NTRU
public key cryptosystem

https://youtu.be/0h01vgILqGk

한성대학교 HANSUNG UNIVERSITY

CryptoCraft LAB
https://crypto.modoo.at

Contents

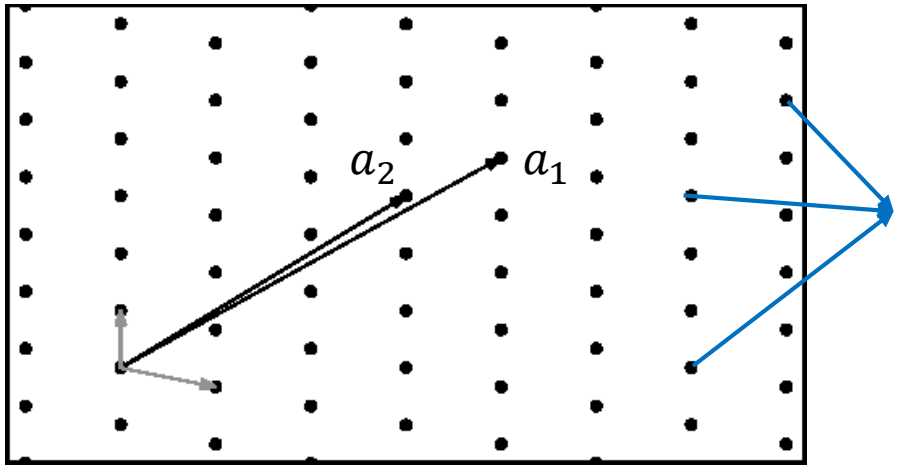
CryptoCraft LAB

Lattice based cryptography review

$$\textbf{Lattice } L = \left\{ \sum_{i=1}^{n} a_i s_i \;\middle|\; s_i \in Z \right\}$$: R^n에서 정수 계수(S_i)를 갖는 모든 기저들(a_i)의 선형 결합

$*\ R^n$의 $basis$: $\{a_1, a_2, \ldots, a_i\}$

→ a_i 로 vector space R^n 과 L 생성



b : lattice $= a_1 s_1 + a_2 s_2 + \ldots + a_n s_n$

$ex)\ bk = a_1 + 3a_2 \cdots$

기저들(a_i)의 선형결합으로 이루는 점들(b)의 집합

Lattice based cryptography review

- 격자 상의 계산의 어려움에 기반한 **NP-hard 문제**
  - **SVP(shortest vector problem)** : basis vector A가 주어질 때, 0이 아닌 벡터 중 가장 짧은 벡터는?
  - **CVP(closest vector problem)** : basis vector A가 주어질 때, target vector t와 가장 가까운 점 p는?

(t는 격자 위의 점일 필요 x)

- lattice based public key cryptosystem
  - **NTRU** : SVP의 어려움에 기반
- lattice reduction : 다른 기저를 갖는 격자 L이 주어질 때, 직교하는 벡터를 통해 축소 기저를 찾는 다항식 시간 알고리즘
  - **LLL algorithm** : 격자 기반 축소 알고리즘 → NTRU 공격
- lattice reduction은 짧은 벡터를 생성하도록 설계되어 있어 SVP 해결에 도움이 됨

→ **이를 막기 위한 parameter 설정 필요**

NIST round2 - NTRU

csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions

NTRU *(merger of NTRUEncrypt and NTRU-HRSS-KEM)*	Zip File (4MB) IP Statements Website	Cong Chen Oussama Danba Jeffrey Hoffstein Andreas Hulsing Joost Rijneveld John M. Schanck Peter Schwabe William Whyte Zhenfei Zhang	Submit Comment View Comments
NTRU Prime	Zip File (3MB) Website	Daniel J. Bernstein Chitchanok Chuengsatiansup Tanja Lange Christine van Vredendaal	Submit Comment View Comments

NTRU 변형

❖ NTRU-HRSS-KEM

- parameter 제한 : NTRUKEM743

❖ NTRU Prime

- N(소수), (Z / q) [x] / (x^N - x-1) 형식의 유한체 사용

❖ NTRUSign

- 전자서명

❖ 키 분배, 영지식 증명 등

NTRU 효율적 구현에 대한 연구 진행..

- Hoffstein과 Silverman은 보안성은 유지한 채 연산량을 줄이기 위해 특별한 형태의 다항식 사용 제안
- Bailey 등은 NTRU가 **리소스가 제한된 장치 상에서 효과적으로 구현될 수 있다**는 것을 증명
- Gaubatz, Kaps 및 Sunar는 **3,000개 이하의 게이트**만을 사용하는 NTRU의 하드웨어 구현을 제안
  → 센서 노드 상에서 공개 키 암호의 사용이 가능함을 증명
- NTRU를 변형하여 IoT 디바이스에 적용 / AVX2를 사용하여 최적화한 결과도 있음
- **연산량 감소의 필요성 여전히 존재**
  - 가장 큰 비중을 차지하는 **다항식 컨볼루션 연산**에 대한 연산량 감소 필요

NTRU public key cryptosystem

Table 1. Performance analysis of NTRU with existing public key cryptosystems

Algorithm	Message Size (bits)	Key size (bits)	Key generation (ms)	Encryption (ms)	Decryption (ms)
RSA1024	1024	1024	1432	4.28	48.5
ECC168	160	169	65	140	67
NTRU263	416	1841	19.8	1.9	3.5

https://www.kci.go.kr/kciportal/ci/sereArticleSearch/kci?sereArticleSearchBean.artiId=ART002345480)

❖ RLWE를 기반으로 **polynomial ring에서 기본 연산 수행**

- $Z[x]$: Z에 대한 다항식 링 → 정수 계수를 갖는 모든 다항식들의 집합
- $R = Z[x]/(Xn - 1)$: 모든 다항식들의 집합은 ring R 에서 정의

  → 계수가 정수이고, n-1차 다항식 사용 : $a = a_0 + a_1x^1 + \ ... + a_{n-1}x^{n-1}$

❖ 기본 연산

- **Circular Convolution** : 순환 합성곱 : O(NlogN)

  → **다항식 곱셈**에 사용 → 시간 소모가 가장 많은 과정 → 연산량 줄일 필요가 있음

- RSA(modular multiplication), ECC(Elliptic Curve Addition), NTRU(Convolution)

  → Convolution 연산은 기존의 공개키 암호의 연산보다 **암/복호화가 빠르고 효율적**

❖ 격자에서 짧은 벡터를 찾는 어려움(SVP)을 기반으로 안전성을 제공 & 복호화

- 양자 컴퓨팅 공격에도 안전

➢ **빠른 연산 속도 / SW, HW 구현 용이 / 적은 메모리 사용 / 키 생성 쉬움 / 양자 알고리즘 공격에 안전**

NTRU public key cryptosystem

❖ parameter

- $\boldsymbol{n}$(소수), $\boldsymbol{p}, \boldsymbol{q}$($2^x$: 2의 거듭제곱)
  - → $\gcd(\boldsymbol{p}, \boldsymbol{q}) = \boldsymbol{1}$ (p와 q는 소수일 필요 없고 **서로소**)
  - → 공개 파라미터

❖ 4 sets : $\boldsymbol{L_f}$, $\boldsymbol{Lg}$, $\boldsymbol{Lr}$, $\boldsymbol{Lm}$

- $sampling\ f, g, r, m$: randomness → polynomial : 확률론적 알고리즘
- ex) hps2048509 의 sampling : $n = 509, q = 2048$ → 임의의 $4064bits(= 508byte)$
  - 4064bit → mod 3 → 각 byte를 -1, 0, 1으로 변환 (0, 1, 2지만 -1 ≡ 2 mod 3)
  - 509번째 계수는 항상 0으로 set → 다항식 f : 509개의 작은 계수들
- 다항식 만든 후 fisher-Yates shuffle 등의 알고리즘으로 무작위로 섞는 방법이 가장 간단
  - → 다항식이 큰 경우 일정 시간 안에 실행 불가 등의 문제점 존재
  - → NTRU는 shuffle대신 정렬을 사용

*sampling algorithm은 NTRU 변형마다 다름

*확률적 / 무작위 알고리즘 (probabilistic / randomized algorithm)
난수를 발생시켜 진행과정을 결정 (의사난수발생기 사용) → 결과값을 예측하지 못하도록 함
아주 작은 확률로 틀릴 가능성 존재 but 효율적인 알고리즘

NTRU public key cryptosystem

❖ sorting

- 셔플, 정렬 시간 등에서 다항식 길이 등의 정보 노출 가능성 존재
- constant time sorting 방법을 만드는 것이 shuffle보다 쉬움
- 덧셈, 뺄셈, 논리 연산 등의 상수 시간 연산만 사용 가능
- 비밀데이터에 의존하는 분기 사용x, 메모리 접근 시 캐시 주의

*constant time
어떤 문제를 풀이하는데 필요한 수학적 연산 시간이
주어진 입력 자료에 관계 없이 일정할 때의 연산 시간

❖ Hamming Weight

- $r(x), F(x)$ 의 계수들은 **임의로 선택하여 제어 가능** → 곱셈 연산 없이 계산되도록 이진 계수$(\mathbf{0}, \mathbf{1})$사용
- $r(x)$의 가중치 HW(r) → HW$(r) \cdot n$ 번의 연산 필요
- **낮은 hamming weight**를 갖는 $r(x), F(x)$를 사용 → **암/복호화 효율적**

  → HW가 너무 적은 경우 **보안성 저하**

  → IEEE P1361.1 표준 초안에서는 (N, p, q)에 의해 대략적인 HW값이 주어짐

NTRU parameters

❖ 현재 **NIST competition에 parameter set 4개** 정의됨

- hps2048509, hps2048677, hps4096821, hrss701

  → 다항식 계수를 결정하는 n : 509, 677, 821, 701

  → 2의 거듭제곱 q : 2048, 2048, 4096, 8192

  → 모든 set에서 mod 2^{16} 보다 큰 모듈러 축소는 없으므로 16번째 비트의 overflow는 결과에 영향 없음

❖ NIST competition에 설정된 **보안 목표 달성 위해** 선택됨

- 변형들도 대체로 동일하지만 샘플 공간 등 일부 세부 사항이 다름

❖ 매개변수를 조절하여 **보안 강도를 환경에 맞도록 최적화 가능**

NTRU – 기본 연산 : circular convolution (∗)

- convolution
  - 두 함수 중 하나를 역전시켜 이동하면서 다른 함수와의 곱을 더함→ 새로운 함수 생성
- **circular convolution**
  - 주기성을 갖는 신호에서의 convolution
  - **modular** 연산 : **주기성**을 가짐
- n^2번의 정수 곱셈이 필요
  - but **NTRU의 convolution**은 일반적으로 **a, b중 하나가 작은 계수를 갖는 다항식**

    → 곱셈 쉽고, n^2번 필요 없음 → **빠른 연산 가능**

$$c_k = \sum_{i=0}^{k} a_i\, b_{k-i} + \sum_{i=k+1}^{n-1} a_i\, b_{n+k-i} = \sum_{i+j=k \bmod n} a_i\, b_j$$

$$a = a_0 + a_1 x^1 + \ \ldots + a_{n-1} x^{n-1} = [a_0, a_1, \ldots, a_{n-1}]$$

$$b = b_0 + b_1 x^1 + \ \ldots + b_{n-1} x^{n-1} = [b_0, b_1, \ldots, b_{n-1}]$$

$$\mathrm{c(x)} = a(x) * b(x)$$

c_k: c(x)의 계수

NTRU – 기본 연산 : circular convolution (∗)

❖ **modular multiplication of polynomial = circular convolution**

$$c_k = \sum_{i=0}^{k} a_i\, b_{k-i} + \sum_{i=k+1}^{n-1} a_i\, b_{n+k-i} = \sum_{i+j=k \bmod n} a_i\, b_j$$

$$c_k = a_0 b_k + a_1 b_{k-1} \ldots + a_k b_0 + a_{k+1} b_{n-1} + \ldots + a_{n-1} b_{k+1}$$

$ex)\ n = 3$

$$c_1 = a_0 b_1 + a_1 b_0 + a_2 b_2$$

➢ 카라츠바, 톰쿡 등의 알고리즘으로 연산량 감소 가능

↑ 순환 컨볼루션 = 다항식 모듈러 곱셈 →

∗ $n=3$, a와 b는 $n-1$ 차 다항식

$a(x) = 2x^2 + 3x + 5 = [a_0, a_1, a_2] = [5, 3, 2]$

$b(x) = 3x^2 + 4x + 7 = [b_0, b_1, b_2] = [7, 4, 3]$

∗ $a(x)$와 $b(x)$의 곱셈 ⇒ 4차 다항식이 나오는데
mod n 에서 reduction (mod 3)
⇒ $x^4 \to x$, $x^3 \to$ 상수항
⇒ 2차 다항식 "

∗ x^2의 계수 c_2

⇒ – 끼리의 곱의 합 = $a_0 \cdot b_2 + a_1 \cdot b_1 + a_2 \cdot b_0$

∗ x의 계수 c_1

⇒ i) – 끼리 곱의 합 = $a_1 \cdot b_0 + a_0 \cdot b_1$] i + ii
ii) – 끼리 곱의 합 = $a_2 \cdot b_2$
↳ x^4 에서 x로 모듈러 축소됨

∴ $c_1 = a_1 \cdot b_0 + a_0 \cdot b_1 + a_2 \cdot b_2$
(x의 계수) (x^4의 계수)

NTRU – key generation (1)

❖ $parameter$

- n(소수) / p, q → $gcd(p, q) = 1$ / $q = 2^x$: 공개 파라미터

  → $\gcd(p, q)$가 1보다 커지면 안전성 감소 (p가 q를 나누는 경우, e = m이 됨)

❖ $sampling$

- 작은 계수를 갖는 $n-1$차 다항식 f, g를 뽑음 ($f \in Lf, g \in Lg$)
- g는 공개키 생성 시 사용 후 버려짐
- $f * f^{-1} \equiv 1 \ (mod\ p)$, $f * f^{-1} \equiv 1 \ (mod\ q)$

  → F_p, Fq 는 각각 $\boldsymbol{mod\ p}, \boldsymbol{mod\ q}$ 상에서의 $\boldsymbol{f}$의 역원

  → $f = 1 + pF$의 형태로 선택 → $mod\ p$상에서 $f = 1$이 됨 ($0 \equiv p\ mod\ p$)

  → 역원 존재하지 않을 경우 다시 선택

NTRU – key generation (2)

- $private\ key$
  - (f, F_p)
- $public\ key$
  - $h = p * Fq * g\ (mod\ q)$

NTRU – encryption

❖ ***sampling***

- message $\boldsymbol{m}$, random polynomial $\boldsymbol{r}$ 뽑음 ($m \in \mathrm{Lm}$, $\mathrm{r} \in \mathrm{Lr}$)
- r : 비밀데이터, 사용 후 버려짐

❖ ***encryption***

- $\boldsymbol{e} \equiv \boldsymbol{r} * \boldsymbol{h} + \boldsymbol{m} \, (\boldsymbol{mod}\ \boldsymbol{q})$

NTRU – decryption (1)

❖ $computing\ polynomial\ a \equiv f * e\ (mod\ q)$

$\equiv f * (r * h + m)\ (mod\ q)$

$\equiv f * r * h + f * m\ (mod\ q)$

$\equiv f * r * p * F_q * g + f * m\ (mod\ q)$

$\equiv r * p * g + f * m\ (mod\ q)$

❖ **복호화 실패 방지 위해** 다항식 a의 계수들은 $-q/2 \sim q/2$ 사이에 있어야 함 : decryption error x

- **적절한 parameter 선택 시 $-q/2 \sim q/2$ 범위**에 오게 됨

→ $\boldsymbol{q > (6d + 1)p}$ 의 경우 복호화 실패 X

→ $r * p * g + f * m\ (mod\ q) \equiv \boldsymbol{r * p * g + f * m}$ → 정확히 같은 값이므로 복호화 가능

NTRU – decryption (2)

❖ ***decryption***

- $r * p * g + f * m$

  $= r * p * g + (1 + pF) * m$

  $= r * p * g + (1 + pF) * m \, (mod \, p)$

  $\equiv 0 + (1 + 0)m = m$ → 복호화 성공

Q & A