HOMOMORPHIC ENCRYPTION AND LATTICE BASED CRYPTOGRAPHY

Abderrahmane Nitaj

Laboratoire de Mathématiques Nicolas Oresme

Université de Caen Normandie, France

Nouakchott, February 15-26, 2016

Contents

Contents

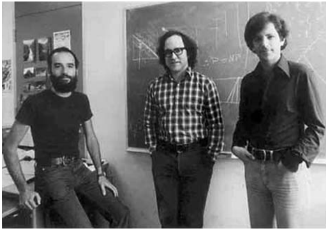
Cryptography in daily life

1. Cell phone conversations
2. Emails
3. Shopping online
4. Online banking
5. Aircraft Communications
6. Satellite communications
7. Government communications
8. Medical records
9. Cloud storage (Dropbox, Microsoft One Drive, Google Drive,...)

RSA

RSA

- Invented in 1978 by Rivest, Shamir and Adleman.



- Hard Problem : IFP

Integer Factorization Problem:

Let $N = pq$ be the product of two large prime numbers p and q. The integer factorization problem is to find p and q.

RSA

RSA

- Invented in 1978 by Rivest, Shamir and Adleman.

- Hard Problem : IFP

Integer Factorization Problem:

Let $N = pq$ be the product of two large prime numbers p and q. The integer factorization problem is to find p and q.

RSA

RSA

- Invented in 1978 by Rivest, Shamir and Adleman.
- Hard Problem : IFP

Integer Factorization Problem:

Let $N = pq$ be the product of two large prime numbers p and q. The integer factorization problem is to find p and q.

Diffie-Hellman and El Gamal

- Diffie-Hellman: invented in 1976 by W. Diffie and M. Hellman.
- El Gamal: invented in 1985 by T. El Gamal.

- Hard Problem: DLP

Discrete Logarithm Problem:

Let g and b be two positive integers and p be prime number. Find x such that $g^x \equiv b \pmod{p}$.

Diffie-Hellman and El Gamal

- Diffie-Hellman: invented in 1976 by W. Diffie and M. Hellman.
- El Gamal: invented in 1985 by T. El Gamal.

- Hard Problem: DLP

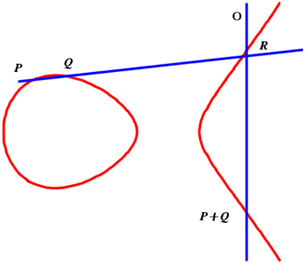
Discrete Logarithm Problem:

Let g and b be two positive integers and p be prime number. Find x such that $g^x \equiv b \pmod{p}$.

ECC

ECC

- ECC (Elliptic Curve Cryptography), invented in 1985 (independently) by Koblitz and Miller.



- Hard Problem: ECLDP

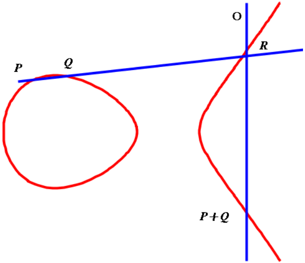
Elliptic Curve Discrete Logarithm Problem :

Let P and Q be tow points on an elliptic curve E. Find n such that $nP = Q$.

ECC

ECC

- ECC (Elliptic Curve Cryptography), invented in 1985 (independently) by Koblitz and Miller.



- Hard Problem: ECLDP

Elliptic Curve Discrete Logarithm Problem :

Let P and Q be tow points on an elliptic curve E. Find n such that $nP = Q$.

Contents

Desirable cryptographic properties

For the cloud

- Store encrypted data on the cloud.
- Allow the cloud to process on the encrypted data
- Perform computations or search on data without decrypting it.
- Decrypt the result to get the same answer as performing an analogous operation on the original data.

Desirable cryptographic properties

Example

- Store the emails on the cloud.
- Encrypt an inquiry and perform it on the cloud without decrypting it.
- Decrypt the result to get the same answer as performing an analogous operation on the original data.

The simplified scheme

- Encrypt a data x as Enc(x) and store it on the cloud.
- Encrypt a function f as Enc(f) and send it to the cloud.
- Ask the cloud to perform Enc(f)[Enc(x)]=Enc($f(x)$).
- Download Enc($f(x)$) and decrypt it to get $f(x)$.

Desirable cryptographic properties

Example

- Store the emails on the cloud.
- Encrypt an inquiry and perform it on the cloud without decrypting it.
- Decrypt the result to get the same answer as performing an analogous operation on the original data.

The simplified scheme

- Encrypt a data x as $\mathsf{Enc}(x)$ and store it on the cloud.
- Encrypt a function f as $\mathsf{Enc}(f)$ and send it to the cloud.
- Ask the cloud to perform $\mathsf{Enc}(f)[\mathsf{Enc}(x)]=\mathsf{Enc}(f(x))$.
- Download $\mathsf{Enc}(f(x))$ and decrypt it to get $f(x)$.

Homomorphic systems

The concept of homomorphic encryption

- It allows certain types of operations to be carried out on the encrypted data without the need to decrypt them.
- Proposed by Rivest, Adleman, and Dertouzos in 1978.
- Many schemes are partially homomorphic.
- In 2009, Gentry presented the first fully homomorphic encryption scheme: totally impracticable.

Homomorphic systems

Homomorphism

If $(G_1, *)$ and $(G_2, \otimes)$ are two groups, then a function $f : G_1 \longrightarrow G_2$ is a group homomorphism if

$$f(x * y) = f(x) \otimes f(y)$$

for all $x, y \in G_1$
Examples: $f(x) = e^x$, $f(x) = \log(x)$,....

Partially homomorphic encryption

- Additively homomorphic: Enc$(x)+$Enc$(y)=$ Enc$(x + y)$.
- Multiplicatively: Enc$(x)\times$ Enc$(y)=$ Enc$(x \times y)$.

Fully homomorphic encryption (FHE)

Fully homomorphic encryption allows to do arbitrary computations on encrypted data without decrypting it.

The RSA example

RSA addition: not homomorphic

Private		The cloud
m_1	$\xrightarrow{RSA(N,e)}$	$c_1 \equiv m_1^e \pmod{N}$
m_2	$\xrightarrow{RSA(N,e)}$	$c_2 \equiv m_2^e \pmod{N}$
		$\downarrow \oplus$
$(m_1^e + m_2^e)^d \not\equiv m_1 + m_2$	$\xleftarrow{RSA(N,d)}$	$c_1 + c_2 \equiv m_1^e + m_2^e \pmod{N}$

RSA multiplication: homomorphic

Private		The cloud
m_1	$\xrightarrow{RSA(N,e)}$	$c_1 \equiv m_1^e \pmod{N}$
m_2	$\xrightarrow{RSA(N,e)}$	$c_2 \equiv m_2^e \pmod{N}$
		$\downarrow \otimes$
$((m_1 m_2)^e)^d \equiv m_1 m_2$	$\xleftarrow{RSA(N,d)}$	$c_1 c_2 \equiv (m_1 m_2)^e \pmod{N}$

The RSA example

RSA addition: not homomorphic

Private		The cloud
m_1	$\xrightarrow{RSA(N,e)}$	$c_1 \equiv m_1^e \pmod N$
m_2	$\xrightarrow{RSA(N,e)}$	$c_2 \equiv m_2^e \pmod N$
		$\downarrow \oplus$
$(m_1^e + m_2^e)^d \not\equiv m_1 + m_2$	$\xleftarrow{RSA(N,d)}$	$c_1 + c_2 \equiv m_1^e + m_2^e \pmod N$

RSA multiplication: homomorphic

Private		The cloud
m_1	$\xrightarrow{RSA(N,e)}$	$c_1 \equiv m_1^e \pmod N$
m_2	$\xrightarrow{RSA(N,e)}$	$c_2 \equiv m_2^e \pmod N$
		$\downarrow \otimes$
$((m_1 m_2)^e)^d \equiv m_1 m_2$	$\xleftarrow{RSA(N,d)}$	$c_1 c_2 \equiv (m_1 m_2)^e \pmod N$

Partial homomorphic systems

RSA: multiplicatively homomorphic

Given $c_1 \equiv m_1^e \pmod{N}$, $c_2 \equiv m_2^e \pmod{N}$. Then

$$c_1 \times c_2 \equiv m_1^e \times m_2^e \equiv (m_1 \times m_2)^e \pmod{N}.$$

ElGamal: multiplicatively homomorphic

Given $c_1 = \left(g^{a_1}, g^{a_1 b_1} m_1\right) \pmod{p}$, $c_2 = \left(g^{a_2}, g^{a_2 b_2} m_2\right) \pmod{p}$. Then

$$c_1 \times c_2 = \left(g^{a_1+a_2}, g^{a_1 b_1 + a_2 b_2} m_1 m_2\right) \pmod{p}.$$

Paillier: additively homomorphic

Given $c_1 = g^{m_1} r_1^N \pmod{N^2}$, $c_2 = g^{m_2} r_2^N \pmod{N^2}$. Then

$$c_1 \times c_2 = g^{m_1+m_2} (rs)^N \pmod{N^2}.$$

DGHV: Somewhat Homomorphic Encryption

DGHV: 2010

- Invented by van Dijk, Gentry, Halevi, and Vaikuntanathan.
- The first fully homomorphic encryption over the integers.
- Choose a secret large prime key p.
- Choose a large integer q.
- Choose a small integer $r < \frac{p}{2}$.
- Encrypt $m \in \{0,1\}$ as $c = qp + 2r + m$.
- Decrypt c using $(c \mod p) \mod 2 = m$.

Homomorphic properties of DGHV

$$c_1 = q_1p + 2r_1 + m_1, \qquad c_2 = q_2p + 2r_2 + m_2$$

Addition

$$c_1 + c_2 = (q_1 + q_2)p + 2(r_1 + r_2) + m_1 + m_2.$$

Hence Enc($m_1 + m_2$)=Enc(m_1)+Enc(m_2).

Multiplication

$$c_1 \times c_2 = (c_2q_1 + c_1q_2 - q_1q_2p)p + 2(2r_1r_2 + r_1m_2 + r_2m_1) + m_1 \times m_2.$$

Hence Enc($m_1 \times m_2$)=Enc(m_1)$\times$ Enc(m_2).

Contents

Learning With Errors

LWE

- Invented by O. Regev in 2005.
- Security based on the GapSVP problem.
- Provable Security.

Definition

The GapSVP problem: Let $\mathcal{L}$ be a lattice with a basis B. Let $\lambda_1(\mathcal{L})$ be the length of the shortest nonzero vector of $\mathcal{L}$. Let $\gamma > 0$ and $r > 0$. Decide whether $\lambda_1(\mathcal{L}) < r$ or $\lambda_1(\mathcal{L}) > \gamma r$.

Learning With Errors

Example

- Easy: solve the system

$$\begin{bmatrix} 17 & 42 & -127 \\ 24 & 3 & 71 \\ -7 & -23 & 45 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -3265 \\ 246 \\ 1202 \end{bmatrix}$$

- Harder: solve the system

$$\underbrace{\begin{bmatrix} 117 & 422 & -127 \\ 214 & 23 & 71 \\ -17 & -223 & 45 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}_{S} \underbrace{\begin{matrix} + \\ + \\ + \end{matrix}}_{} \underbrace{\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix}}_{E} \underbrace{\begin{matrix} = \\ = \\ = \end{matrix}}_{} \underbrace{\begin{bmatrix} -4718 \\ 4177 \\ 2485 \end{bmatrix}}_{P}$$

- $AS + E = P$: LWE equation over $\mathbb{Z}$.

Learning With Errors

Example

- Hard: solve the system

$$\begin{bmatrix} 17 & 42 & 127 \\ 24 & 3 & 71 \\ 7 & 23 & 45 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 116 & \pmod{503} \\ 158 & \pmod{503} \\ 271 & \pmod{503} \end{bmatrix}$$

- Much harder: solve the system

$$\underbrace{\begin{bmatrix} 117 & 422 & 127 \\ 214 & 23 & 71 \\ 17 & 223 & 45 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}_{S} \underbrace{+}_{+} \underbrace{\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix}}_{E} \underbrace{=}_{=} \underbrace{\begin{bmatrix} 144 & \pmod{503} \\ 229 & \pmod{503} \\ 503 & \pmod{503} \end{bmatrix}}_{P}$$

- $AS + E = P$: LWE equation over $\mathbb{Z}_{503}$.

Learning With Errors

LWE Key Generation

Algorithm 1 : LWE Key Generation

Require: Integers n, m, l, q.
Ensure: A private key S and a public key (A, P).

1: Choose $S \in \mathbb{Z}_q^{n\times l}$ at random.
2: Choose $A \in \mathbb{Z}_q^{m\times n}$ at random.
3: Choose $E \in \mathbb{Z}_q^{m\times l}$ according to $\chi(E) = e^{-\pi\|E\|^2/r^2}$ for some $r > 0$.
4: Compute $P = AS + E \pmod{q}$. Hence $P \in \mathbb{Z}_q^{m\times l}$.
5: The private key is S.
6: The public key is (A, P).

Learning With Errors

LWE: Encryption

Algorithm 2 : LWE Encryption

Require: Integers n, m, l, t, r, q, a public key (A, P) and a plaintext $M \in \mathbb{Z}_t^{l\times 1}$.

Ensure: A ciphertext (u, c).

1: Choose $a \in [-r, r]^{m\times 1}$ at random.

2: Compute $u = A^T a \pmod{q} \in \mathbb{Z}_q^{n\times 1}$.

3: Compute $c = P^T a + \left[\frac{Mq}{t}\right] \pmod{q} \in \mathbb{Z}_q^{l\times 1}$.

4: The ciphertext is (u, c).

Learning With Errors

LWE: Decryption

Algorithm 3 : LWE Decryption

Require: Integers n, m, l, t, r, q, a private key S and a ciphertext (u, c).

Ensure: A plaintext M.

1: Compute $v = c - S^T u$ and $M = \left[\frac{tv}{q}\right]$.

Learning With Errors

Correctness of decryption

We have

$$\begin{aligned} v &= c - S^T u \\ &= (AS+E)^T a - S^T A^T a + \left[\frac{Mq}{t}\right] \\ &= E^T a + \left[\frac{Mq}{t}\right]. \end{aligned}$$

Hence

$$\left[\frac{tv}{q}\right] = \left[\frac{tE^T a}{q} + \frac{t}{q}\left[\frac{Mq}{t}\right]\right].$$

With suitable parameters, the term $\frac{tE^T a}{q}$ is negligible and $\frac{t}{q}\left[\frac{Mq}{t}\right] = M$.

Consequently $\left[\frac{tv}{q}\right] = M$.

LWE

Hard Problem

Equations

- The public equation $P = AS + E \pmod q$.
- The public ciphertext $c = P^T a + \left[\frac{Mq}{t}\right] \pmod q$.
- Can be reduced to the approximate-SVP and GapSVP.

q-ary lattices

Let $A \in \mathbb{Z}_q^{n\times l}$ for some integers q, n, l.

- The q-ary lattice:

$$\Lambda_q(A) = \left\{ y \in \mathbb{Z}^l : \quad y \equiv A^T s \pmod q \quad \text{for some} \quad s \in \mathbb{Z}^n \right\}.$$

- The orthogonal q-ary lattice:

$$\Lambda_q^{\perp}(A) = \left\{ y \in \mathbb{Z}^l : \quad Ay \equiv 0 \pmod q \right\}.$$

Contents

NTRU

NTRU

- Invented by Hoffstein, Pipher et Silverman in 1996.
- Security based on the Shortest Vector Problem (SVP).
- Various versions between 1996 and 2001.

Definition

The Shortest Vector Problem (SVP): Given a basis matrix B for $\mathcal{L}$, compute a non-zero vector $v \in \mathcal{L}$ such that $\|v\|$ is minimal, that is $\|v\| = \lambda_1(\mathcal{L})$.

NTRU: Ring of Convolution $\Pi = \mathbb{Z}[X]/(X^N - 1)$

Polynomials

$f = \sum_{i=0}^{N-1} f_i X^i, \qquad g = \sum_{i=0}^{N-1} g_i X^i,$

Sum

$f + g = (f_0 + g_0, f_1 + g_1, \cdots, f_{N-1} + g_{N-1}).$

Product

$f * g = h = (h_0, h_1, \cdots, h_{N-1})$ with

$$\boxed{h_k = \sum_{i+j \equiv k \pmod{N}} f_i g_j.}$$

NTRU: Ring of Convolution $\Pi = \mathbb{Z}[X]/(X^N - 1)$

Polynomials

$f = \sum_{i=0}^{N-1} f_i X^i, \qquad g = \sum_{i=0}^{N-1} g_i X^i,$

Sum

$f + g = (f_0 + g_0, f_1 + g_1, \cdots, f_{N-1} + g_{N-1}).$

Product

$f * g = h = (h_0, h_1, \cdots, h_{N-1})$ with

$$h_k = \sum_{i+j \equiv k \pmod N} f_i g_j.$$

NTRU: Ring of Convolution $\Pi = \mathbb{Z}[X]/(X^N - 1)$

Polynomials

$f = \sum_{i=0}^{N-1} f_i X^i, \qquad g = \sum_{i=0}^{N-1} g_i X^i,$

Sum

$f + g = (f_0 + g_0, f_1 + g_1, \cdots, f_{N-1} + g_{N-1}).$

Product

$f * g = h = (h_0, h_1, \cdots, h_{N-1})$ with

$$\boxed{h_k = \sum_{i+j \equiv k \pmod{N}} f_i g_j.}$$

NTRU: Ring of Convolution $\Pi = \mathbb{Z}[X]/(X^N - 1)$

Convolution

$$\underbrace{f = (f_0, f_1, \cdots, f_{N-1}), \qquad g = (g_0, g_1, \cdots, g_{N-1})}_{f * g = h = (h_0, h_1, \cdots, h_{N-1})}.$$

	1	X	$\cdots$	X^k	$\cdots$	X^{N-1}
	f_0g_0	f_0g_1	$\cdots$	f_0g_k	$\cdots$	f_0g_{N-1}
$+$	f_1g_{N-1}	f_1g_0	$\cdots$	f_1g_{k-1}	$\cdots$	f_1g_{N-2}
$+$	f_2g_{N-2}	f_2g_{N-1}	$\cdots$	f_2g_{k-2}	$\cdots$	f_2g_{N-3}
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\cdots$	$\vdots$	$\vdots$
$+$	$f_{N-2}g_2$	$f_{N-2}g_3$	$\cdots$	$f_{N-2}g_{k+2}$	$\cdots$	$f_{N-2}g_1$
$+$	$f_{N-1}g_1$	$f_{N-1}g_2$	$\cdots$	$f_{N-1}g_{k+1}$	$\cdots$	$f_{N-1}g_0$
$h =$	h_0	h_1	$\cdots$	h_k	$\cdots$	h_{N-1}

NTRU Parameters

- $N =$ a prime number (e.g. $N = 167,\ 251,\ 347,\ 503$).
- $q =$ a large modulus (e.g. $q = 128,\ 256$).
- $p =$ a small modulus (e.g. $p = 3$).

NTRU Algorithms

Key Generation:

- Randomly choose two private polynomials f and g.
- Compute the inverse of f modulo q: $f * f_q = 1 \pmod{q}$.
- Compute the inverse of f modulo p: $f * f_p = 1 \pmod{p}$.
- Compute the public key $h = f_q * g \pmod{q}$.

NTRU Algorithms

Encryption:

- m is a plaintext in the form of a polynomial mod q.
- Randomly choose a private polynomial r.
- Compute the encrypted message $e = m + pr * h \pmod{q}$.

Decryption:

- Compute $a = f * e = f * (m + pr * h) = f * m + pr * g \pmod{q}$.
- Compute $a * f_p = (f * m + pr * g) * f_p = m \pmod{p}$.

NTRU Algorithms

Encryption:

- m is a plaintext in the form of a polynomial mod q.
- Randomly choose a private polynomial r.
- Compute the encrypted message $e = m + pr * h \pmod{q}$.

Decryption:

- Compute $a = f * e = f * (m + pr * h) = f * m + pr * g \pmod{q}$.
- Compute $a * f_p = (f * m + pr * g) * f_p = m \pmod{p}$.

NTRU Algorithms

Encryption:

- m is a plaintext in the form of a polynomial mod q.
- Randomly choose a private polynomial r.
- Compute the encrypted message $e = m + pr * h \pmod{q}$.

Decryption:

- Compute $a = f * e = f * (m + pr * h) = f * m + pr * g \pmod{q}$.
- Compute $a * f_p = (f * m + pr * g) * f_p = m \pmod{p}$.

NTRU

Correctness of decryption

We have

$$
\begin{aligned}
a &\equiv f * e \pmod q \\
a &\equiv f * (p * r * h + m) \pmod q \\
a &\equiv f * r * (p * g * f_q) + f * m \pmod q \\
a &\equiv p * r * g * f * f_q + f * m \pmod q \\
a &\equiv p * r * g + f * m \pmod q.
\end{aligned}
$$

If $p * r * g + f * m \in \left[-\frac{q}{2}, \frac{q}{2}\right]$, then

$$m \equiv a * f_p \mod p.$$

MAPLE p. 24

NTRU

Example

Key generation

- Public parameters $N = 13$, $p = 3$, $q = 8$.
- Private keys $f = X^{12} + X^{11} + X^{10} + X^9 + X^8 + X^7 + 1$, $g = X^{12} + X^5 - X^4 + X^3 - X^2 + X - 1$.
- $f * f_p \equiv 1 \pmod p$ with $f_p = 2X^{12}+2X^{11}+2X^{10}+2X^9+2X^8+2X^7+2X^5+2X^4+2X^3+2X^2+2X$.
- $f * f_q \equiv 1 \pmod q$ with $f_q = X^{12}+X^{11}+X^{10}+X^9+X^8+X^7+2X^6+X^5+X^4+X^3+X^2+X+2$.
- The public key is $h \equiv g * f_q \pmod q = 2X^{12} + 2X^{11} + 2X^9 + 2X^7 + 3X^5 + 2X^3 + 2X$.

NTRU

Example

Encryption

- Message $m = X^{10} + X^8 + X^7 + X^4 + X^3 + 1.$
- Random error $r = X^{12} + X^{11} + X^8 + X^7 + 1.$
- The ciphertext $e = \equiv p * r * h + m \pmod{q} \equiv 5X^{12}+2X^{11}+3X^{10}+2X^9+5X^8+3X^7+2X^6+5X^5+6X^4+4X^3+2X.$

NTRU

Example

Decryption

- $$\begin{aligned} a &\equiv f * e \pmod q \\ &\equiv 6X^{12} + 3X^{11} + 6X^{10} + 2X^9 + 3X^8 + 4X^7 \\ &\quad +6X^6 + 6X^5 + 4X^4 + 7X^3 + X^2 + 6X + 3. \end{aligned}$$

- $$\begin{aligned} m &\equiv f_p * a \pmod p \\ &\equiv X^{10} + X^8 + X^7 + X^4 + X^3 + 1, \end{aligned}$$

Contents

Introduction to lattices

Definition

Let n and d be two positive integers. Let $b_1 \cdots, b_d \in \mathbb{R}^n$ be d linearly independent vectors. The lattice $\mathcal{L}$ generated by $(b_1 \cdots, b_d)$ is the set

$$\mathcal{L} = \sum_{i=1}^{d} \mathbb{Z} b_i = \left\{ \sum_{i=1}^{d} x_i b_i \mid x_i \in \mathbb{Z} \right\}.$$

The vectors $b_1 \cdots, b_d$ are called a vector basis of $\mathcal{L}$. The lattice rank is n and the lattice dimension is d. If $n = d$ then $\mathcal{L}$ is called a full rank lattice.

Introduction to lattices

Example: Lattice with dimension 2

$$b_1 = \left[\begin{array}{c} 1 \\ 0 \end{array}\right],\ b_2 = \left[\begin{array}{c} 0.5 \\ 1 \end{array}\right],\ \mathcal{L} = \left\{v,\ v = x_1 b_1 + x_2 b_2,\ (x_1, x_2) \in \mathbb{Z}^2\right\}.$$

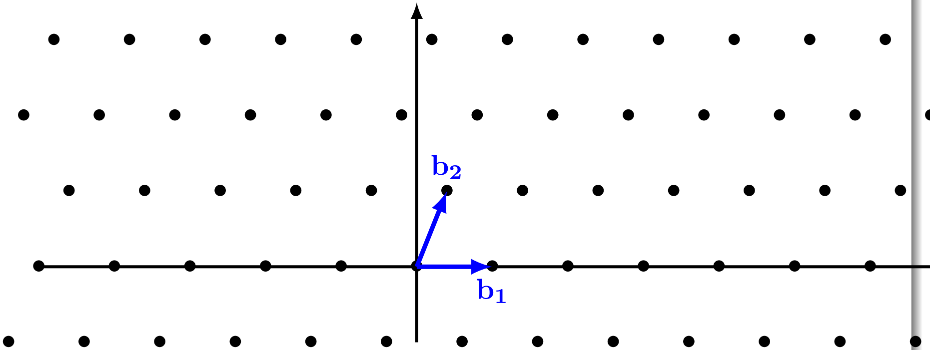


Figure: The lattice with the basis (b_1, b_2)

Introduction to lattices

Introduction to lattices

How to find v?

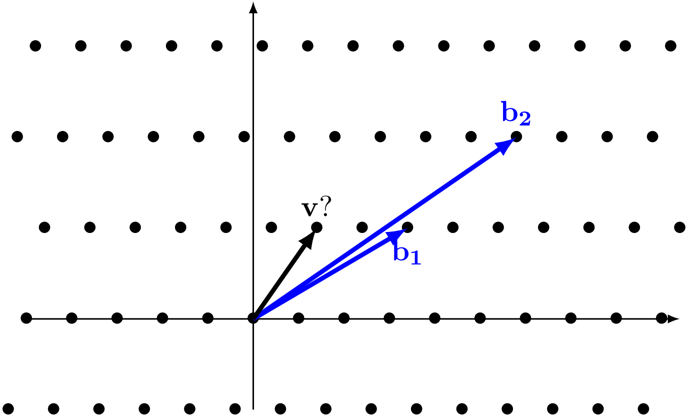


Figure: A lattice with *a bad* basis (b_1, b_2)

Introduction to lattices

How to find v?

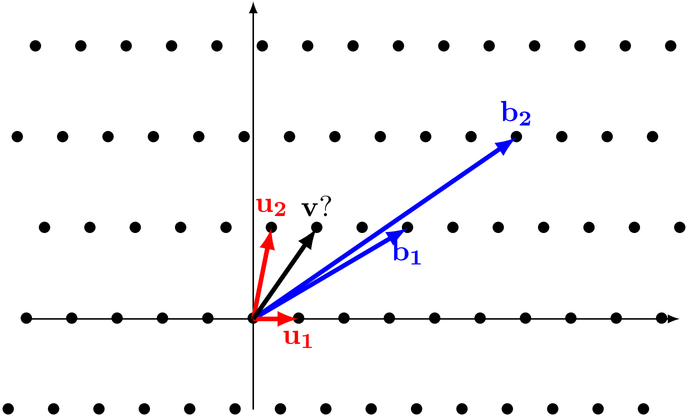


Figure: The same lattice with *a good* basis (u_1, u_2)

Short vectors

Definition (The Shortest Vector Problem (SVP))

Given a basis matrix B for $\mathcal{L}$, compute a non-zero vector $v \in \mathcal{L}$ such that $\|v\|$ is minimal, that is $\|v\| = \lambda_1(\mathcal{L})$.

Short vectors

The shortest vector

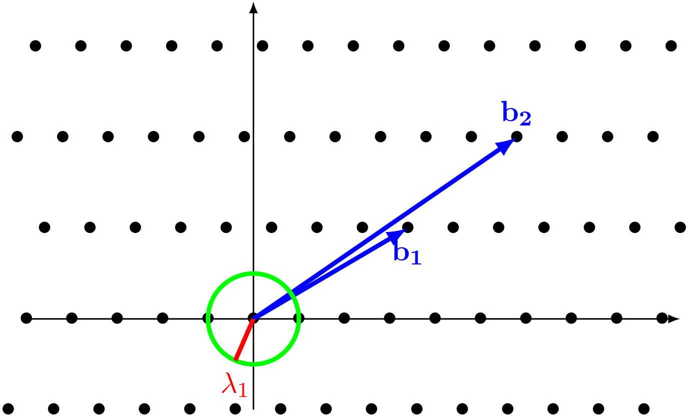


Figure: The shortest vectors

Closest Vectors

Definition (The Closest Vector Problem (CVP))

Given a basis matrix B for $\mathcal{L}$ and a vector $v \notin \mathcal{L}$, compute a vector $v_0 \in \mathcal{L}$ such that $\|v - v_0\|$ is minimal.

Closest Vectors

The closest vector

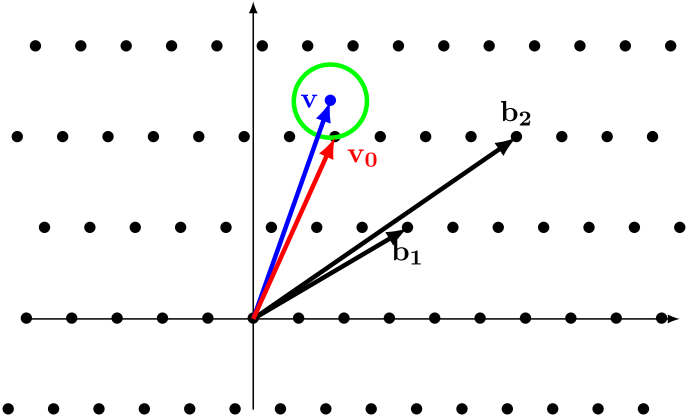


Figure: The closest vector to v is v_0

Contents

Bibliography

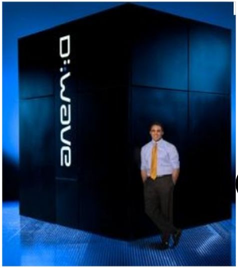
1. P.W. Shor: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM J. Computing 26, pp. 1484–1509 (1997).
2. O. Regev: On lattices, learning with errors, random linear codes, and cryptography, STOC 2005, ACM (2005) p. 84–93.
3. J. Hoffstein: J. Pipher, and J. H. Silverman, NTRU: A Ring Based Public Key Cryptosystem in Algorithmic Number Theory. Lecture Notes in Computer Science 1423, Springer-Verlag, pp. 267–288, 1998.
4. Pittet Shillong: 2013, November 18 - 29, Fourier analysis of groups in combinatorics, CIMPA-UNESCO-MESR-MINECO-INDIA research school: North Eastern Hill University, Shillong. https://hal.archives-ouvertes.fr/CIMPA/cel-00963668v1
5. A. Nitaj: Quantum and Post Quantum Cryptography. http://www.math.unicaen.fr/~nitaj/postquant.pdf

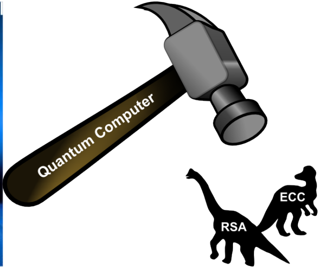
Contents

Conclusion

Lattice based cryptography

- Can be used to build cryptographic schemes (GGH, NTRU, LWE,...).
- Can be used to build fully homomorphic encryption, Digital signatures, identity based encryption IBE, hash functions.
- Many hard problems (SVP, CVP, ...).
- Fast implementation.
- Resistance to quantum computers and NSA.





Merci

Thank you

شكرًا