X un limite superiore per la soluzione desiderata X_0, se*

$$X < \frac{1}{2} N^{1/\delta - \varepsilon}$$

allora in tempo polinomiale in $(\log N, \delta, 1/\varepsilon)$, possiamo trovare un intero x_0 con $p(x_0) = 0 \ (mod\ N)$ e $|x_0| < X$

Dimostrazione. L'algoritmo di riduzione della base di un reticolo opera su una matrice di dimensione $h\delta = \mathcal{O}(\delta/\varepsilon)$, e le entrate delle matrice non sono troppo grandi; come visto nella sezione degli algoritmi, richiede un tempo polinomiale, ciò vale anche per gli altri passaggi. □

Corollario 3. Con le ipotesi del teorema precedente, eccetto per

$$X \leq N^{1/\delta}$$

in tempo polinomiale in $(\log N, 2^\delta)$, possiamo trovare numeri interi x_0 tali che $p(x_0) = 0 \ mod(\ N)$ e $|x_0| \leq X$

Dimostrazione. Ricopriamo l'intervallo $\left[-N^{1/\delta}, N^{1/\delta}\right]$ con quattro intervalli I_i di lunghezza $\frac{1}{2}N^{1/\delta}$ ciascuno centrato in un intero x_i, e applichiamo il teorema precedente con $\varepsilon = 1/\log N$ al polinomio $p_i(x) = p(x + x_i)$ per trovare la soluzione $x_0 = x + x_i$ negli intervalli I_i, in un tempo polinomiale in $(\log N, 2^\delta)$ □

4.3 Attacchi all'RSA

4.3.1 Messaggi Stereotipati

Un'importante applicazione di questo metodo, come già preannunciato, è un attacco all'RSA con esponente di cifratura piccolo e con parte del messaggio stereotipato.
Supponiamo che il testo in chiaro m consista di due pezzi:

1. Una parte conosciuta $B = 2^k b$, ad esempio la rappresentazione ASCII di una data conosciuta.

2. Una parte sconosciuta x, che contiene l'informazione vera e propria, lunga meno di un terzo di quella di N

Supponiamo che sia criptato con $e = 3$, quindi il testo cifrato c è dato da $c = m^3 = (B + x)^3 \ (mod\ N)$. Se conosciamo B, c, N, possiamo applicare il presente risultato al polinomio $p(x) = (B+x)^3 - c$ e ricavare x_0 che soddisfa:

$$p(x_0) = (B + x_0)^3 - c = 0 \ \ (mod\ N)$$

nell'ipotesi che $|x_0| < N^{1/3}$.

Nota 2. Il limite X dipende dal modulo N. Se x_0 ha 250 bit e N ne ha 512, e l'esponente usato è 3, la tecnica fallisce perchè $x_0 > n^{1/3}$, ma se incrementiamo a un modulo di 1024 bit mantenendo costante il numero di bit di x_0 possiamo portare l'attaco.

L'attacco funziona in egual modo sia quando i bit di x_0 sono i bit più significativi sia nel caso opposto.

4.3.2 Messaggi con padding casuale

Supponiamo che due messaggi m ed m' soddisfino una relazione affine conosciuta:

$$m' = m + r$$

con r noto, Supponiamo di conoscere i testi cifrati con esponente 3:

$$c = m^3 \ (mod\ N)$$
$$c' = (m')^3 = m^3 + 3m^2r + 3mr^2 + r^3 \ (mod\ N)$$

Possiamo ricavare m col metodo di Franklin e Reiter da c, c', r e N:

$$m = \frac{r(c' + 2c - r^3)}{c' - c + 2r^3} = \frac{r(3m^3 + 3m^2r + 3mr^2)}{3m^2r + 3mr^2 + 3r^3} \ (mod\ N)$$

Quello che noi non sappiamo è la relazione tra m ed m', ma sappiamo che r è piccolo:

$$m' = m + r$$
$$|r| < N^{1/9}$$

Possiamo ancora trovare m?
Si può immaginare una situazione dove i messaggi M sono soggetti a padding casuale prima di essere criptati tramite l'RSA con esponente 3. Potrebbe essere che M venga traslato a sinistra di k bit, e che venga aggiunta una

stringa casuale di k bit, detta R, per formare il testo in chiaro m, il testo cifrato c sarà il cubo di quest'ultimo modulo N:

$$c = m^3 = (2^k M + R)^3 \ (mod\ N)$$

Supponiamo che lo stesso messaggio M sia criptato due volte, con padding R ed R' differenti. Sia $R' = R + r$ così che il secondo testo in chiaro sia $m' = m + r$, ora avremo i seguenti due testi cifrati:

$$c = m^3 = (2^k M + R)^3 \ (mod\ N),$$
$$c' = (m')^3 = (2^k M + R')^3 = (m + r)^3 \ (mod\ N)$$

Si possono ricavare m ed r conoscendo c, c' ed N?
Possiamo eliminare m dalle due equazioni sopra:

$$r^9 + (3c - 3c')r^6 + (3c^2 + 21cc' + 3(c')^2)r^3 + (c - c')^3 = 0 \ (mod\ N).$$

Questo è un polinomio in una variabile di grado 9 $(mod\ N)$, se la sua soluzione soddisfa $|r| < N^{1/9}$, possiamo applicare il metodo sopra esposto per ricavare r ed in seguito applicare il metodo di Franklin e Reiter per ricavare m e ricavarci quindi M.
Come prima questo metodo funziona sia nei bit più significativi che in quelli meno.
É chiaro che se il padding è fatto di un numero di bit inferiore a un nono rispetto ad N e con esponente 3, l'algoritmo trova il testo in chiaro, per esempio con una chiave RSA di 1024 bit, tollera facilmente un padding di 100 bit.
Ci sono tuttavia vari metodi per evitare questo attacco:

1. Confondere il messaggio con altri metodi.

2. Distribuire il padding in diversi blocchi non contigui, così che il presente attacco debba essere modificato.

3. Distribuire il padding su tutto il messaggio: ad esempio si possono aggiungere due bit ogni otto.

4. Aumentare la lunghezza del padding per diminuire l'efficienza dell'algoritmo.

5. Rendere il padding dipendente dal messaggio ad esempio tramite una funzione *hash*.

6. Usare esponenti pubblici più grandi, ad esempio per $e = 7$ su una chiave RSA di 1024 bit l'attacco tollera al massimo un padding di 21 bit.

4.4 Conclusione

In questo elaborato abbiamo visto un metodo di applicazione di un algoritmo di risoluzione del problema del vettore di norma minima in un reticolo come attacco al sistema di crittografia a chiave pubblica RSA.
L'attacco, come visto, si può apportare solo in casi particolari, quindi con esponenti piccoli, come $e = 3$, con messaggi stereotipati o con padding casuale, inoltre si è mostrato come possa fallire non appena si incrementi o l'esponente o anche solo la lunghezza del padding.
Il problema del vettore di norma minima è comunque un problema intrattabile quando la dimensione del reticolo diventa abbastanza grande, proprio per questo l'uso visto in questo elaborato non è l'unico possibile in crittografia, infatti stanno nascendo numerosi sistemi crittografici basati su questo.

Bibliografia

[1] Son Y.Yan, *Cryptanalytic Attacks on RSA*, Springer, 2008.

[2] Phong Q. Nguyen and Jacques Stern, *The Two Faces of Lattices in Cryptology*, École Normale Supériure Départment d'Informatique,2001.

[3] Dan Boneh, *Twenty Years of Attacks on the RSA Cryptosystem*, Notices of The AMS, Volume 42, Number 2, 1999

[4] Henri Cohen, *A course in Computational Algebraic Number Theory*, Springer, Terza Edizione, 1996

[5] Don Coppersmith, *Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities*, Journal of CRYPTOLOGY, 1997

[6] W.Trappe, L.C.Washington, *Introduction to Cryptography with Coding Theory*, Pearson-Pentice Hall, 2009