Problema Learning With Errors e sue applicazioni Crittografiche

Luzietti Manuel (n.0000937684)

November 2022

Capitolo 1

Introduzione

La crittografia Lattice based ha ottenuto inizialmente molto successo grazie al fatto che le possibili costruzioni di sistemi crittografici basati su primitive di tale tipo sarebbero accompagnate da prove di sicurezza basate su istanze worst-case di problemi Lattice Based difficili. Il primo schema Lattice based fu proposto da Ajtai e Dwork ed in seguito semplificato e migliorato da Regev. Uno dei più grandi successi di Regev nel lavorare su tali schemi, fu l'introduzione di un problema intermedio chiamato Learning With Error (LWE), che era relativamente facile da usare in applicazioni crittografiche e difficile almeno quanto istanze worst-case di problemi Lattice based. Nella seguente relazione si vuole introdurre il problema LWE, analizzandone la struttura, dando cenni sulla difficoltà di tale problema, studiando uno degli algoritmi (classici) risolutivi più efficienti ad oggi conosciuti (BKW algorithm) ed infine esponendo alcune possibili implementazioni crittografiche.

Essendo LWE riducibile ad istanze di problemi Lattice based e non essendoci ad oggi algoritmi efficienti per risolvere LWE o istanze di problemi Lattice-based a cui LWE può essere ridotto, LWE viene oggi considerato come primitiva per creare sistemi crittografici Quantum-proof, ovvero resistenti anche ad attacchi computazionali di computer quantistici. Questo è un punto molto importante in un presente in cui, grazie all'algoritmo di Shor, i principali sistemi crittografici moderni basati su problemi come fattorizzazione di numeri primi, logaritmi discreti e logaritmi discreti su curve ellittiche, sono in pericolo. Da qui la nascita della Post-Quantum Cryptography, ovvero crittografia che sia resistente ad attacchi crittoanalitici supportati da computer quantistici. I problemi Lattice based sono una di quelle branche promettenti per un futuro standard crittografico. Proprio quest' anno infatti il NIST ha

selezionato i primi quattro algoritmi che faranno parte del primo standard di Post-Quantum Cryptography , tra i quali è presente CRYSTALS-Kyber, un algoritmo di cifratura a chiave pubblica e scambio di chiavi, basato su M-LWE (un'estensione di LWE).

Un altro punto importante è la possibilità di creare schemi crittografici completamente Omomorfici, che permettono di eseguire computazioni su dati cifrati senza alterarne il contenuto informativo e senza conoscerne le chiavi.

Capitolo 2

LWE problem

LWE, introdotto da Oded Regev, è una generalizzazione del problema più noto LPN (Learning parity with Noise) che è presunto essere NP-hard. Le ragioni per cui LWE è popolare come primitiva crittografica risiedono nella sua semplicità concettuale e nelle argomentazioni teoriche riguardanti la sua difficoltà risolutiva. Infatti in seguito alla riduzione da problemi Lattice-based (worst-case) a problemi LWE (average-case), la costruzione di un sistema crittografico basato su questa primitiva risulterebbe sicura sotto l'assunzione che i problemi Lattice-based (worst-case) sono difficili.

Definizione LWE. Siano n, q interi positivi, χ distribuzione di probabilità su $\mathbb{Z}$ e s vettore segreto che segue distribuzione uniforme su $\mathbb{Z}_q^n$. Denotiamo con $L_{s,\chi}$ la distribuzione di probabilità su $\mathbb{Z}_q^n \times \mathbb{Z}_q$ ottenuta scegliendo a da distribuzione uniforme su $\mathbb{Z}_q^n$, e $e \in \mathbb{Z}$ in accordo a χ, che restituisce in output $(a, c) = (a, \langle a, s \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. Trovare $s \in \mathbb{Z}_q^n$ avendo campioni di coppie $(a_i, c_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ che seguono la distribuzione $L_{s,\chi}$. Problema anche noto come Search-LWE che di distingue da Decision-LWE il cui scopo è decidere se i campioni $(a_i, c_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ seguono la distribuzione $L_{s,\chi}$ o la distribuzione uniforme su $\mathbb{Z}_q^n \times \mathbb{Z}_q$. Il modulo q è tipicamente scelto polinomiale in n, e χ distribuzione Gaussiana su $\mathbb{Z}$ con media 0 e deviazione standard $\sigma = \alpha q$, per qualche α.

In poche parole il search-LWE problem richiedere di recuperare un vettore segreto s da una sequenza di equazioni lineari casuali approssimate. Un

esempio di input potrebbe essere:

$$\begin{aligned} 14s_1 + 15s_2 + 5s_3 + 2s_4 &\approx 8 \ (\mod 17) \\ 14s_1 + 15s_2 + 5s_3 + 2s_4 &\approx 8 \ (\mod 17) \\ 14s_1 + 15s_2 + 5s_3 + 2s_4 &\approx 8 \ (\mod 17) \\ &\ldots \end{aligned} \tag{2.1}$$

Se non fosse per l'errore recuperare s sarebbe facile, infatti dopo circa n equazioni, potremmo usare l'Eliminazione Gaussiana ed in tempo polinomiale avere il vettore segreto incognito. Ma introdurre l'errore rende il problema significativamente più difficile. Infatti usando l'eliminazione Gaussiana non preoccupandoci degli errori, amplificherebbe questi non lasciando residui dell'informazione originale.

Capitolo 3

Risoluzione di problema LWE: studio di algoritmo BKW

Di seguito analizzeremo l'algoritmo proposto da Blum, Kalai, Wasserman, da cui prende il nome BKW, che risulta essere il miglior algoritmo al momento per la risoluzione di LWE.

Sia $N(\mu, \sigma^2)$ Gaussiana di media μ e deviazione standard σ, vogliamo il wrapping di questa in modulo q per poi discretizzarla in $\mathbb{Z}$ per adattarla al problema di LWE in $\mathbb{Z}_q$. Per wrap di $N(0, \sigma^2)$, denotiamo con $p(\phi)$ la funzione di densità di probabilità determinata da σ, definiamo $\theta := \phi \mod q$ e $p'(\theta)$ come

$$p'(\theta) = \sum_{k=-\infty}^{\infty} p(\theta + qk) \; per \; -q/2 < \theta \leq q/2 \tag{3.1}$$

Visto che all'aumentare di $|k|$ il contributo di $p(\theta + kq)$ diminuisce rapidamente, possiamo selezionare un taglio di $p'(\theta)$ e lavorare con questa approssimazione. Denotiamo allora con $\chi_{\alpha,q}$ la distribuzione in accordo a p' e arrotondata all'intero più vicino nell'intervallo $]-q/2, q/2]$, dove $\sigma = \alpha q$. Ottenendo:

$$Pr[X = x] = \int_{x-\frac{1}{2}}^{x+\frac{1}{2}} p'(t)dt \tag{3.2}$$

dove questa nel nostro caso può essere calcolata esplicitamente in quanto sia q che k sono polinomiali in n.

Algoritmo BKW fu inizialmente proposto come metodo per risolvere problema LPN. Questo può essere adattato ad search-decision LWE, con com-

plessità $2^O(n)$ quando il modulo è scelto polinomiale in n. Algoritmo BKW adattato a versione Search di LWE può essere visto come composto di tre parti, analoghe alle parti di cui è scomposta la risoluzione di un sistema lineare.

sample reduction Simile all'eliminazione Gaussiana ma invece di considerare ogni componente indipendentemente, si considerano blocchi di b componenti ad ogni iterazione.

hypothesis testing test di candidati come parte di soluzione al fine di recuperare componenti di vettore segreto s.

back substitution al fine di poter continuare processo ma su istanza di LWE ridotta.

Il motivo per cui non é possibile usare l'eliminazione Gaussiana così come la conosciamo risiede nel fatto che le addizioni di righe che eseguiamo nel tentativo di ridurre la matrice a forma triangolare, amplificano l'errore a tal punto che recuperare s risulta impossibile. Quindi lo logica su cui é basata BKW è quella di usare un maggior numero di righe ma eliminare più variabili contemporaneamente con singole addizioni di righe. Se riusciamo ad effettuare una riduzione del genere usando poche operazioni, l' errore risultante sarà piccolo a sufficienza da permetterci di recuperare parte di s ad ogni iterazione di algoritmo.

3.1 Sample reduction

Dato $n \in \mathbb{Z}$, si selezioni un intero positivo $b \leq n$, e sia $a := \lceil n/b \rceil$. Dato un oracolo che denotiamo con $L_{s,\chi}$, denotiamo un oracolo $B_{s,\chi,l}$ ad esso correlato, che restituisce in output campioni dove le prime $b \cdot l$ coordinate di ogni a_i sono zero, generati secondo la seguente distribuzione:

- se $l = 0$ allora $B_{s,\chi,0}$ è semplicemente $L_{s,\chi}$;
- se $0 < l < a$ allora la distribuzione $B_{s,\chi,l}$ è ottenuta attraverso la differenza di due vettori da $B_{s,\chi,l-1}$, in cui le ultime $n - b \cdot l$ coordinate sono quelle non nulle.

Possiamo quindi descrivere la prima parte di BKW come una serie (costruita ricorsivamente) di oracoli $B_{s,\chi,l}$, con $0 \leq l < a$. Faremo uso di un

insieme di tabelle T (che vengono mantenute nel corso delle chiamate agli oracoli) per memorizzare vettori che verranno usati per ridurre i campioni che derivano dagli oracoli. Più precisamente:

1. Per $l = 0$ possiamo ottenere campioni da $B_{s,\chi,l}$ semplicemente chiamando l'oracolo LWE $L_{s,\chi}$ e dando in output il suo output.

2. Per $l = 1$, chiediamo fino a $(q^b - 1)/2$ (sfruttiamo la simemtria di $\mathbb{Z}_q$) campioni all' oracolo $B_{s,\chi,0}$, con distinti vettori non nulli per le prime b cordinate di a. Usiamo questi campioni per popoplare la tabella T^1, indicizzata da prime b cordinate di a. Memorizziamo (a, c) in tabella. Durante il corso della popolazione della tabella se incontriamo un campione (a', c') cui prime b cordinate combaciano con un campione già in tabella (o sua negazione), diamo in output $(a' \pm a, c' \pm c)$ come campione di $B_{s,\chi,1}$. Chiamate sucessive a oracolo $B_{s,\chi,1}$ procedono in maniera simile ma usando la stessa tabella T^1.

3. Per $1 < l < a$, procediamo come sopra ma usando tabella T^l, costruita chiamando fino a $(q^b - 1)/2$ l'oracolo $B_{s,\chi,l-1}$ per ridurre qualsiasi campione in output da $B_{s,\chi,l-1}$ avente le b cordinate nel l-esimo blocco già in T^l, per generare campione da $B_{s,\chi,l}$.

Quindi dato oracolo LWE $L_{s,\chi}$, che da in output campioni della forma $(a, \langle a, s\rangle + e)$ dove $a, s \in \mathbb{Z}_q^n$, l'oracolo $B_{s,\chi,a-1}$ può essere visto come un altro oracolo LWE che restituisce in output campioni della forma $(a', \langle a', s'\rangle + e')$ dove $a', s' \in \mathbb{Z}_q^n$, con $k = n \mod b$ se k non divide n, e con $k = b$ altrimenti. Mentre e è generato con una distribuzione differente da quella originale, ma correlata ad esso. Il vettore s' rappresenta le ultime k componenti di s.

$T^l \leftarrow$ array indicizzato da $\mathbb{Z}_q^b$;
chiamata a $B_{s,\chi,l-1}$ per ottenere (a, c);
if $a_{(b\cdot(l-1),b\cdot l)}$ è vettore di tutti zero **then**
  return (a, c);
end if
while $T^l_{a_{(b\cdot(l-1),b\cdot l)}} = \emptyset$ and $T^l_{-a_{(b\cdot(l-1),b\cdot l)}} = \emptyset$ **do**
  $T^l_{a_{(b\cdot(l-1),b\cdot l)}} \leftarrow (a, c)$
  chiama $B_{s,\chi,l-1} perottenere(a, c)$;
  if $a_{(b\cdot(l-1),b\cdot l)}$ è vettore di tutti zero **then**
    return (a, c);
  end if
end while
if $T^l_{a_{(b\cdot(l-1),b\cdot l)}} \neq \emptyset$ **then**
  $(a', c') \leftarrow T^l_{a_{(b\cdot(l-1),b\cdot l)}}$
  return $(a - a', c - c')$;
else
  $(a', c') \leftarrow T^l_{-a_{(b\cdot(l-1),b\cdot l)}}$
  return $(a + a', c + c')$;
end if

Algorithm 1: Pseudocodice oracolo

3.2 Hypothesis Testing

Possiamo formulare il problema di risolvere LWE come il problema di distinguere tra due possibili distribuzioni. Dati m campioni in $\mathbb{Z}_q^d \times \mathbb{Z}_q$ da $B_{s,\chi,a}$, segue che abbiamo $\mathbb{Z}_q^d$ ipotesi da testare. Esamineremo ogni ipotesi da cui deriveremo un set di valori di errore come risultato (uno per campione). Se l'ipotesi s', sotto vettore di s è incorretta la distribuzione di questi valori di errore sarà distribuita quasi uniforme, mentre se l'ipotesi è corretta saranno distribuiti secondo χ_a.

Notare che se se la distribuzione degli errori associata ai campioni ottenuti da $L_{s,\chi}$ è $\chi_{\alpha,q}$, allora tenendo conto che la somma di v variabili indipendenti che seguono la distribuzione Gaussiana danno origine ad una variabile che segue la distribuzione Gaussiana con varianza $v\cdot\sigma^2$ e media $v\cdot\mu$, segue che errori associati a campioni ottenuti da $B_{s,\chi,l}$ seguono distribuzione $\chi_{\sqrt{2^l},q}$, infatti aggiungiamo 2^l Gaussiane discrete (ad ogni operazione di somma/sottrazione

di vettori durante la fase di eliminazione, eseguiamo una somma/sottrazione di errori) producendo una Gaussiana discreta con deviazione standard incrementata di un fattore $\sqrt{2^l}$. Per semplcità denoteremo questa distribuzione con X_l. In poche parole assumiamo che l'oracolo finale $B_{s,\chi,a}$ restituisce campioni con errori che sono distribuiti con una deviazione standard maggiore dell'originale, che sono usati per testare le ipotesi.

Come strategia di testing delle ipotesi, possiamo pensare ai campioni restituiti da $B_{s,\chi,a}$ come dante origine ad equazioni della forma:

$$f_i = -c_i \pm j + \sum_{k=0}^{d-1} (a_i)_{(k)} x_{(k)} \; per \; 0 \leq j < q/2 \tag{3.3}$$

Dato un numero di questi campioni, per dare una stima di s', testiamo le q^d ipotesi per costruire un array di score S indicizzato dalle possibili combinazioni in $\mathbb{Z}_q^d$.

Una funzione W assegna pesi ad elementi in $\mathbb{Z}_q$ che rappresentano l'errore sotto l'ipotesi $s' = v$. Per ogni vettore ipotesi v, sommiamo gli errori pesati $W(-c_i + \sum_{k=0}^{d-1}(a_i)_{(k)} \cdot v_{(k)})$. Se W è scelta tale che il contatore S_v aumenta proporzionalmente alla verosimiglianza che v sia l'ipotesi esatta, allora il contatore $S_{s'}$ crescerà più rapidamente.

Require: F- set di m campioni che seguono $B_{s,\chi,a}$
Require: W - funzione peso che mappa membri di $\mathbb{Z}_q$ in $\mathbb{R}$

$S \leftarrow$ array indicizzato da Z_q^d
for $v \in \mathbb{Z}_{||}$ **do**
  $w_v \leftarrow \emptyset$
  for $f_i \in F$ **do**
    $j \leftarrow \langle a_i, v \rangle - c_i$
    $w_v \leftarrow w_v \cup W(j)$
  end for
  $S_v \leftarrow \sum_{w_i \in w_v} w_i/m$
end for
return S

Algorithm 2: Pseudocodice contatore

Tenendo conto che χ_a denota la distribuzione di errori se la scelta del vettore v è giusta, denotiamo ora con U_a la distribuzione di errori se la scelta di v è sbagliata ($v \neq s'$). Secondo il Lemma di Neyman-Pearons, il test più

potente per constatare se un insieme di campioni segue una di due distribuzione conosciute è il logaritmo del rapporto tra funzioni di verosimiglianza. Per $\lceil -q/2 \rceil \leq j \leq \lfloor q/2 \rfloor$, impostiamo:

$$W(j) := \log_2 \left(\frac{Pr[e \leftarrow \chi_a : e = j]}{Pr[e \leftarrow U_a : e = j]}\right) \tag{3.4}$$

Stabiliamo anche la relazione tra $\tilde{p_j}$ (probabilità che errore sia j se scelta di v è sbagliata) e p_j (probabilità che errore sia j se scelta di v è giusta). Data la scelta sbagliata v per s', per ogni elemento $f_i = -c_i + \sum_{k=0}^{d-1} (a_i)_{(k)} x_{(k)}$ con $c_i = \langle a_i, s' \rangle - e_i$, si può dimostrare che la probabilità di errore j di apparire è:

$$\tilde{p_j} := Pr[e \leftarrow U_a : e = j] = \frac{q^{d-1} - p_j}{q^d - 1} \tag{3.5}$$

se q è primo.

Per stage finale di backsubtitution, dobbiamo assicurarci che lo score per la giusta scelta di $v = s'$ sia il più alto tra gli elementi di S. Quindi ci rimane da stabilire il numero $m = |F|$ di campioni necessari per far si che lo score per la scelta corretta di, sia il più alto. Sotto assunzione che campioni siano indipendenti, per il teorema del Limite Centrale, la distribuzione di S_v approssima la distribuzione Normale al crescere di m. Se $N(\mu, \sigma^2)$ denota la distribuzione Normale con media μ e deviazione standard σ, denotiamo la distribuzione per $v = s'$ con $D_c = N(E_c, Var_c)$ e per $v \neq s'$ con $D_w = N(E_w, Var_w)$. Usando le comuni formule per valore atteso e varianza possiamo ricavare:

$$\begin{aligned}
E_c &= E(S_v | v = s') = \sum_{j=\lceil -q/2 \rceil}^{\lfloor q/2 \rfloor} p_j w_j = p_0 w_0 + 2 \cdot \sum_{j=1}^{j=\lfloor q/2 \rfloor} p_j w_j \\
E_w &= E(S_v | v \neq s') = \sum_{j=\lceil -q/2 \rceil}^{\lceil q/2 \rceil \tilde{p}_j w_j} \\
Var_c &:= Var(S_v | v = s') = \frac{1}{m} \sum_{j=\lceil -q/2 \rceil}^{\lfloor q/2 \rfloor} p_j \cdot (w_j - E_c)^2 \\
Var_w &:= Var(S_v | v \neq s') = \frac{1}{m} \sum_{j=\lceil -q/2 \rceil}^{\lfloor q/2 \rfloor} \tilde{p}_j \cdot (w_j - E_w)^2
\end{aligned} \tag{3.6}$$

Denotiamo ora con Y_h la variabile casuale determinata dal rank di uno score corretto $S_{s'}$ in lista di h elementi. Per una lista di q^d elementi e un rank quindi $0 \leq r < q^d$, la probabilità che Y_{q^d} assuma rank r è data da una distribuzione di probabilità composta (normale-binomiale). Tenendo conto di questa e di E_c, E_W, Var_c, Var_w definiti come sopra, possiamo ricavare il numero di campioni m necessari per far si che Y_{q^d} abbia rank 0 con probabilità e' risolvendo:

$$e' = \int_x [\frac{1}{2}(1 + erf(\frac{x - E_w}{\sqrt{2Var_w}}))]^{q^d - 1} \cdot (\frac{1}{\sqrt{2_c}} e^{-\frac{(x-E_c)^2}{2Var_c}})dx \tag{3.7}$$

dove erf è la funzione d'errore di Gauss.

Un modo alternativo di scegliere una soluzione v è quella di continuare a chiedere campioni da $B_{s,\chi,a}$ finché la distanza tra prima e seconda ipotesi non supera un certo valore soglia.

3.3 Back Substitution

Una volta ottenuta una soluzione candidata per s', possiamo eseguire la sostituzione nelle tabelle T^i in modo simile a risolvere un sistema lineare triangolare. Dopo la back substitution, rieseguiamo l'algritmo BKW dallo stage uno, ma con tabelle T^i già riempite. Per recuperare le prossime b componenti di s chiediamo quindi m nuovi campioni che sono ridotti usando le tabelle modificate T^i ed eseguendo Hypothesis testing su questi m campioni.

Capitolo 4

Difficoltà di LWE

In questo capitolo verrà esposta una breve descrizione sulla difficoltà del problema LWE. Prima però una piccola premessa sulle basi necessarie.

Lattice

Un Lattice (reticolo) n-dimensionale Λ è ogni sott'insieme di $\mathbb{R}^n$ tale che è:

1. sottogruppo additivo: $0 \in \Lambda$, e $-x, x+y \in \Lambda$ per ogni $x, y \in \Lambda$
2. discreto: ogni $x \in \Lambda$ ha un intorno in $\mathbb{R}^n$ in cui x è unico punto del Lattice.

La distanza minima di un Lattice Λ è la lunghezza del vettore non nullo più corto del Lattice.

$$\lambda_1(\Lambda) := \min_{v \in \Lambda \setminus \{0\}} ||v|| \tag{4.1}$$

Un Lattice è generato da combinazione lineare di vettori lineari indipendenti con coefficienti interi, questi formano una base del Lattice $B = \{b_1, ..., b_k\}$.

$$\Lambda = \Lambda(B) := B \cdot \mathbb{Z}^k = \{\sum_{i=1}^{k} z_i b_i : z_i \in \mathbb{Z}\} \tag{4.2}$$

L'intero k è il rango della base. Tratteremo per semplicità solo Lattice che hanno rango massimo ($k = n$). La base B non è unica: per ogni matrice unimodulare[1] $U \in \mathbb{Z}^{n \times n}$, $B \cdot U$ è una base di $\Lambda(B)$ perchè $U \cdot \mathbb{Z}^n = \mathbb{Z}^n$.

[1] matrice con determinante ± 1

Il duale (o reciproco) di un Lattice $\Lambda \in \mathbb{R}^n$ è definito come :

$$\Lambda^* := \{w : \langle w, \Lambda \rangle \subseteq \mathbb{Z}\} \tag{4.3}$$

ovvero l'insieme di vettori di $\mathbb{R}^n$ cui prodotto scalare con i vettori in Λ, sono interi. Λ^* è un Lattice a sua volta. Esempio: $\mathbb{Z}^n$ è un Lattice, il suo duale è $\mathbb{Z}^n$ che è a sua volta un Lattice. Si può verificare anche che se B è la base di un Lattice Λ, allora $(B^{-1})^t$ è base di Λ^*.

Dato che Lattice è sottogruppo additivo di $\mathbb{R}^n$, abbiamo gruppo $\mathbb{R}^n/\Lambda$ con classi laterali:

$$c + \Lambda = \{c + v : v \in \Lambda\}, c \in \mathbb{R}^n \tag{4.4}$$

Distribuzione Gaussiana discreta

Definiamo la distribuzione Gaussiana discreta su Λ con parametro r, denotata da $D_{\Lambda,r}$, come la distribuzione di probabilità che assegna massa proporzionale a $e^{-\pi||x/r||^2}$ ad ogni punto del Lattice Λ. Campionamenti da distribuzione $D_{\Lambda,r}$ sono vettori del Lattice Λ con norma circa $\sqrt{n}r$.

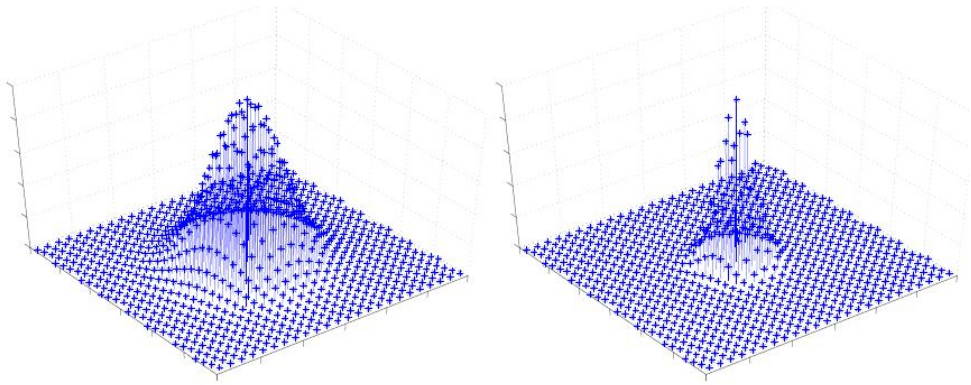

Figura 4.1: A sx $D_{\Lambda,2}$, a dx $D_{\Lambda,1}$ per Lattice bidimensionale. Asse z rappresenta probabilità

Più precisamente, per ogni intero positivo n e per ogni reale $s > 0$, definiamo la funzione Gaussiana $\rho_s : \mathbb{R}^n \to \mathbb{R}^+$ di larghezza s come:

$$\rho_s(X) := exp(-\pi||x||^2/s^2) = \rho(X/s) \tag{4.5}$$

La distribuzione Gaussiana continua D_s di larghezza s, su $\mathbb{R}^n$ è definita avere funzione di densità di probabilità proporzionale a ρ_s:

$$f(x) := \rho_s(s) / \int_{\mathbb{R}^n} \rho_s(z)dz = \rho_s(s)/s^n \tag{4.6}$$

Per $c + \Lambda \subset \mathbb{R}^n$ classe laterale di Lattice Λ, e larghezza s, la distribuzione di probabilità Gaussiana discreta $D_{c+\Lambda,s}$ è semplicemente la distribuzione Gaussiana ridotta alla classe laterale:

$$D_{c+\Lambda,s}(x) = \begin{cases} \rho_s(x) & \text{se } x \in c + \Lambda \\ 0 & \text{altrimenti} \end{cases} \tag{4.7}$$

Problemi geometrici su Lattice

Mentre problemi algebrici ,come decidere se un vettore è contenuto in Lattice o computare il duale di un Lattice, sono semplici, i problemi geometrici risultano invece generalmente difficili.

GapSVP$_\Upsilon$

Dato Lattice Λ è chiesto di approssimare $\lambda_1(\Lambda)$ entro un fattore moltiplicativo Υ. Problema è NP-Hard per Υ piccoli, e facile per fattori d'approssimazione esponenziali $\Upsilon = s^{O(n)}$. Nella letteratura crittografica Lattice-based l'interesse è comunque posto su fattori d'approssimazione polinomiali della forma $\Upsilon = n^c$ dove c è una costante tipicamente almeno 1. Per fattori d'approssimazione come questi il problema non è NP-hard ma è comunque considerato difficile. Infatti l'algoritmo più veloce al momento esegue in tempo $2^{O(n)}$, e anche computer quantistici sembrano non produrre miglioramenti.

SIVP$_\Upsilon$

Trovare un set di n vettori linearmente indipendenti in dato Lattice, tale che la lunghezza del vettore più lungo nel set sia al massimo Υ volte più lungo del più piccolo possibile per ogni set di questo tipo. Tutte le proprietà menzionate in GapSVP si applicano anche a SIVP.

BDD

Per certa distanza $d > 0$, dato Lattice Λ e punto x con distanza da Λ al più d, è chiesto di trovare il vettore del Lattice più vicino ad x. Fin tanto che $d < \lambda_1(\Lambda)/2$ la soluzione al problema è unica se esiste.

4.1 Hardness

La proposizione principale, stipulata da Regev, su cui si basa la difficoltà di LWE è la seguente:

> *Sia $q \geq 2$ intero e α numero reale in (0,1). Assumiamo di aver accesso ad Oracolo LWE con modulo q e parametro d'errore α. Dato come input qualsiasi Lattice Λ, un numero polinomiale abbastanza grande di campioni da distribuzione Gaussiana discreta $D_{\Lambda^*,r}$, per r non troppo piccolo, e punto x con distanza da Λ al più $\alpha q/(\sqrt{2}r)$, allora possiamo calcolare (l'unico) punto del Lattice più vicino ad x in tempo polinomiale.*

Per capire l'importanza di tale proposizione è utile metterla in contrasto con un altro risultato ottenuto da Y.-K. Liu, Lyubashevsky e Micciancio oltre che da Regev e Aharonov:

> *Dato come input qualsiasi Lattice Λ, un numero polinomiale abbastanza grande di campioni da distribuzione Gaussiana discreta $D_{\Lambda^*,r}$, per r non troppo piccolo, e punto x con distanza da Λ al più $O(\sqrt{\log n}/r$, possiamo calcolare (l'unico) punto del Lattice più vicino ad x in tempo polinomiale.*

Si noti che nessun oracolo LWE è stato utilizzato in quest'ultimo algoritmo. Quindi la proposizione di Regev mostra che usando oracolo LWE è possibile allargare il raggio di decoding da $O(\sqrt{\log n}/r$ a $\alpha q/(\sqrt{2}r)$. Questo può già essere interpretato come un indicazione che il problema LWE è difficle, in quanto questo ci permetterebbe di risolvere un problema worst-case in contesto Lattice (BDD dato indizio su forma di campioni da distribuzione Gaussiana discreta) che altrimenti non sapremmo come risolvere. Più nello specifico per $\alpha q = \Omega(\sqrt{n})$ il miglior algoritmo che abbiamo per risolvere il problema richiede tempo esponenziale. In altre parole se possiamo risolvere problema BDD usando LWE, questo non può essere più semplice di BDD.

Sarebbe preferibile comunque relazionare LWE a problemi Lattice-based più standard. Tale risultato fu ottenuto da Peiker. Essi scoprì una riduzione in tempo polinomiale da problema GapSVP$_\Upsilon$ a BDD (con distanza inferiore a $\lambda_1/poly(n)$).Questo ci dice che finché $\alpha q/r <= \lambda_1(\Lambda)/poly(n)$, LWE risulta difficile almeno quanto la variante worst-case di problema GapSVP in cui abbiamo campioni da $D_{\Lambda^*,r}$

L'approccio originale usato da Regev è differente. Regev dimostrò che per ogni $d > 0$ esiste un efficiente riduzione quantistica dal problema di campionare da $D^*_\Lambda, \sqrt{n}/d$ al problema di risolvere BDD su Λ con distanza d. Combinando questa riduzione con proposizioni viste, si può mostrare che risolvere LWE con modulo q polinomiale implica una soluzione quantistica a problemi worst-case Lattice-based standard.

Più precisamente assumiamo che $\alpha q \geq 2\sqrt{n}$. Iniziamo con creare alcuni campioni da $D_{\Lambda^*,r}$ per r grande (che può essere fatto efficientemente). Ora usiamo la proposizione per ottenere soluzione a BDD su Λ con distanza $\sqrt{2n}/r$. Ora si applica la riduzione quantistica per ottenere campioni da $D_{\Lambda^*,r/\sqrt{2}}$. Ora si itera i precedenti due step per ottenere campioni da $D_{\Lambda^*,r/2}, D_{\Lambda^*,r/2\sqrt{2}}$,.. Dopo un numero polinomiale di steps, otteniamo campioni da $D_{\Lambda^*,r'}$ per un piccolo $r' = poly(n)/\lambda_1(\Lambda)$. Ora ripetendo ancora una volta lo step 1 e utilizzando la riduzione da GapSVP a BDD è possibile ottenere una soluzione per GapSVP.

Alternativamente prelevando circa n campioni da $D_{\Lambda^*,r'}$ otteniamo una soluzione per problema SIVP.

Diamo ora cenno sulla prova del teorema:

Prendiamo per semplicità $\mathbb{Z}^n$ come Lattice. Assumiamo di avere punto x vicino a punto di Lattice v a noi sconosciuto. Possiamo creare campioni di LWE usando $s = v \mod q$ come segreto. Usando oracolo LWE possiamo recuperare s, che ci dà la cigra meno significativa di v in base q. Per recuperare l'intero vettore possiamo ripetere il processo su punto $(x - s)/q$, che è vicino a punto su Lattice (v-s)/q, e usando ancora oracolo LWE per ottenere questa volta la seconda cifra di v in base q. Procediamo fino ad ottenere tutto il vettore v.

Il cuore della prova tuttavia risiede nel produrre campioni LWE con segreto s. Si prende campione da $D_{\mathbb{Z}^n,r}$ e si restituisce in ouput:

$$(a = y \mod q, b = \lfloor \langle y, x \rangle \rceil \mod q) \in \mathbb{Z}_q^n \times \mathbb{Z}_q \qquad (4.8)$$

Visto che r non troppo piccolo, la distribuzione di a è essenzialmente uniforme in $\mathbb{Z}_q^n$ come richiesto. Una volta fissato a, visto che y è distribuito in accordo a distribuzione Gaussiana discreta su set di punti $q\mathbb{Z} + a$. Notare che se $x = v$ allora b è esattamente $\langle a, s \rangle$, corrispondente a equazione LWE senza errori. Nel caso in cui $x = v + e$ per qualche vettore e di norma al più $\alpha q/(\sqrt{2}r)$, otteniamo un termine di errore in seconda coordinata della forma $\langle e, y \rangle$. Essendo prodotto scalare di vettore fissato di tale norma, con vettore ottenuto da distribuzione Gaussiana discreta che ha al più norma r, l'errore è essenzialmente distribuito normalmente con deviazione standard al più αq, come richiesto.

Capitolo 5

Ring-LWE

Schemi crittografici basati su LWE richiederebbero uno spazio delle chiavi troppo grande, tipicamente dell'ordine n^2. Questo perché per applicazioni pratiche, tipicamente si necessita di almeno n vettori $a_1, .., a_n \in \mathbb{Z}_q^n$. Ridurre tale ad almeno grandezza lineare è desiderabile da un punto di vista pratico. Un modo tipico di raggiungere tale obiettivo è assumere un qualche tipo di struttura nei campioni LWE. Un esempio di struttura tipicamente usato è il seguente:

> Assumiamo che n sia una potenza di 2 e assumiamo che vettori a arrivino in blocchi di n campioni $a_1, .., a_n \in \mathbb{Z}_q^n$, dove $a_1 = (a_1, .., a_n)$ è scelto uniformemente come sempre, e i rimanenti vettori sono dati come $ai = (a_i, ..., a_n, -a_1, .., -a_{i-1})$. In questo modo per rapresentare n vettori ora sono necessari solo $O(n)$ elementi di $\mathbb{Z}_q$.

Matematicamente parlando, quanto esposto può essere interpretato come la sostituzione di $\mathbb{Z}_q^n$ con $\mathbb{Z}_q[x]/(x^n + 1)$, ovvero l'anello dei polinomi con coefficenti interi modulo q di grado al più $n - 1$, dove ad ogni campione di R-LWE corrispondono n campioni di LWE. Altre scelte di Anelli sono possibili ma i più usati sono forse quelli di questo tipo. Il requisito che n sia una potenza di 2 serve per garantire che $x^n + 1$ sia non riducibile sui numeri razionali. Questo perché per alcune versioni del problema, se questa condizione non è verificata, il problema risulta più facile e la ragione è da attribuire alla scomposizione in fattori del polinomio $x^n + 1$.

La domanda ora è: questa versione di LWE è difficile a sua volta?
Stehlé, Steinfeld, Tanaka e Xagawa hanno dimostrato una riduzione quantistica da ring-SIS a ring-LWE. Ring-SIS é la versione su anello del problema

SIS (Short Integer Solution). Questo può essere visto come il problema duale LWE. La versione originaria di SIS può essere così definito:

> Dati una sequenza di vettori $a_1, a_2, ..$ scelti uniformemente da $\mathbb{Z}_q^n$ ed é chiesto di trovare una combinazione di questi con coefficienti piccoli che diano come risultato 0 in modulo q. Equivalentemente può essere visto anche come il problema di trovare vettori piccoli in Lattice casuale. Si può dimostrare che per q almeno polinomiale in n, risolvere SIS implica una soluzione a problemi Lattice-based come SIVP e GapSVP.

Si può dimostrare a sua volta che la risoluzione di Ring-SIS implica a sua volta la soluzione a problemi worst-case Lattice-based, anche se ristretti alla famiglia dei Lattice ideali che sono Lattice con struttura algebrica aggiuntiva. È ragionevole pensare che problemi Lattice-based su tali Lattice ideali siano difficili dato che non è ancora noto un metodo per trarre vantaggio di tale struttura.

La riduzione da Ring-LWE a Ring-SIS ha però il difetto di dipendere dal numero di campioni LWE m. Si noti che in LWE standard il numero di campioni è del tutto insignificante per quanto riguarda la robustezza del problema.

Lyubashevsky, Peikert, e Regev trovarono un altro risultato riguardante la difficoltà di Ring-LWE, basato sugli stessi principi su cui era basata la dimostrazione per la riduzione da BDD a LWE di Regev, stipulando che risolvere Ring-LWE sotto determinate condizioni risulta essere difficile almeno quanto risolvere SVP_Υ su Lattice ideale arbitrario, con $\Upsilon = poly(n)/\alpha$ con $\alpha < 1$.

Capitolo 6

Costruzioni Crittografiche

Qui di seguito verranno esposte alcune delle possibili implementazioni crittografiche costruite su (Ring-)LWE.

6.1 Regev's LWE cryptosystem

Regev propose il primo schema crittografico a chiave pubblica basato su LWE, dove le chiavi pubbliche sono $O(n^2)$ bit, mentre ciphertext e chiavi segrete sono $O(n)$ bit. Ogni ciphertext contiene un singolo bit di informazione da crittografare. Vengono fissati n , modulo q ,χ su $\mathbb{Z}$ come distribuzione errori , e m numero di campioni, come parametri del problema LWE, scelti in modo da soddisfare varie condizioni di sicurezza e di corretta decriptazione del messaggio.

chiavi Chiave segreta è segreto s di LWE scelto uniformemente su $\mathbb{Z}_q^n$. La chiave pubblica è data da $m \approx (n+1)\log q$ campioni della forma $(\hat{a}_i, b_i = \langle s, \hat{a}_i \rangle + e_i) \in \mathbb{Z}_q^{n+1}$ ottenuti da distribuzione LWE $A_{s,\chi}$ e trattati come vettori colonna della matrice A:

$$A = \begin{bmatrix} \hat{A} \\ b^t \end{bmatrix} \in \mathbb{Z}_q^{(n+1)\times m} \tag{6.1}$$

dove $b^t = s^t\hat{A} + e^t \mod q$. In contesto multi-utente, $\hat{A}$ può essere condiviso tra tutti gli utenti e la chiave pubblica dell'utente risulta solo b. Si noti che per definizione vale la seguente relazione:

$$(-s, 1)^t \cdot A = e^t \approx 0 (mod\ q) \tag{6.2}$$

encription Per criptare un bit μ usando la chiave pubblica A, si sceglie $x \in \{0,1\}^m$ uniformemente a caso e si restituisce in output:

$$c = A \cdot x + (0, \mu \cdot \lfloor \frac{q}{2} \rceil) \in \mathbb{Z}_q^{n+1} \tag{6.3}$$

decrypt Per decriptarre usando la chiave privata s, si calcola:

$$\begin{aligned} (-s,1)^t \cdot c &= (-s,1)^t \cdot A \cdot x + \mu \cdot \lfloor \frac{q}{2} \rceil \\ &= e^t \cdot x + \mu \lfloor \frac{q}{2} \rceil \\ &\approx \mu \cdot \lfloor \frac{q}{2} \rceil (mod\ q) \end{aligned} \tag{6.4}$$

e si testa se è più vicino a 0 o a $\lfloor \frac{q}{2} \rceil$ modulo q. La decriptazione del messaggio è corretta fin tanto che l'errore accumulato $\langle e, x \rangle \in \mathbb{Z}$ ha valore assoluto minore di $q/4$. Questo può essere ottenuto semplicemente scegliendo q grande abbastanza in relazione alla distribuzione di errore χ e il numero di campioni m.

Sicurezza

Questo sistema è sicuro contro un intercettatore passivo assumendo che la versione Decision-LWE sia difficile (si può dimostreare che esiste riduzione da Decision a Search LWE, quindi Decision almeno difficile quanto Search LWE). Le due idee principali sono le seguenti:

1. una chiave pubblica propriamente formata A è indistinguibile da una malformata (scelta uniformemente a caso)

2. Cifrare con tale chiave malformata è sicuro a livello informativo.

Ricordiamo che siamo in versione decisionale quindi vogliamo mostrare che non è possibile distinguere tra (A,c) ben formati o mal formati. Procediamo con serie di esperimenti che producono A, c in modi differenti:

- Nel primo esperimento la chiave pubblica A è scelta uniformemente da $\mathbb{Z}_q^{(n+1)\times m}$, invece di essere generata da campioni LWE (quindi non c'è chiave segreta corrispondente). Cyphertext c creato calcolandolo nella maniera classica usando A chiave malformata e μ bit da cifrare. Questo esperimento è indistinguibile da quello reale sotto assunzioni LWE.

Questo può essere mostrato con riduzione: un ipotetico attaccante B che mira a distinguere tra i due esperimenti può essere trasformato in algoritmo D che mira a distinguire tra campioni LWE e quelli scelti uniformemente a caso. Attacca decision-LWE: D semplicemente colleziona campioni di input in matrice A, cifra μ usando A per ottenere cyphertext c e lo passa in input ad attaccante B, restituendo in output la stessa risposta. È chiaro che D simula perfettamente l'esperimento reale o l'altro a seconda che campione di input sia da distribuzione LWE o da quella uniforme (rispettivamente). Quindi D e A hanno stesso vantaggio di poter distinguere tra i due. Visto che per ipotesi il vantaggio di D deve essere insignificante così sarà quello di B.

- Nel secondo esperimento la chiave pubblica A è ancora scelta uniformemente a caso, ma ora anche il cyphertext c è scelto uniformemente a caso e indipendentemente da A. Possiamo affermare che l'esperimento è statisticamente indistinguibile dal precendente, infatti anche un attaccante con potenza di calcolo infinita ha solo un vantaggio trascurabile nel distinguerli. Ciò perché si può dimostrare che per m sufficientemente grande, con $(A, u = A \cdot x)$ per $A \leftarrow \mathbb{Z}_q^{(n+1)\times m}$ uniforme , e per $x \leftarrow \{0,1\}^m$, i campioni risultano statisticamente indistinguibile da campioni uniformemente a caso grazie ad un lemma stipulato da R. Impagliazzo, L. Levin, and M.Luby chiamato "lefthover hash lemma" . Si noti che aggiungere vettore $(0, \mu \cdot \lfloor \frac{q}{2} \rceil)$ ad u ne preserva la distribuzione uniforme.

In conclusione dato che gli esperimenti sono indistinguibili per ogni bit μ fissato, e visto che l'ultimo esperimento non dipende da μ affatto, gli esperimenti reali per $\mu = 0, 1$ sono anch'essi indistinguibili. Nonostante ciò il sistema rimane comunque vulnerabile ad attacchi di tipo CCA in quanto risulta sicuro solo ad attacchi CPA. Micciancio e Peikert hanno in seguito creato un sistema basato su LWE sicuro anche da attacchi di tipo CCA.

6.2 Ring-LWE Cryptosystem

Questo sistema è stato introdotto da Lyubashevsky, Peikert e Regev, basandosi su anelli $R = \mathbb{Z}[X]/(X^n + 1)$ con n potenza di due, e in futuro esteso ad altri anelli.

chiavi chiave pubblica è campione Ring-LWE della forma:

$$(a, b \approx s \cdot a) \in R_q \times R_q \tag{6.5}$$

per $a \in R_q$ scelto uniformemente a caso, e per $s \in R$ chiave privata, dove s ed errore sono tratti da distribuzione χ.

encription per cifrare messaggio $\mu \in R_2$ (stringa di n bit), si genera il ciphertext come:

$$(u \approx a \cdot r, v \approx b \cdot r + \mu \cdot \lfloor \frac{q}{2} \rceil) \in R_q \times R_q \tag{6.6}$$

dove $r \in R$ ed errore sono tratti da χ.

decryption per decifrare il messaggio, si calcola:

$$v - s \cdot u \approx \mu \cdot \lfloor \frac{q}{2} \rceil + b \cdot r - s \cdot a \cdot r \approx \mu \cdot \lfloor \frac{q}{2} \rceil \tag{6.7}$$

ricavando μ rimuovendo errore di approssimazione.

6.3 R-LWE signature

Il primo schema R-LWE per la firma digitale fu sviluppato da Lyubashevsky nel 2009 e ridefinito in seguito. Nel 2012 Lyubashevsky , Gunesyu e Popplemann pubblicarono un paper in cui fu definito lo schema per firma digitale conosciuto come GLP. Qui di seguito useremo GLP come esempio di schema di firma digitale basato su LWE. Lo schema lavora su anello $R = \mathbb{Z}_q [x] / (x^n + 1)$. La firma GLP usa polinomi che sono piccoli rispetto alla norma infinito (il coefficiente più grande in valore assoluto del polinomio). Per ottenere polinomi con norma infinito entro un certo valore soglia, si usa la distribuzione uniforme su set di valori interi $[-b, b]$ dove $b < q$, per i coefficienti del polinomio da generare. GLP richiede un algoritmo per trasformare stringhe di b bit in polinomi piccoli, con norma entro un certo valore b con certa distribuzione e che abbia k coefficienti diversi da 0 . GLP richiede anche l'uso di un processo chiamato "rejection sampling". In questa tecnica se la norma infinito di un polinomio supera una certa soglia β viene scartato e il processo di firma ricomincia da capo. Tale processo serve per far si che la firma in output non sia correlabile alla chiave segreta del firmatario. Per GLP $\beta = (b - k)$, dove b è il range della distribuzione uniforme e k il numero di coefficienti non zero in polinomio.

setup Si definiscono i seguenti:

- **R** anello di polinomi a coefficienti interi modulo q, con q primo e n potenza di due.
- **a** polinomio fissato di R
- **n** grado massimo dei polinomi
- **b** valore assoluto massimo dei coefficienti del polinomio
- **k** numero di coefficienti del polinomio diversi da zero.

possibili valori possono essere:

$$n = 1024, q = 59393, b = 16383, k = 16 \tag{6.8}$$

La sicurezza dello schema dipende anche da questi parametri.

Il polinomio a dovrebbe essere scelto in modo che possa essere fidato da tutti i firmatari e verificatori. Un modo di ovviare a ciò è prendendo una stringa di bit casuale pubblica, che verrà usata come input per generare il polinomio a attraverso funzione one-way. Se la funzione è sicura e la stringa è pubblica, gli utilizzatori possono usare a in sicurezza.

generazione chiavi Entità che vuole firmare messaggio conoscendo R, a, n, q, b, k genera la propria chiave pubblica:

- genera due polinomi piccoli s, e con norma al massimo b come specificato prima, che saranno chiave privata.
- calcola $t = a \cdot s + e$
- pubblica t come sua chiave pubblica

Come si può notare la sicurezza dello schema è basata sul problema R-LWE.

firma Per firmare messaggio m:

1. si genera due polinomi piccoli y_1, y_2
2. si calcola $w = a \cdot y_1 + y_2$
3. si mappa w in stringa di bit $\hat{w}$
4. si calcola $c = F((Hash(\hat{w}|m))$ dove c è polinomio piccolo con k coefficienti non nulli.

5. si calcola $z_1 = s \cdot c + y_1$
6. si calcola $z_2 = e \cdot c + y_2$
7. fin tanto che le norme (infinito) di z_1 e z_2 non sono $\leq \beta$, andare a step 1.
8. la firma è la tripla di polinomi (c, z_1, z_2)
9. si trasmette il messaggio m con tripla di polinomi.

verifica della firma Per verificare messaggio m espresso come stringa di bit, il verificatore deve essere in possesso di:

- parametri dello schema R, a, n, q, b, k
- chiave pubblica t del firmatario
- firma (c, z_1, z_2)
- messagio m

Per verificare la firma esegue:

1. si verifica che la norma di z_1, z_2 sia minore di β, se così non è si rifiuta la firma.
2. si computa $w' = a \cdot z_1 + z_2 - t \cdot c$
3. si mappa w' in stringa di bit $\hat{w'}$
4. si calcola $c' = F(Hash(\hat{w'}|m))$
5. se $c' \neq c$ si rifiuta la firma, altrimenti si accetta la firma come valida.

6.4 R-LWE key exchange

Il primo algoritmo di scambio di chiavi fu descritto da Ding, Xie, Lin nel 2012. Seguirono altri algoritmi basati sui lavori di Peikert. Il setup è simile di quello definito per la firma digitale con R-LWE definito sopra. L'algoritmo genererà polinomi piccoli nel rispetto della norma infinito, e che abbiano quindi coefficienti di al più valore assoluto b. Per raggiungere tale scopo i coefficienti possono essere ottenuti da distribuzione uniforme su insieme di valori interi come $[-b, b]$ o possono essere ottenuti da distribuzione normale discreta centrata in 0 con parametro σ oer far si che i valori dei coefficienti

rientrino in $[-(q-1)/2, (q-1)/2]$. Un altro modo di ottenere ciò è attraverso una distribuzione binomiale discreta.

setup Siano definiti i seguenti parametri:

- **R** anello di polinomi a coefficienti interi modulo q, con q primo e n potenza di due.
- **a** polinomio fissato in R
- **n** grado massimo dei polinomi
- parametri per distribuzione usata per i coefficienti

Ancora una volta la sicurezza dello schema dipende anche dalla scelta dei parametri e la scelta di a dovrebbe essere fatta in modo che possa essere verificata da tutti i partecipanti allo scambio di chiavi.

scambio di chiavi Chiamiamo le due parti **A** e **B**, e definiti i parametri dello schema R, a, n, q, b, ogni parte genera la propria chiave pubblica nel seguente modo:

- si creano due polinomi piccoli s_i, e_i con coefficienti scelti (in questo caso di esempio) uniformemente dal set di interi $[-b, b]$, che costitueranno la chiave privata
- si calcola $t_i = a \cdot s_i + e_i$, che sarà la chiave pubblica

Quindi la parte A sarà in possesso di s_a, e_a, t_a mentre la parte B avrà s_b, e_b, t_b.

1. parte A invia t_a a parte B.
2. parte B allora calcola $v = t_a \cdot s_b = a \cdot s_a \cdot s_b + e_a \cdot s_b$

Si noti che $s_a \cdot e_b$ non è uguale a $s_b \cdot e_a$ anche se questi sono il prodotto di polinomi piccoli. In step successivo di algoritmo, B calcola un vettore di riconciliazione che invierà ad A e che permetterà ad A e B di arrivare a stesso segreto condiviso. Un modo di effettuare ciò è il seguente (stesso passo effettuato in New Hope, un algoritmo di scambio di chiavi):

- B crea 256 tuple di 4 elementi da coefficenti di v separati da 256 elementi. Esempio: se prima tupla è $v_0, v_{256}, v_{512}, v_{768}$), la seconda sarà $(v_1, v_{257}, v_{513}, v_{769})$e così via.

- sia g vettore (1/2,1/2,1/2,1/2), h il vettore $(0, 0, 0, 2)$ e b un bit a caso. Sia L un Lattice con i seguenti quattro vettori come base:

$$\begin{aligned} &(1, 0, 0, 0) \\ &(0, 1, 0, 0) \\ &(0, 0, 1, 0) \\ &(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}) \end{aligned} \tag{6.9}$$

  Per ogni tupla di quattro elementi considerata come vettore x con elementi da $\mathbb{Z}_q$, B cerca il vettore più vicino a $(4/q)x + 4bg - bhmod4$ in Lattice L.Gli elementi di questo vettore sono usati come coefficienti in polinomio r di grado 1024, in stessa posizione in cui tupla di coefficienti era stata estratta in precedenza da v.

- B forma $c = 2^{14} \cdot r + t_b$ e invia c ad A

A riceve da B polinomio C e può ricostruire facilmente t_b e r. A forma anche una matrice D da trasposizione di quattro vettori base usati per il Lattice. Così parte A ora esegue:

- crea $v' = t_b \cdot s_a = a \cdot s_b \cdot s_a + e_b \cdot s_a$
- crea 256 tuple di 4 elementi da coefficienti di v' e 256 tuple di quattro elementi da coefficienti di r usando stesso schema che B ha usato in precedenza.
- per ogni set di tuple di v' e r, $f = (v_i, v_{i+256}, v_{i+512}, v_{i+768})$ e $g = (r_i, r_{i+256}, r_{i+512}, r_{i+768})$, A forma $k = f - (q/4) \cdot Dg$ e poi calcola vettore u con ogni elemento $u_i = k/q - \lfloor (k_i/q) \rceil$. Se la somma dei valori assoluti dei valori degli elementi di u sono minori di 1 allora un bit di chiave è posto ad 1. Se la somma è maggiore di 1 il bit di chiave è posto a 0. Ripetendo questo procedimento per i 256 set di tuple, la parte A crea una chiave da 256 bit.

Dopo aver calcolato il vettore r di riconciliazione, B esegue gli stessi passi su polinomio v che A ha eseguito su polinomio v'. Così le parti A e B calcoleranno la stessa chiave di 256 bit con probabilità molto elevata.

Bibliografia

[1] R.Kübler A.May C.Sohler A.Esser F.Heuer. *Dissection BKW*. Youtube. 2018. URL: https://www.youtube.com/watch?v=3Or3Puh7OOM.

[2] Martin Albrecht. *01 Albrecht on The BKW Algorithm and Variants for Solving LWE*. Youtube. 2015. URL: https://www.youtube.com/watch?v=LUZAedRx-Ds.

[3] Martin Albrecht et al. «On the complexity of the BKW algorithm on LWE». In: *Designs, Codes and Cryptography* 74.2 (feb. 2015), p. 26. DOI: 10.1007/s10623-013-9864-x. URL: https://hal.inria.fr/hal-00921517.

[4] Martin R. Albrecht. «Solving LWE with BKW». In: (nov. 2013).

[5] Chuck Easttom. «An Analysis of Leading Lattice-Based Asymmetric Cryptographic Primitives». In: *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*. 2019, pp. 0811–0818. DOI: 10.1109/CCWC.2019.8666459.

[6] Daniele Micciancio. *The Relation Between SIS and LWE*. Youtube. 2020. URL: https://www.youtube.com/watch?v=H3trFEeJ2Wg.

[7] Daniele Micciancio e Oded Regev. «Lattice-based Cryptography». In: *Post-Quantum Cryptography*. A cura di Daniel J. Bernstein, Johannes Buchmann e Erik Dahmen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 147–191. ISBN: 978-3-540-88702-7. DOI: 10.1007/978-3-540-88702-7_5. URL: https://doi.org/10.1007/978-3-540-88702-7_5.

[8] Chris Peikert. «A Decade of Lattice Cryptography». In: *Foundations and Trends® in Theoretical Computer Science* 10.4 (2016), pp. 283–424. ISSN: 1551-305X. DOI: 10.1561/0400000074. URL: http://dx.doi.org/10.1561/0400000074.

[9] Chris Peikert. *Algebraic Lattices and Ring Learning with Errors*. Youtube. 2020. URL: https://www.youtube.com/watch?v=Lo-_ZBqGa7I.

[10] Oded Regev. «On Lattices, Learning with Errors, Random Linear Codes, and Cryptography». In: vol. 56. Gen. 2005, pp. 84–93. DOI: 10.1145/1568318.1568324.

[11] Oded Regev. «The Learning with Errors Problem (Invited Survey)». In: *2010 IEEE 25th Annual Conference on Computational Complexity*. 2010, pp. 191–204. DOI: 10.1109/CCC.2010.26.

[12] Damien Stehlè. «The LWE problem from lattices to cryptography». In: (giu. 2015).