

12.2 Criptografía basada en retículas

Def: Una retícula L se define como el conj. de combinaciones lineales enteras de un número de vectores independientes a_1, a_2, \dots, a_n (base) y con coef. enteros $x_i \in \mathbb{Z}$:

$$L = \{x_1 a_1 + x_2 a_2 + \dots + x_n a_n\}$$

$$\dim a_i = m$$

$$n = \text{rango de la retícula}$$

Para esquemas PQC se usa que las coordenadas de los vectores de la base estén en el anillo de enteros \mathbb{Z}_q .

Por lo general q se escoge como primo o potencia de dos.

Las operaciones se hacen "mod q "

Notación: $a_i \in \mathbb{Z}_q^m$

$$A = \{a_1, a_2, \dots, a_n\} \in \mathbb{Z}_q^{m \times n}$$

12.2.1 El problema "Learning With Errors" (LWE)

Recordando: Para resolver el DLP (discrete logarithm problem), un atacante tenía que determinar un entero secreto x tg $d^x \equiv \beta \pmod p$ dados \mathbb{Z}_p^* , generador d y el target β en el campo.

Podíamos pensar a d como la base.

Para el LWE se usa un problema similar, pero en una retícula.

La idea base es tomar cierta cant. da de elementos de la base $a_1, a_2, \dots, a_n \in \mathbb{Z}_q^m$ y multiplicarlos con coeficientes enteros secretos s_1, \dots, s_p para llegar al punto objetivo t en la retícula:

$$s_1 a_1 + s_2 a_2 + \dots + s_n a_n = t$$

El problema es similar al de DLP:

- dado un punto de comienzo \leftarrow base
- \nearrow el punto final \leftarrow target t
- el adversario quiere encontrar (s_1, \dots, s_n)

En forma matricial $A \cdot s = t$, donde
 $s = (s_1, s_2, \dots, s_n)$

Ej: Retícula sobre \mathbb{Z}_{17} de rango $n=4$. Vectores de dim. $m=7$, $A \in \mathbb{Z}_{17}^{7 \times 4}$, t vector dim 7
 El problema es:

$$A \cdot s = \begin{pmatrix} 10 & 13 & 16 & 10 \\ 12 & 5 & 14 & 12 \\ 10 & 11 & 8 & 11 \\ 7 & 7 & 5 & 3 \\ 8 & 13 & 8 & 9 \\ 14 & 13 & 13 & 16 \\ 7 & 6 & 10 & 4 \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} \equiv \begin{pmatrix} 2 \\ 14 \\ 7 \\ 6 \\ 10 \\ 8 \\ 16 \end{pmatrix} = t \bmod 17$$

Es "fácil" encontrar s , es un sistema de ecuaciones lineales.

la solución es

$$\mathbf{s} = \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} 6 \\ 2 \\ 12 \\ 3 \end{pmatrix}$$

Pero afortunadamente, podemos convertir este problema en uno difícil:

Consideremos t un punto **no** de la retícula pero **Cerrano** a un punto en la retícula

Matemáticamente se traduce en introducir un error:

$$A \cdot s + e = t'$$

En el ejemplo se vería así:

$$\begin{pmatrix} 10 & 13 & 16 & 10 \\ 12 & 5 & 14 & 12 \\ 10 & 11 & 8 & 11 \\ 7 & 7 & 5 & 3 \\ 8 & 13 & 8 & 9 \\ 14 & 13 & 13 & 16 \\ 7 & 6 & 10 & 4 \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{pmatrix} \equiv \begin{pmatrix} 1 \\ 13 \\ 8 \\ 6 \\ 10 \\ 9 \\ 15 \end{pmatrix} \pmod{17}$$

Un ataque usando eliminación Gaussiana o cualquier otra técnica para resolver ec. lineal no va a ser capaz de encontrar la solución correcta, ya que t' no es un punto de la retícula.

los vectores t y t' son muy cercanos y podemos ver que pequeñas variaciones de los coeficientes de la retícula, por ej. con errores en $e_i \in \{-1, 0, 1\}$, son suficiente para crear un problema no trivial. En el ejemplo:

$$\mathbf{e} = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{pmatrix} = \begin{pmatrix} -1 \\ -1 \\ 1 \\ 0 \\ 0 \\ 1 \\ -1 \end{pmatrix} \equiv \begin{pmatrix} 16 \\ 16 \\ 1 \\ 0 \\ 0 \\ 1 \\ 16 \end{pmatrix} \pmod{17}$$

En conclusión, introducir errores a nuestro sistema nos permite alcanzar la meta:

"Técnicas estándar para resolver sistemas de ec. lineales no pueden ser usadas para recuperar los valores desconocidos de s y e eficientemente".

Para dist. en la vida real, al escoger n y m suficientemente grandes, nos aseguramos de que la fuerza bruta tampoco pueda resolver el problema.

En realidad se vuelve un problema NP-completo los cuales son incluso difíciles en computadoras cuánticas de larga escala.

Esta def. nos permite crear sistemas que son "quantum computer-resistant":

Definition 12.2.2 Learning With Errors Problem (LWE)

Given a set of n basis vectors $\mathbf{a}_i \in \mathbb{Z}_q^m$ represented by matrix \mathbf{A} and a point $\mathbf{t} \in \mathbb{Z}_q^m$.

The LWE is the problem of determining a set of secret coefficients $\mathbf{s} = (s_1, s_2, \dots, s_n)$, with $s_i \in \mathbb{Z}_q$, such that:

$$\mathbf{A} \cdot \mathbf{s} + \mathbf{e} \equiv \mathbf{t} \pmod{q}$$

where \mathbf{e} is an unknown error vector consisting of small integers modulo q .

12.2.2 A simple LWE-Based Encryption System

El problema LWE permite construir un esquema de cifrado de llave pública donde Alice cifra un mensaje y Bob lo descifra. Durante el cifrado se añade un error aleatorio pequeño para ocultar información y este debe ser manejado cuidadosamente para que Bob pueda recuperar los bits originales aún con perturbaciones.

Codificación de bits

El mensaje consiste en bits $m: 630, 17$. Para usar LWE, cada bit debe mapearse a un valor del grupo cíclico \mathbb{Z}_q .

Solución óptima: usar valores alejados en el circuito modular:

Para maximizar la distancia entre los valores asignados, se propone

$$\bar{m}_i = \text{enc}_G(m_i) = \left\lfloor \frac{q}{2} \right\rfloor \cdot m_i$$

o decir

- Si $m_i = 0 \rightarrow \bar{m}_i = 0$
- Si $m_i = 1 \rightarrow \bar{m}_i = \left\lfloor \frac{q}{2} \right\rfloor$

Con $q=61$:

- $m_i = 0 \rightarrow \bar{m}_i = 0$
- $m_i = 1 \rightarrow \bar{m}_i = 30$

Esto coloca a los codigos en extremos opuestos del círculo, reduciendo el riesgo de confusión

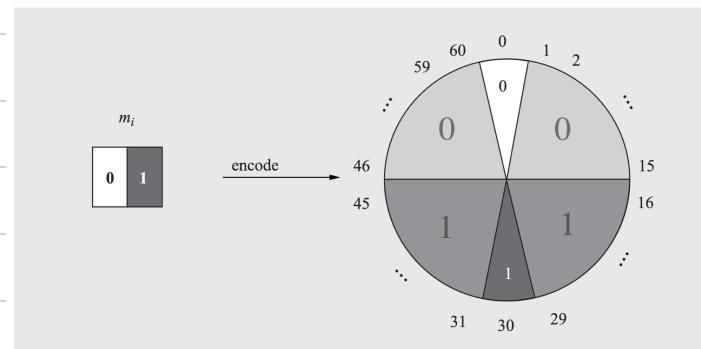


Fig. 12.4 Safe mapping of bits m_i to coefficients $\bar{m}_i \in \mathbb{Z}_{61}$

Describir

Para decidir si el valor recibido corresponde a 0 o 1 se usan regiones como rebanadas de pizza alrededor de $0 \times q/2$.

$$m_i = \text{dec}(\bar{m}_i) = \begin{cases} 0 & \text{si } -\left\lfloor \frac{q}{4} \right\rfloor \leq \bar{m}_i \leq \left\lfloor \frac{q}{4} \right\rfloor \\ 1 & \text{en otro caso} \end{cases}$$

Con $q=61$:

- $\lfloor \frac{q}{4} \rfloor = 15$
- todo valor entre 45 y 60 y entre 0 y 15 se codifican a 0
- todo valor entre 16 y 45 $\rightarrow 1$

Esto hace la codificación tolerante al error generado en el cifrado LWE.

Resumen del esquema de cifrado y descifrado Simple-LWE

El esquema Simple-LWE es una versión educativa y simplificada de un esquema basado en LWE. Solo permite cifrar un bit por vez pero nos sirve para entender como funcionan los cifrados basados en retículas.

Consta de 3 fases:

Generación de llaves

↓
Cifrado

↓
Descifrado

① Generación de llaves

Simple-LWE Key Generation

Output: public key: $k_{pub} = (\mathbf{t}, \mathbf{A})$ with $\mathbf{t} \in \mathbb{Z}_q^k$ and $\mathbf{A} \in \mathbb{Z}_q^{k \times n}$

private key: $k_{pr} = \mathbf{s} \in \mathbb{Z}_q^n$

Pasos:

1. Choose n random vectors $\mathbf{a}_i \in \mathbb{Z}_q^k$ and combine them in a matrix $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n) \in \mathbb{Z}_q^{k \times n}$.
2. Generate a random secret key \mathbf{s} from "small" integers.
3. Build a random error vector \mathbf{e} from "small" integers.
4. Compute $\mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$.
5. Return the public key $k_{pub} = (\mathbf{t}, \mathbf{A})$ and the private key $k_{pr} = \mathbf{s}$.

② Cifrado: El remitente usa la llave pública para cifrar un solo bit en $\{0, 1\}$

Simple-LWE Encryption

Input: public key $k_{pub} = (\mathbf{t}, \mathbf{A})$, message $m \in \{0, 1\}$

Output: ciphertext $\mathbf{c} = (\mathbf{c}_{aux}, c_{msg})$ with $\mathbf{c}_{aux} \in \mathbb{Z}_q^n$ and $c_{msg} \in \mathbb{Z}_q$

Pasos

1. Sample small random integers into vectors \mathbf{r} , \mathbf{e}_{aux} and a value e_{msg} .
2. Encode the message m : $\bar{m} = \text{enc}(m) \in \mathbb{Z}_q$.
3. Compute $\mathbf{c}_{aux} = \mathbf{A}^T \cdot \mathbf{r} + \mathbf{e}_{aux}$.
4. Compute $c_{msg} = \mathbf{t}^T \cdot \mathbf{r} + e_{msg} + \bar{m}$.
5. Return the ciphertext $\mathbf{c} = (\mathbf{c}_{aux}, c_{msg})$.

③ Descifrado:

El receptor usa la llave privada s para recuperar el bit.

$$\mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$$

Simple-LWE Decryption

Input: private key $k_{pr} = \mathbf{s} \in \mathbb{Z}_q^n$, ciphertext $\mathbf{c} = (\mathbf{c}_{aux}, c_{msg})$

Output: message $m \in \{0, 1\}$

Pasos:

1. Return message $m = \text{dec}(c_{msg} - \mathbf{s}^T \cdot \mathbf{c}_{aux})$.

usa la función que

es tolerante a errores para recuperar el bit

$\mathbf{s}^T \cdot \mathbf{A}^T \cdot \mathbf{r} + \mathbf{s}^T \cdot \mathbf{e}_{aux}$ se elimina porque conoce s y deja el mensaje más un ruido manejable

Prueba de corrección y seguridad

Partan de la ecuación de generación de llave
 $t = A \cdot s + e$

Seguridad

Se consideran dos tipos de ataque

① El atacante quiere calcular la llave privada a partir de la pública:

El atacante quiere obtener s , lo cual sería resolver el problema LWE que se supone es difícil si los parámetros los escogimos bien.

② Recuperar el mensaje a partir del cifrado

El cifrado es el par

$$c_{aux} = A^T r + e_{aux}, \quad c_{msg} = t^T r + e_{msg} + m$$

- entender c_{aux} : el atacante no conoce r ni $e_{aux} \Rightarrow$ problema LWE
- entender $c_{msg} \Rightarrow$ problema LWE

Por lo tanto, tanto romper la llave como descifrar el mensaje son problemas LWE.

Corrección:

Se puede demostrar que con alta probabilidad y errores suficientemente pequeños, al calcular $c_{msg} - s^T c_{aux}$ se obtiene un valor cercano a m y la función de descodificación tamaña al bit correcto.

LIMITACIONES DEL ESQUEMA SIMPLE-LWE

Solo es pedagógico no es práctico.

① Cifra solo 1 bit:

Para aplicaciones reales, por ej. cifrar una llave simétrica de 256 bits, hay que extender el esquema a múltiples bits de manera eficiente.

② Expansión del cifrado

Un solo bit se convierte en un cifrado con $n+1=4$ elementos de \mathbb{Z}_6 . (el ejemplo)

Cada valor necesita 6 bits para representar todos los posibles valores $c_i \in \{0, \dots, 60\}$. Entonces necesitamos un total 24 bits. Considerando que 1 bit \rightarrow 24 bits en aplicaciones reales, esto es demasiado.

Mejora: reducir esta expansión.

③ Llaves muy grandes

El usa de matrices para llaves públicas involucra complejidad cuadrática en términos de almacenamiento y ancho de banda.

El esquema FROB KEM basado en LWE usa $A \in \mathbb{Z}_q^{640 \times 640}$ por muy pequeño y $\mathbb{Z}_q^{1344 \times 1344}$ para su conj. de parámetros más grande.

④ Distribución de los errores.

Nunca se especifica lo de "errores" pequeños.

Este es un problema no trivial y bastante estudiado.
Se usa por ej. las distribuciones Gaussiana discreta
o binomial.

12.2.3 The Ring Learning Problem with Errors

los esquemas modernos sustituyen las matrices por polinomios, lo cual reduce drásticamente el almacenamiento y la complejidad computacional.

Definition 12.2.3 The ring $R_q = \mathbb{Z}_q[x]/(x^n + 1)$

The polynomial ring $\mathbb{Z}_q[x]/(x^n + 1)$ consists of all polynomials with a maximum degree of $n - 1$ with coefficients from \mathbb{Z}_q and n being a power of two, i.e., $n = 2^i$.

The ring operations addition, subtraction and multiplication are performed as regular polynomial arithmetic, with the results being reduced modulo the cyclotomic polynomial $x^n + 1$. All integer coefficients are reduced modulo q .

Definición del problema Ring-LWE

Reemplazamos la matriz grande de LWE
por un polinomio único $a(x)$.

Definition 12.2.4 Ring-LWE Problem

Let R_q denote the ring $\mathbb{Z}[x]_q/(x^n + 1)$, where q is a prime and the positive integer n is a power of two. Given are polynomials \mathbf{a} and $\mathbf{t} \in R_q$.

Ring-LWE is the problem of determining a secret polynomial $s \in R_q$ such that:

$$\mathbf{a}(x) \cdot s(x) + e(x) = \mathbf{t}(x)$$

where the error vector e is a polynomial in the ring R_q with small integer coefficients obtained from a discrete distribution D .

a, t
tienen coef.
grandes)

s y e valores
pequeños

Relación entre LWE estándar y Ring-LWE

Similitudes

- * ambos generan retículos con buenas propiedades
- * Resolver Ring-LWE se hace tan difícil como LWE

Diferencias clave

LWE:

- usa matriz A grande $K \times n$
- muy costoso computacionalmente

Ring-LWE

- Solo usa un polinomio $a(x)$ de grado $n-1$
- Reduce masivamente el costo computacional
- Ocupa menos memoria y ancho de banda

Desventaja del Ring-LWE

Introduce más estructura algebraica, lo que abre la posibilidad teórica de ataques que la explotan.

No se conocen ataques prácticos que lo rompan con parámetros seguros.

→ Modulo-LWE: equilibrio, eficiencia y seguridad. Es la base de muchos esq post cuánticos.