Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.35, NO.1, Feb. 2025 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2025.35.1.11

NTRU+KEM의 이항식 파라미터 제안 및 효율적인 구현*

이 미 라,[1†] 이 승 우,[1] 김 종 현,[2] 박 종 환[3‡]
[1,2]고려대학교 (대학원생, 박사후 연구원), [3]상명대학교 (교수)

New Parameter of NTRU+KEM with Cyclotomic Binomial and Efficient Implementation*

Mira Lee,[1†] Seungwoo Lee,[1] Jonghyun Kim,[2] Jong Hwan Park[3‡]
[1,2]Korea University (Graduate student, Postdoctoral researcher),
[3]Sangmyung University (Professor)

요 약

NTRU+는 삼항식 환 격자 기반 양자 내성 PKE/KEM 기법이다. 다항식 환 격자 연구에서 효율적인 곱셈 연산을 위해 NTT(Number Theoretic Transform)을 적용하는 것이 주목받고 있다. NTRU+KEM은 NTT 적용이 가능한 삼항식 환을 기반으로 설계되었다. 이러한 삼항식 구조는 다양한 안전성 수준의 파라미터를 제공한다. NTRU+KEM의 모듈러스는 복호화 실패율에 영향을 미치는데, 삼항식 환의 구조상 이항식 환에 비해 복호화 실패율이 높으므로 모듈러스를 줄이기 어렵다. 따라서 NTRU+KEM을 이항식 환으로 변형하면 모듈러스의 감소를 기대할 수 있다. 한편, 기존 NTRU+KEM864는 192비트의 안전성을 목표로 설계되었으나 최신 분석에서 비트 안전성이 179로 측정되어 파라미터 셋의 개선 여지가 있다. 본 논문에서는 NTRU+KEM의 NTT 적용이 가능한 이항식 파라미터를 제안한다. 제안 파라미터는 기존의 NTRU+KEM864, NTRU+KEM1152 사이의 키 및 암호문 크기와 복호화 실패율을 유지하면서도, 보안성과 효율성을 향상시킨다. 특히, 제안 파라미터는 218비트 안전성을 달성하며, 키 생성 알고리즘에서 NTRU+KEM864보다 CPU 사이클을 4.3% 절감하여 성능을 개선한다.

ABSTRACT

NTRU+ is a lattice-based quantum-resistant PKE/KEM scheme utilizing cyclotomic trinomials. This paper introduces a new NTRU+KEM parameter based on a cyclotomic binomial to enhance security and efficiency. NTRU+KEM leverages the number theoretic transform(NTT), providing flexible security parameters. However, cyclotomic trinomials have a higher decryption failure probability than cyclotomic binomials, suggesting that transitioning to cyclotomic binomials can reduce the modulus and improve efficiency. The proposed parameter achieves 218-bit security, balances efficiency and decryption failure probability, and reduces CPU cycles in the key generation algorithm by 4.3% compared to NTRU+KEM864.

Keywords: Post-Quantum Cryptography, Key Encapsulation Mechanism, NTRU+

Received(10. 04. 2024), Modified(12. 05. 2024), Accepted(01. 20. 2025)

* 본 논문은 2024년도 한국정보보호학회 하계학술대회에 발표한 우수논문을 개선 및 확장한 것임.

* 본 연구는 2024년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No. 2021-0-00518, 블록체인 데이터 암호화 기반의 프라이버시 보호 기술개발)을 받아 수행된 연구입니다.

† 주저자, mrlee01@korea.ac.kr

‡ 교신저자, jhpark@smu.ac.kr(Corresponding author)

I. 서 론

양자컴퓨터의 상용화가 도래함에 따라 현재 사용 중인 공개키 암호는 향후 그 안전성이 보장되지 않음이 알려져 있다. 이에 미국 국립표준기술연구소(NIST)에서는 양자컴퓨터의 공격에도 안전한 공개키 암호인 양자 내성 암호에 대한 표준화 공모전을 진행하는 등, 세계적으로 양자 내성 암호의 필요성이 대두되어 왔다. 격자 기반 암호는 안전성이 격자 구조 상에서 난제의 어려움에 의존하는 암호로, 양자컴퓨터에도 안전한 것으로 알려져 있을 뿐만 아니라 다른 난제에 기반한 양자 내성 암호에 비하여 비교적 우월한 효율성을 지녀 선호되고 있다.

격자 기반 양자 내성 암호는 효율성을 위해 ring, module 등의 수학적 구조에 기반하여 설계되는 경우가 있는데, 일부 다항식 환 구조에서는 곱셈 고속화 알고리즘인 number theoretic transform (NTT)를 사용할 수 있다[1]. NTT는 다항식 환을 인수분해 하여 얻은 작은 다항식 환에서 연산 수행 후 결과를 결합하는 방식으로 동작하며, 작은 다항식 환에서의 연산 결과는 원래 다항식 환의 연산 결과와 동일하다[2]. 다항식 환의 순환적 구조로 인해 곱셈 연산 결과는 환 내에서 리덕션되며, 다항식 환의 항의 수가 적을수록 연산 과정이 간단해진다. 따라서 Ring 기반 격자 암호 기법에서 NTT를 적용할 때 이항식 환 $R_q = \mathbb{Z}_q[x]/(x^n+1)$이 선호된다. 이항식 환은 항의 개수가 적어 다른 환에 비해 리덕션 과정이 간단하고, 곱셈 결과의 계수가 n개 항의 합으로 구성되어 복호화 실패율이 낮다는 특징을 가진다[2]. 그러나 NTT를 적용하기 위해 이항식 환이 인수분해 가능하도록 환의 차수 n을 2의 거듭제곱으로 설정해야 하므로 사용 가능한 다항식 차수의 간극이 크다.

NTRU는 NIST 표준화 공모전 3라운드에 진출했던 기법으로 소수 n을 차수로 가지는 이항식 환 $R_q = \mathbb{Z}_q[x]/(x^n-1)$을 사용한다[3]. 이 경우 사용할 수 있는 다항식의 차수 n이 2의 거듭제곱 형태인 이항식 환에 비해 밀집되어 있지만, 다항식 환의 인수분해가 필요한 NTT를 적용하기 어렵다. 이러한 문제를 개선하기 위해 2019년에 NTRU 기법의 파라미터를 유연하게 선택할 수 있는 장점을 가지면서 NTT 적용이 가능한 삼항식 환 기반 NTTRU[4], 2023년에 이항식 환에 NTT와 벡터 디코딩으로 효율성을 높인 NEV[6]이 제안되었다.

NTRU+KEM은 NTRU[3]과 NTTRU[4] 등을 기반으로 설계된 기법이다[5]. NTRU+KEM은 NTRU[3]의 average-case 기반 안전성을 worst-case로 엄밀하게 증명하였다. NTTRU[4]의 장점인 NTT를 적용할 수 있고, 다항식 차수 선택의 유연성을 가진다. NTRU+KEM은 삼항식 환을 사용하는데, 삼항식 환은 이항식 환에 비해 항의 개수가 많아 다항식 연산 시 리덕션 과정이 복잡하다. 또한 곱셈 연산 시 그 결과는 n개 이상인 항의 합으로 이루어져 이항식 환을 이용하는 것에 비해 복호화 실패율이 높다[2]. 복호화 실패율을 개선하기 위해 NTRU+KEM의 모듈러스를 크게 설정할 수 있지만, 모듈러스가 커질수록 다항식 계수의 상한값이 증가하게 되므로 키와 암호문을 저장하기 위해 넓은 공간이 요구된다. 정리하면, NTRU+KEM은 삼항식 환을 사용하여 완화된 간극의 다항식 차수를 가지고, NTT를 이용하여 곱셈 연산이 효율적이지만, 이항식 환에 비해 복호화 실패율이 높아 키 및 암호문 크기를 줄이기 어렵다. 기존에 제안되었던 NTRU+KEM[5]의 파라미터 중 NTRU+KEM864는 192비트의 안전성을 요구하는 NIST 레벨 3를 목표로 제안되었으나, 최신 안전성 평가에서는 179비트로 측정되었다. 시간이 지남에 따라 암호 기법의 안전성은 더욱 감소할 가능성이 높으므로, 이에 대한 대책이 필요할 것이다.

본 논문에서는 이항식 환 기반 NTRU+KEM의 새로운 파라미터를 제안하며, 해당 파라미터를 구현하고 분석하여 그 적합성을 입증하고자 한다. 제안 파라미터의 키 및 암호문 크기와 안전성은 기존의 NTRU+KEM864, 1152 사이이며, 복호화 실패율은 두 파라미터에 비해 더 낮게 측정되었다. 특히 새로운 파라미터의 비트 안전성은 NIST의 보안 요구 레벨 3(192비트)를 상회하며, CPU 사이클은 기존 NTRU+KEM864와 비슷한 수준으로 측정되었다. 반면, NTT를 이용해 다항식 환을 인수분해 할수록 연산 효율성이 높아지지만 모듈러스 또한 증가하므로 키 및 암호문 크기가 증가하게 된다. 본 논문에서는 이를 고려하여 모듈러스를 선정하였으나, 이를 구현하기 위해 7차 다항식의 연산이 필요하다. 그럼에도 기존 삼항식 파라미터들보다 모듈러스가 작다는 점을 이용하여 구현 측면에서 계산 효율성을 향상했다. 본 논문을 통해 NTRU+KEM의 파라미터 선택의 유연성을 높이고, 레벨 3의 요구사항을 충족하면서도 효율적인 파라미터를 제안하고자 한다.

II. 배경 지식

2.1 KEM 기법

KEM(Key Encapsulation Mechanism)은 공개키 암호화 방식을 사용하여 비밀키를 안전하게 공유하는 기술이다. 세부 알고리즘은 다음과 같다.

- **Gen**(1^λ): 보안 매개변수(λ)를 입력받아 공개키(pk)와 개인키(sk)를 출력한다.
- **Encap**(pk): 공개키(pk)를 입력받아 비밀키(K)와 암호문(c)을 출력한다.
- **Decap**(sk,c): 개인키(sk)와 암호문(c)을 입력받아 비밀키(K)를 출력한다.

2.2 NTRU+KEM 기법[5]

NTRU+는 KpqC 표준화 공모전 2라운드에 진출한 격자 기반 PKE/KEM기법으로, 양자컴퓨팅 환경에서 난제라고 알려진 RLWE, NTRU 문제의 어려움에 기반한다[5]. NTRU+KEM은 이러한 어려움을 이용해 암호화된 환경에서 안전하게 비밀키를 전달하는 기법이다. NTRU+KEM은 삼항식 환 $R_q=\mathbb{Z}_q[x]/(x^n-x^{n/2}+1)$을 사용하며 다항식 차수 $n\in\{576,768,864,1152\}$일 때 모듈러스 $q=3457$를 제안했다[5]. 세부 알고리즘은 다음과 같다.

- **Gen**(1^λ)
  1) 양의 정수 n에 대해 집합 $\{-1,0,1\}^n$으로 정의된 중심 이항 분포 ψ_1^n에서 f',g를 선택한다.
  2) 비밀키 $f=3f'+1$을 계산한다.
     이때 R_q 상에 f, g의 역원이 존재해야 한다.
  3) 공개키 $h=3\times g\times f^{-1}$를 계산한다.
  4) 공개키와 비밀키 (h,f)를 출력한다.

- **Encap**(pk)
  1) $\{0,1\}^n$에서 m을, ψ_1^n에서 r을 선택한다.
  2) $(K,r)=\mathrm{H}(m)$, $u=\mathrm{G}(r)$을 계산한다.
     $\mathrm{H}(\cdot)$은 랜덤 오라클로 정의된 해시함수이고, $\mathrm{G}(\cdot)$은 입력값에 대해 $\{0,1\}^*\to\{0,1\}^{2n}$인 값을 출력하는 해시함수이다.
  3) $M=\mathrm{SOTP}(m,u)$를 계산한다.
     SOTP (semi-generalized one-time pad)는 입력값 u를 이용하여 m을 중심 이항 분포로 확장하여 출력한다.
  4) 암호문 $c=hr+M$을 계산한다.
  5) 비밀키와 암호문 (K,c)를 출력한다.

- **Decap**(sk,c)
  1) $M'=(cf \bmod q) \bmod 3$를 계산한다.
  2) $r'=(c-M')h^{-1}$를 계산한다.
  3) $m'=\mathrm{Inv}(M',\mathrm{G}(r'))$를 계산한다.
     Inv는 SOTP의 역과정으로 M'을 입력 받아 $\{0,1\}^*$을 만족하는 m'을 출력한다.
  4) $(K,r'')=\mathrm{H}(m')$를 계산한다.
  5) $r'=r''$이면 비밀키 K를 출력하고, 그 외의 경우 오류를 나타내는 $\perp$를 출력한다.

2.3 복호화 실패율

2.3.1 NTRU+KEM의 복호화 실패율

2.2절의 Decap 과정 중 1)의 식을 전개하면 다음과 같다.

$$\begin{aligned}M &= (cf \bmod q) \bmod 3\\ &= ((r\times h+M)\times f \bmod q) \bmod 3\\ &= ((r\times 3gf^{-1}+M)\times f \bmod q) \bmod 3\\ &= (3rg+Mf \bmod q) \bmod 3\\ &= (3rg+M(3f'+1) \bmod q) \bmod 3\\ &= (3(rg+Mf')+M \bmod q) \bmod 3\end{aligned}$$

임의의 정수 i에 대해 $3\times i \bmod 3=0$이므로 M을 복원하기 위해 $cf \bmod 3$을 계산해야 한다. 이 과정에서 R_q의 원소인 cf가 q로 리덕션 되지 않아야 정상적인 복원이 가능하다. 따라서 NTRU+KEM의 평균(average-case) 복호화 실패율은 cf가 q로 리덕션 될 확률이며, q가 홀수이므로 다음과 같이 구할 수 있다.

$$\begin{aligned}\delta &= \Pr\left[|3(rg+Mf')+M|_\infty \le (q-1)/2\right]\\ &\ge \Pr\left[|3(rg+Mf')|_\infty+|M|_\infty \le (q-1)/2\right]\\ &\ge \Pr\left[|3(rg+Mf')|_\infty+1 \le (q-1)/2\right]\\ &\ge \Pr\left[|rg+Mf'|_\infty \le (q-3)/6\right]\end{aligned}$$

복호화 실패율 식에 M이 포함되면 M과 r을 임의로 선택할 수 있는 IND-CCA 공격자가 값들을 조작하여 복호화를 실패하게 하거나 추가 정보를 얻어내는 취약점이 존재한다. 이러한 문제를 해결하고자 NTRU+KEM는 입력값이 무작위한 성질을 갖게 하면서 중심 이항 분포를 만족하는 값으로 출력하는 SOTP 함수를 이용한다. 즉, NTRU+KEM은 SOTP 함수를 이용해 M의 분포를 중심 이항 분포로 확장했고, 최악의 경우(worst-case) 복호화 실패율 δ'에 대해 $\delta \approx \delta'$를 만족한다는 것을 증명했다. 최악의 경우 복호화 실패율은 다음과 같이 구할 수 있으며, ψ_{R}은 랜덤한 공간 R에 연관된 분포이고 $\mathcal{M}'$은 M'의 공간을 의미한다[5].

$$\delta' = \delta + \|\psi_{\mathrm{R}}\| \cdot \left(1+\sqrt{(\ln|\mathcal{M}'| - \ln\|\psi_{\mathrm{R}}\|)/2}\right)$$

2.3.2 다항식 환에 따른 복호화 실패율 비교

2.3.1절의 복호화 실패율을 구하는 과정은 다항식 환에 대한 의존 없이 이루어지지만, 복호화 시 다항식 형태로 출력되는 비밀키 K의 계수 분포는 다르다. 본 절에서는 이러한 차이점이 복호화 실패율에 미치는 영향을 다항식 차수가 $n=4$인 다항식 환 R_q에서 벡터-행렬 곱셈을 통해 보이고자 한다. 다항식 $a(x)=\sum_{i=0}^{3} a_i x^i$, $b(x)=\sum_{i=0}^{3} b_i x^i$, $c(x)=\sum_{i=0}^{3} c_i x^i$에 대해 $a(x)\times b(x)=c(x)$는 다음을 만족한다.

- 이항식 환 $R_q = \mathbb{Z}_q[x]/(x^4+1)$

$x^4=-1$이므로 $a(x)\times b(x)=c(x)$는 다음과 같이 벡터-행렬 곱셈으로 나타낼 수 있다.

$$\begin{bmatrix} a_0 & -a_3 & -a_2 & -a_1 \\ a_1 & a_0 & -a_3 & -a_2 \\ a_2 & a_1 & a_0 & -a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \cdot \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

위 식을 전개하면 n개 항의 합으로 이루어지는데, $c_3=(a_3b_0+a_1b_2)+(a_2b_1+a_0b_3)$처럼 2개씩 묶으면 $n/2$개의 독립적인 확률 변수들의 합이 된다. 따라서 위 곱셈의 분포는 $a,b,a',b' \in \{-1,0,1\}$일 때 $c=ab+a'b' \in \{0,\pm1,\pm2\}$를 만족하는 $n/2$개의 확률 변수들의 합이므로 $c_i \in \{0,\pm1,\pm2\}^{n/2}$이다.

Table 1. Probability distribution of $c=ab+a'b'$, where $a,b,a',b' \leftarrow \{-1,0,1\}$

±2	±1	0
1/64	3/16	19/32

Table 1.은 $ab+a'b'$의 분포와 확률을 나타낸다.

- 삼항식 환 $R_q = \mathbb{Z}_q[x]/(x^4-x^2+1)$

$x^4=x^2-1$이므로 $a(x)\times b(x)=c(x)$는 다음과 같이 벡터-행렬 곱셈으로 나타낼 수 있다.

$$\begin{bmatrix} a_0 & -a_3 & -a_2 & -a_1-a_3 \\ a_1 & a_0 & -a_3 & -a_2 \\ a_2 & a_1+a_3 & a_0+a_2 & a_1 \\ a_3 & a_2 & a_1+a_3 & a_0+a_2 \end{bmatrix} \cdot \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

위 식을 전개하면 $n/2$를 기준으로 계수에 더해지는 항의 개수가 달라진다. $[n/2, n-1]$일 때 $c_3=(a_3b_0+b_2(a_1+a_3))+(a_2b_1+b_3(a_0+a_2))$처럼 2개씩 묶으면 $c=ab+b'(a'+a) \in \{0,\pm1,\pm2,\pm3\}$인 $n/4$개의 독립적인 확률 변수들의 합이 되고, $[0, n/2-1]$일 때 차수 i에 따라 $ab+a'b'$인 독립된 확률 변수 $(n+i)/4$개, $ab+b'(a+a')$인 독립된 확률 변수 $(n-i)/4$개의 합으로 이루어진다. Table 2.는 $ab+b'(a+a')$의 확률을 나타낸다.

2.3.1절에서 NTRU+KEM에서 복호화에 성공하기 위해 Decap 알고리즘의 입력값인 $c=hr+M$의 계수들이 모듈러스로 리덕션 되지 않아야 했다. 이때 NTRU+KEM은 다항식 환에서 동작하므로 Encap 알고리즘의 출력값은 다항식이다. 따라서 앞서 구한 복호화 실패율은 c의 계수 하나에 해당한다. 다항식 환의 차수를 n으로 설정했다면 암호문 c의 항이 n개이므로 복호화에 성공하는 경우는 n개의 계수들이 q보다 작을 확률과 같다. 따라서 복호화 실패율은 $1-(\text{복호화 성공 확률})^{\text{항의 개수}}$이다. 구현 시 저장 공간을 절약하기 위해 계수들의 범위를 $[-(q-1)/2, (q-1)/2]$로 설정할 수 있다. 즉, 각 항의 계수들이 $[-(q-1)/4, (q-1)/4]$에 속할 확률을

Table 2. Probability distribution of $c=ab+b'(a+a')$

±3	±2	±1	0
1/128	1/32	23/128	9/16

α라고 하면 NTRU+KEM의 복호화 실패율은 $(1-(\alpha)^n)$으로 구할 수 있다.

앞선 분석에서 이항식 환에서의 곱셈 시 계수 범위는 [−2,2]이고, 삼항식 환에서의 범위는 [−3,3]이다. 즉, 다항식 곱셈 시 이항식 환에서의 계수 분포가 삼항식 환에 비해 좁은 것을 확인할 수 있다. NTRU+KEM의 복호화 실패율은 모듈러스에 의해 결정되므로, 동일한 모듈러스를 사용하는 경우 계수 범위가 분포가 넓은 삼항식 환을 사용할 때 복호화 실패율이 상대적으로 높게 측정된다. 따라서 이항식 환은 삼항식 환에 비해 복호화 실패율이 낮으므로 모듈러스를 작게 설정할 수 있다.

2.4 Number theoretic transform (NTT)

NTT는 DFT(Discrete Fourier Transform)을 소수 q에 대한 모듈러스 체계인 $\mathbb{Z}/q$로 변형한 것이다. NTT는 일부 격자 기반 양자 내성 암호 기법에서 곱셈 연산을 효율적으로 수행하기 위해 사용된다. 양의 정수 n를 다항식 차수로 가지는 $f(x)$에 대해 다항식 환을 $R_q=\mathbb{Z}_q[x]/f(x)$로 정의하고, $f(x)=x^n+1$일 때 NTT 과정은 다음과 같다.

1) $\mathbb{Z}_q$에서 $w^{2n}\equiv 1 \bmod q$인 w를 찾는다.
2) $m<2n$인 m에 대해 $w^m\neq 1 \bmod q$이면 $w^n\equiv -1 \bmod q$이므로 $1\equiv -w^n \bmod q$이다.
3) $f(x)=x^n+1=x^n-w^n$을 인수분해 하면 $R_q=\mathbb{Z}[x]/(x^{n/2}+w^{n/2})(x^{n/2}-w^{n/2})$이다.

이때 1)을 만족하는 w가 항상 존재하지는 않지만, 페르마 소정리를 이용하면 모듈러스 q에 대해 $w^{2n}\equiv 1 \bmod q$를 만족하는 q, w를 설정할 수 있다. $\mathbb{Z}_q$의 생성원 g와 소수 q에 대해 $g^{q-1}\equiv 1 \bmod q$이므로 임의의 정수 k에 대해 $q=2nk+1$이면 $g^{q-1}=g^{2nk}\equiv 1 \bmod q$이다. 따라서 $q=2nk+1$, $w=g^k$으로 설정하면 $g^{q-1}=(g^k)^{2n}=w^{2n}=1 \bmod q$이므로 $f(x)$를 인수분해 할 수 있으므로 NTT를 이용할 수 있다.

NTT를 이용했을 때 곱셈 연산의 시간 복잡도는 다음처럼 구할 수 있다. 예를 들어 $f(x)=x^n+1$일 때 $q=2nk+1$로 설정하면 $f(x)=x^n-w^n$이므로 $f(x)$를 $\log n$번의 레이어로 인수분해 할 수 있고, 각 레이어마다 n번의 곱셈 연산이 발생하므로 시간 복잡도는 $O(n\log n)$이다.

$f(x)=x^n+1$일 때 다항식의 차수가 n이면 NTT를 이용해 최대 $\log n$개의 레이어를 인수분해 할 수 있다. 이때 연산들은 작은 다항식 환에서 이뤄지기 때문에 연산 효율성이 높아지지만 모듈러스 q 값이 커지게 된다. NTRU+KEM에서 키 및 암호문 크기는 $\log q$에 비례하기 때문에 키를 저장하거나 암호문을 전송할 때 더 많은 자원이 요구되게 된다. 따라서 기법의 전반적인 효율성을 높이기 위해 q를 줄여야 한다. 이를 위해 NTT 수행 시 $f(x)$의 인수분해 레이어 수를 줄인다면 q를 감소시킬 수 있다. 구체적으로, 임의의 양의 정수 i에 대해 $w^{n/2^i}\equiv -1 \bmod q$로 설정하면 $\log n-i$개의 레이어만을 인수분해 할 수 있다.

2.5 다항식 분할(spilt)과 역원 계산 방법[6]

NEV[6]은 NTRU[3]의 암호화 과정을 최적화시킨 기법이다. 특히 이항식 환 $R_q=\mathbb{Z}_q[x]/(x^n+1)$를 사용하며 환의 다항식을 분할한 뒤 NTT를 적용함으로써 모듈러스를 줄이는 방법을 제안하였다. NEV[6]의 다항식 분할 방법은 다음과 같다.

$x^2=y$일 때 이항식 환 $\mathbb{Z}_q[x]/(x^4-w)$ 상의 연산은 $\mathbb{Z}_q[y]/(y^2-w)$에서 계산할 수 있다. 예를 들어 다항식 $a(x)$은 다음과 같이 나타낼 수 있다.

$$\begin{aligned} a(x) &= a_0+a_1x+a_2x^2+a_3x^3 \\ &= (a_0+a_2y)+(a_1+a_3y)x \end{aligned}$$

위 식에서 $a_0(y):=a_0+a_2y$, $a_1(y):=a_1+a_3y$라고 하면 $a(x)=a_0(y)+a_1(y)x$와 같고, $a(x)$는 분할된 다항식 환 $\mathbb{Z}_q[y]/(y^2-w)$의 다항식들로 계산된다. 따라서, 다항식을 분할하면 고차 다항식을 저차 다항식의 연산으로 변환할 수 있어, 계산의 복잡도를 줄이고 효율성을 높일 수 있다. 특히, 분할된 다항식들은 독립적으로 계산될 수 있으므로 병렬 처리에도 유리하다. 본 논문에서는 NEV[6]의 다항식 분할 방법을 이용하여 7차 다항식의 역원을 구할 때 발생하는 연산 비용을 효과적으로 줄이고자 하였다.

다항식의 차수가 커질수록 다항식의 역원을 구하기 위해 고려해야 하는 요소가 증가함에 따라 그 과

정이 복잡해진다. 이때 다항식 분할 방법을 이용하면 보다 효율적으로 다항식의 역원을 구할 수 있다. 다항식 분할 방법을 이용해 $a(x)$의 역원 $b(x)$을 구하는 과정은 다음과 같다.

$$
\begin{aligned}
&a(x)\cdot b(x)=1\\
&(a_0(y)+a_1(y)x)(b_0(y)+b_1(y)x)=1\\
&\begin{bmatrix}a_0(y) & a_1(y)y\\ a_1(y) & a_0(y)\end{bmatrix}\begin{bmatrix}b_0(y)\\ b_1(y)\end{bmatrix}=\begin{bmatrix}1\\ 0\end{bmatrix}\\
&\begin{bmatrix}b_0(y)\\ b_1(y)\end{bmatrix}=\frac{1}{a_0(y)^2-a_1(y)^2y}\begin{bmatrix}a_0(y) & -a_1(y)y\\ -a_1(y) & a_0(y)\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix}\\
&\qquad=\frac{1}{a_0(y)^2-a_1(y)^2y}\begin{bmatrix}a_0(y)\\ -a_1(y)\end{bmatrix}
\end{aligned}
$$

위 식에서 $b(x)$를 구하려면 새롭게 파생된 역원을 계산해야 한다. $A(y):=a_0(y)^2-a_1(y)^2y$라고 하면 다음과 같이 식을 전개할 수 있다.

$$
\begin{aligned}
A(y)&=a_0(y)^2-a_1(y)^2y\\
&=a_0^2+a_2^2y^2-2a_1a_3y^2+(2a_0a_2-a_1^2-a_3^2y^2)y
\end{aligned}
$$

위 식은 $\mathbb{Z}_q[y]/(y^2-w)$의 다항식으로 $y^2=w$이다. $A_0:=a_0^2+a_2^2w-2a_1a_3w$, $A_1:=(2a_0a_2-a_1^2-a_3^2w)y$로 정의하면 새로운 1차 다항식을 얻을 수 있다. 즉, $A(y)=A_0+A_1y$의 역원 $B(y)=B_0+B_1y$를 구해 위 식에 대입하면 원래 구하고자 했던 $a(x)$의 역원 $b(x)$를 구할 수 있다. $A(y)$의 역원 $B(y)$을 구하는 과정은 다음과 같다.

$$
\begin{aligned}
&A(y)\cdot B(y)=1\\
&(A_0+A_1y)(B_0+B_1y)=1\\
&\begin{bmatrix}A_0 & A_1w\\ A_1 & A_0\end{bmatrix}\begin{bmatrix}B_0\\ B_1\end{bmatrix}=\begin{bmatrix}1\\ 0\end{bmatrix}\\
&\begin{bmatrix}B_0\\ B_1\end{bmatrix}=\frac{1}{A_0^2-A_1^2w}\begin{bmatrix}A_0 & -A_1w\\ -A_1 & A_0\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix}\\
&\qquad=\frac{1}{A_0^2-A_1^2w}\begin{bmatrix}A_0\\ -A_1\end{bmatrix}
\end{aligned}
$$

다항식 분할을 이용하면 다항식 역원 계산 과정의 복잡도를 낮출 수 있다. 본 논문에서 요구하는 다항식 역원을 구하기 위해 2가지 경우를 고려할 수 있으며, 이에 대해 4.2절에서 관련된 분석을 제공한다.

2.6 NTRU 안전성 분석 방법

본 논문에서는 NTRU+KEM의 안전성 분석을 위해 격자 추정기(estimator)[9]를 사용하였다. NTRU+KEM 기법이 기반으로 하는 LWE 문제와 NTRU 문제에 대한 안전성 수준을 측정할 수 있다.

2016년, Albercht 등[10], Cheon 등[11]에 의해 NTRU 격자 기반 문제가 과도 확장 상태(overstretched regime)일 때 취약하다는 논의가 이루진 바 있다. 과도 확장 상태(overstretched regime)란 격자 기저 축소 알고리즘을 이용해 비밀키를 복구하거나 밀집 하위 격자(dense sublattice)를 쉽게 발견할 수 있는 상태를 의미한다. 이때 비밀키 복구가 어려운 상태를 표준 상태(secure regime)라 하고, 표준 상태와 과도 확장 상태를 구분하는 기준점을 피로 지점(fatigue point)라고 한다. 피로 지점은 격자의 강도가 약화되는 순간이며 공격에 취약하게 되는 전환점이 된다. 2021년, Ducas 등[12]은 다항식 차수 n과 모듈러스 q에 대해 $q\approx 0.004\times n^{2.484}$일 때 피로 지점이 됨을 증명하였다. 즉, NTRU 문제 기반 격자 암호 기법에서 모듈러스가 $q\le 0.004\times n^{2.484}$를 만족할 때 안전성이 보장된다.

2.7 Ring 연산 구현 알고리즘

2.7.1 Montgomery 리덕션 알고리즘[7]

Montgomery 리덕션은 큰 수의 모듈러 연산을 나눗셈 대신 곱셈과 비트 연산만으로 처리하는 알고리즘이다. 모듈러스 q, $q<R$인 2의 거듭제곱수 R

Algorithm 1. Montgomery reduction

Input: Integers R, q with $\gcd(R,q)=1$,
integer $q'\in[0,R-1]$ such that
$qq'\equiv -1 \bmod R$, integer $T\in[0,Rq-1]$
Output: Integer $S\in[0,q-1]$
such that $S\equiv TR^{-1} \bmod q$
1) $m\leftarrow((T \bmod R)\,q) \bmod R$
2) $t\leftarrow(T+mq)/R$
3) if $t\ge q$ then return $S\leftarrow t-q$
4) else return $S\leftarrow t$

Fig. 1. Montgomery reduction algorithm

과 정수 x에 대해 Montgomery 형식으로 변환한 $xR \bmod q$을 입력받아 연산을 수행하고 $x \bmod q$를 반환한다. Fig. 1. 알고리즘의 2)에 R로 나누는 연산이 있는데, R은 2의 거듭제곱수이므로 해당 나누기 연산은 비트 연산으로 구현할 수 있다.

2.7.2 Barret 리덕션 알고리즘[8]

Barret 리덕션은 큰 수의 모듈러 연산을 효율적으로 계산하는 알고리즘으로 사전 계산된 값을 이용해 모듈러 연산에 필요한 나눗셈 대신 곱셈 연산을 이용해 연산 속도를 높인다. Barret 리덕션은 Montgomery 리덕션 알고리즘에 비해 필요한 곱셈 연산이 많지만, 입력값이 특정 형식을 만족하지 않아도 된다는 특징이 있다. 따라서 Barret 리덕션은 덧셈, 곱셈 연산의 마지막에 사용되거나 리덕션의 입력값이 Montgomery 형식이 아닐 때 사용된다.

Algorithm 2. Barrett reduction

Input: Integer q, integer $a \in [0, 2q)$
integer k such that $2^k > q$ and
representing the number of bits in q
Output: Integer $S \in [0, q-1]$
such that $S \equiv a \bmod q$
1) const $v := ((1 \ll (k+1)) + q/2)/q$
2) $t = (v \times a + (1 \ll k)) \gg (k+1)$
3) $t^* = q$
4) return $S \leftarrow a - t$

Fig. 2. Barrett reduction algorithm

III. 제안 파라미터

3.1 이항식 파라미터 선정

본 논문에서 제안하는 새로운 파라미터는 이항식 환 $R_q = \mathbb{Z}_q[x]/(x^n+1)$에서 NTT 적용이 가능하고, 기존의 삼항식 파라미터와 비슷한 암호문 크기와 안전성을 가지는 파라미터이다. 이러한 파라미터를 제안함으로써 NTRU+KEM의 파라미터 선택의 자유도를 높이고자 한다. 다항식의 차수 n은 이항식 환에서 NTT를 적용하기 위해 2의 거듭제곱 형태를 만족하면서, 기존에 제안되었던 삼항식 파라미터 중 다항식 차수 $n = 864, 1152$ 사이에 존재하는 1024로 선정하였다. NTT를 적용하려면 모듈러스 q는 양의 정수 k에 대해 $q = 2nk+1$을 만족하는 소수이어야 한다. 이때 NTRU+KEM의 암호문의 크기는 $\log q$에 비례하므로 암호문 크기를 줄이기 위해 2.4절에 따라 q값을 조절할 수 있다. 다음은 NTT 적용 레이어를 조절했을 때 사용 가능한 q의 목록이다.

1) $\log n$개의 레이어를 나누는 경우
$q = 2nk+1$, 12289, 18433, …
2) -1개의 레이어를 나누는 경우
$q = nk+1$, 12289, 13313, …
3) -2개의 레이어를 나누는 경우
$q = 1/2 \times nk+1$, 7681, 10753, …
4) -3개의 레이어를 나누는 경우
$q = 1/4 \times nk+1$, 257, 769, 3329, 7681, …

Table 3.은 앞서 계산한 모듈러스 q에 대해 공개키, 암호문, 비밀키의 크기와 복호화 실패율을 정리한 표이다. 이때 복호화 실패율은 NTRU+KEM[5]에서 증명한 최악의 경우(worst-case) 복호화 실패율을 의미한다. Table 3.을 통해 모듈러스 q가 키 및 암호문 크기와 복호화 실패율에 영향을 미치는 것을 확인할 수 있다. 특히 $q = 257$인 경우 복호화 실패 확률이 1에 가까워지고, q가 일정 이상으로 커지면 복호화 실패율에 큰 변화가 없다. $q = 3329$인 경우 암호문 크기와 복호화 실패율이 절충되었음을 알 수 있고, $q = 7681$인 경우 복호화 실패율은 매우 작지만 암호문 크기가 3329일 때와 큰 차이가 없었기 때문에 두 가지 파라미터를 제안 후보로 선정하였다.

Table 3. Public key, ciphertext and secret key sizes, and worst-case decryption failure rate for binomial parameters with $n = 1024$

	q	pk (byte)	ct (byte)	sk (byte)	$\log_2 \delta$
$\log n$ layer	11289	1792	1792	3584	-1445
	18433	1920	1920	3840	-1445
-1 layer	11289	1792	1792	3584	-1445
	13313	1792	1792	3584	-1445
-2 layer	7681	1664	1664	3328	-1445
	10753	1792	1792	3584	-1445
-3 layer	257	1152	1152	2304	≈ 0
	769	1280	1280	2560	-15
	3329	1536	1536	3072	-418
	7681	1664	1664	3328	-1445

다음의 Table 4.는 앞서 선택한 두 가지 이항식 파라미터와 기존의 삼항식 파라미터의 공개키, 암호문, 비밀키의 크기와 최악의 경우 복호화 실패율을 비교하였다.

Table 2.에서 기존의 삼항식 파라미터와 새로운 파라미터 후보를 비교했을 때, $q=3329,7681$인 경우 기존에 제안되었던 NTRU+KEM864와 1152 사이의 공개키, 비밀키, 그리고 암호문 크기를 가지는 것을 확인할 수 있다. 특히 $q=3329$일 때 기존 제안되었던 모듈러스 q값보다 작으며 복호화 실패율도 더 낮게 측정되었다. $q=7681$일 때는 복호화 실패율이 매우 낮지만, 모듈러스 값이 커짐에 따라 $q=3329$일 때보다 공개키, 비밀키, 그리고 암호문의 크기가 큰 것을 확인할 수 있었다.

Table 4. Comparison of key, ciphertext sizes, and worst-case decryption failure between trinomial and new parameter candidates

Scheme	q	pk (byte)	ct (byte)	sk (byte)	$\log_2\delta$
NTRU+ KEM576	3457	864	864	1728	-487
NTRU+ KEM768	3457	1152	1152	2304	-379
NTRU+ KEM864	3457	1296	1296	2592	-340
NTRU+ KEM1152	3457	1728	1728	3456	-260
New parameter candidates	3329	1536	1536	3072	-418
	7681	1664	1664	3328	-1445

3.2 안전성

NTRU+KEM[5]의 안전성 증명은 삼항식 환에 대한 의존 없이 이루어졌기 때문에 이항식 환에서도 안전성 증명이 유지된다. 구체적인 안전성 분석을 위해 LWE 문제와 NTRU 문제에 대한 안전성 수준을 측정할 수 있는 격자 추정기(estimator)[9]를 사용하였다. NTRU+KEM 기법에 사용되는 모든 다항식(r, g, M, f')의 계수가 중심 이항 분포 ψ_1^n를 따른다는 점을 고려하여 추정기의 인자를 설정하였고, LWE 문제와 NTRU 문제에 대해 안전성을 측정하였다. 다음의 Table 5.은 기존의 삼항식 파라미터와 새로 제안하는 이항식 파라미터의 안전성을 LWE 문제와 NTRU 문제에 대해 측정한 결과이다.

Table 5. Comparison of bit security between NTRU+KEM and our proposal new parameter candidates

Scheme	q	Classical		Quantum	
		LWE	NTRU	LWE	NTRU
NTRU+ KEM576	3457	111	115	101	105
NTRU+ KEM768	3457	156	164	143	149
NTRU+ KEM864	3457	179	189	164	172
`NTRU+ KEM1152	3457	248	266	228	241
New parameter candidates	3329	218	232	200	211
	7681	199	211	182	192

Table 4.에서 제안 파라미터들은 LWE, NTRU 문제에 대해 NTRU+KEM864보다 비트 안전성이 높고, NIST 안전성 레벨 3(192비트 안전성)의 안전성을 충족한다. 제안 파라미터 후보들을 비교했을 때, $q=7681$의 경우 3329에 비해 모듈러스가 크기 때문에 비트 안전성이 낮게 측정되었다. 3.1절에서 두 가지 파라미터를 후보로 고려했으나, $q=7681$는 3329에 비해 키와 암호문 크기뿐만 아니라 안전성에도 이점을 가지지 않는 것으로 확인된다. 따라서 본 논문에서는 적절한 키 및 암호문 크기와 복호화 실패율을 가지면서, 192비트 안전성을 충족하는 $n=1024$, $q=3329$를 제안한다. 또한 제안 파라미터는 2.6절의 $q \le 0.004 \times n^{2.484}$ 관계를 만족한다.

IV. 제안 파라미터 구현

4.1 리덕션 알고리즘과 효율적인 구현

2.5절의 리덕션 알고리즘의 최적화를 위해 두 가지 방안을 고려할 수 있다. 첫 번째는 모듈러스 연산 결과의 절댓값을 줄이는 것이다. 모듈러스가 q일 때 일반적인 연산 결과는 $t = T \bmod q \in [0, q-1]$이나, 범위를 $t \in [-(q-1)/2, (q-1)/2]$로 변환하면 결과값의 절댓값이 감소한다. 이는 연산에서 처리해야 할 수의 크기를 줄여 메모리 사용량을 절감하고 계산 효율성을 높이는 데 도움이 된다. 두 번째는 코드에서 조건문 사용을 지양하는 것이다. 컴퓨터의 CPU는 파이프라인 구조를 통해 명령어를 빠르게 처리한다. 하지만 조건문이 포함되면 프로그램의 흐름이 분기

(branch)되어 CPU의 분기 예측이 실패할 수 있다. 이 경우 파이프라인을 재설정해야 하며, 이는 성능 저하로 이어질 수 있다. 따라서 조건문의 사용을 최소화하여 파이프라인의 효율성을 유지하는 것이 중요하다. 두 가지 방안을 적용한 Montgomery 리덕션 알고리즘은 다음과 같다.

Fig. 3.에서 입력값과 출력값 범위의 절대값을 줄였다. Fig. 1.과 차이점 중 하나는 조건문을 사용하지 않는 것인데, Fig. 1.의 2)에서 $T+mq$는 Fig. 3.의 2)에서 $T-mq$으로 바뀌 중간 연산 값이 $(-q/2,q/2)$이 되도록 하였다. 이때 입력값의 자료형은 int_32이지만 m의 자료형은 int_16으로 구현하였다. 이는 $R=2^{16}$일 때 $Tq' \bmod R$ 연산 결과와 동일하도록 구현한 것으로, $m \in [-(R-1)/2,(R-1)/2]$를 만족한다. $T-mq \in [-(q-1)/2,(q-1)/2]$이므로 중간 연산 값이 $(-q/2,q/2)$안에 속하기 때문에 조건문이 사용되지 않는다.

리덕션 알고리즘을 사용하면 나눗셈 없이 모듈러스 연산이 가능하지만, 추가 연산이 발생하기 때문에 최대한 사용을 줄이는 것이 좋다. 입력값의 범위가 $T \in [-(Rq-1)/2,(Rq-1)/2]$이면 중간 연산 결과는 $t \in [-(q-1)/2,(q-1)/2]$이므로 조건문을 사용하지 않을 범위$(-q/2,q/2)$에 속한다. 하지만 이는 근소한 차이를 가지므로 안정적인 결과를 출력하기 위해 입력값의 최대 범위를 1/2로 제한하는 것이 바람직하다. 실제 구현에서 T는 두 정수의 곱셈 결과이므로 최대 범위를 $(1/2)^2$로 설정하였다. 리덕션 알고리즘에서 q는 NTRU+KEM의 모듈러스와 같으므로 두 정수 $a,b \in [-(q-1)/2,(q-1)/2]$와 곱셈 수 k에 대해 $T=ab \in [-(q-1)^2/4,(q-1)^2/4]$은 다음을 만족한다.

Algorithm 3. New Montgomery reduction

Input: Integers R, q with $\gcd(R,q)=1$,
integer $q' \in [-(R-1)/2,(R-1)/2]$
such that $qq' \equiv -1 \bmod R$,
integer $T \in [-(Rq-1)/2,(Rq-1)/2]$
Output: Integer $S \in [-(q-1)/2,(q-1)/2]$
such that $S \equiv TR^{-1} \bmod q$, int_32
1) int_16 $m \leftarrow Tq'$
2) int_32 $t \leftarrow (T-mq) \gg logR$
3) return $S \leftarrow t$

Fig. 3. New Montgomery reduction algorithm

$$|k \cdot T| \le k \times \frac{(q-1)^2}{4} \le \frac{Rq-1}{8}$$

$$\therefore k = \frac{2^{16} \times 3329 - 1}{2} \times \frac{1}{(3329-1)^2} \simeq 9.85$$

따라서 이항식 환에서 한 번의 Montgomery 리덕션을 수행할 때 최대 9개의 곱셈의 합을 입력받아도 정상적으로 리덕션 결과가 출력된다. Table 6.은 Montgomery 리덕션 알고리즘 입력값의 범위를 고려하지 않았을 때와 최대한 고려했을 때 7차 다항식의 곱셈 시 Montgomery 리덕션 함수 호출 횟수와 CPU 사이클을 비교한다.

Table 6.에 따르면 입력값의 범위를 고려하여 Montgomery 리덕션을 최소한으로 사용하였을 때 함수 호출 횟수와 CPU 사이클이 더 낮게 측정됨을 확인할 수 있다. 입력값을 고려하지 않은 경우는 곱셈 8×8번과 환 연산 7번으로 총 71번, 입력값을 최대로 고려한 경우는 곱셈 8번과 환 연산 7번으로 총 15번 호출되었다. 따라서 Montgomery 리덕션 함수의 입력값 범위는 최적화 요인으로 작용하며, 전체 기법의 성능에 영향을 미치는 것을 알 수 있다.

Table 6. Comparison of function call count and CPU cycles for 7th-degree polynomial multiplication using Montgomery reduction

	Without input range consideration	With input range consideration
Maximum input size	1	8
Function call count	71	15
CPU cycles (cycles)	806	147

4.2 7차 다항식 역원 연산

본 절에서는 2.4.2절의 다항식 분할 방법[6]을 이용하여 다항식의 역원을 구하는 효율적인 조합에 대해 분석한다. 이항식 환 $\mathbb{Z}_q[x]/(x^n+1)$이 제안 파라미터 $n=1024$, $q=3329$를 사용할 때, NTT를 적용하면 $\mathbb{Z}_q[x]/(x^8-w)$에서 연산이 이루어지며, w는 어느 다항식에서 인수분해 되었는지에 따라 값이 달라진다. 본 절에서는 대표적인 하나의 이항식 환만

을 다루며, 해당 이항식 환은 더 이상 인수분해 되지 않게 설정하였기 때문에 $w \neq -1 \bmod q$이다. 다항식을 분할하는 방법으로는 다음의 두 가지가 있다.

1) $\mathbb{Z}_q[x]/(x^8-w)$ 상의 원소를 $x^2=y$에 대해 $\mathbb{Z}_q[y]/(y^4-w)$를 만족하도록 분할한다.
2) $\mathbb{Z}_q[x]/(x^8-w)$ 상의 원소를 $x^2=y$에 대해 $\mathbb{Z}_q[y]/(y^4-w)$로 분할하고, $y^2=z$에 대해 $\mathbb{Z}_q[z]/(z^2-w)$를 만족하도록 분할한다.

첫 번째 방법은 7차 다항식 $a(x)$을 3차 다항식 $a_0(y)+a_1(y)x$로 분할하여 4×4 정사각행렬을 이용해 역원을 구한 다음 최종 역원을 구하는 순서로 이루어진다. 두 번째 방법은 7차 다항식을 3차 다항식으로 분할한 다음, 1차 다항식으로 분할하여 2×2 정사각행렬을 이용해 역원을 구해 거꾸로 거슬러 올라가며 최종적으로 역원을 구한다. Table 7.은 역원을 구하는 두 가지 방법에 대해 리덕션 함수 호출 횟수와 CPU 사이클을 비교한다.

첫 번째 방법은 두 번째 방법에 비해 높은 차수의 연산이 수반되지만 리덕션 함수들이 상대적으로 덜 호출되는 것을 확인할 수 있다. Table 7.은 높은 차수에서 발생하는 복잡한 연산에 비해 리덕션 함수의 호출량이 성능에 영향을 미침을 보여준다.

Table 7. Comparison of function call count and CPU cycles for 7th-degree polynomial inverse using Montgomery reduction and Barret reduciton

	Method 1	Method 2
Montgomery call count	53	61
Barret call count	2	0
CPU cycles (cycles)	504	519

4.3 성능

본 절에서는 기존 삼항식 파라미터와 새로운 이항식 파라미터의 성능을 비교하고, 이를 통해 제안 파라미터의 효율성을 입증하고자 한다.

Table 8.은 기존에 제안되었던 NTRU+KEM의 삼항식 파라미터들과 본 논문에서 제안하는 이항식 파라미터로 NTRU+KEM의 세 가지 알고리즘을 구현하여 측정한 CPU 사이클을 비교한 결과를 나타낸다. 또한 NTRU+KEM의 주요 연산 함수들의 세부 분석을 위해 Table 9.에서 Montgomery 리덕션 함수 호출량을 비교하고, Table 10.에서 CPU 사이클 값을 비교한다. Table 8.과 Table 10.의 CPU 사이클 값은 10만 번 반복된 연산의 평균값으로 측정하였다.

Table 8.에서 NTRU+KEM864와 비교했을 때 제안 파라미터는 Encap, Decap에서 CPU 사이클이 높게 측정되었다. 이는 3.2절에서 언급된 바와 같이, 높은 보안 강도를 확보하기 위한 추가 비용으로 해석할 수 있다. 반면, 제안 파라미터는 Keygen 연산에서 NTRU+KEM864 대비 약 4.3% 향상된 성능을 보였다. 더 자세히 분석하기 위해 Table 9.

Table 8. Comparison of CPU cycles for key generation, encapsulation, and decapsulation between NTRU+KEM and our proposal (cycles)

Scheme	Keygen	Encap	Decap
NTRU+ KEM576	183417	90252	106074
NTRU+ KEM768	205028	108257	130324
NTRU+ KEM864	246053	127902	153089
NTRU+ KEM1152	386243	178179	215054
Our proposal	235437	133877	169915

Table 9. Comparison of Montgomery reduction function call counts for Keygen algorithm between NTRU+KEM and our proposal (calls)

Scheme	NTT	Mult	Inv	NTT inv
NTRU+ KEM576	2304	1008	4176	2880
NTRU+ KEM768	3072	1344	5568	3840
NTRU+ KEM864	3888	1440	7488	4752
NTRU+ KEM1152	5184	2016	8352	6336
Our proposal	3584	1920	6784	4608

Table 10. Comparison of CPU cycles for Kengen algorithm between NTRU+KEM and our proposal (cycles)

Scheme	NTT	Mult	Inv	NTT inv
NTRU+ KEM576	19335	6033	39994	20910
NTRU+ KEM768	24314	8129	53899	27803
NTRU+ KEM864	32012	8753	68088	34604
NTRU+ KEM1152	42244	11835	78247	45156
Our proposal	29561	14329	58350	40572

와 Table 10.에서 Keygen 알고리즘에서 실행되는 주요 연산 함수들(NTT 연산, 다항식 곱셈, 다항식 역원, 역NTT 연산)에 대한 분석을 진행하였다.

Table 9.에서는 NTRU+KEM 기법에서 사용되는 주요 연산 함수들의 Montgomery 리덕션 함수 호출량을 분석하였다. 분석 결과, 제안 파라미터는 NTT와 역NTT 연산에서 기존 삼항식 파라미터인 864와 1152보다 적은 리덕션 함수 호출량을 보였다. 이는 이항식 환의 인수분해 과정이 간단하여 함수 호출량이 줄어든 것으로 해석된다. 반면, 다항식 곱셈의 경우 제안 파라미터의 리덕션 함수 호출량은 삼항식 파라미터 864에 비해 적고, 1152에 비해 많았다. 다음으로 다항식의 역원 연산에서 제안 파라미터는 기존 삼항식 파라미터 864와 1152에 비해 적은 호출량을 갖는다. 제안 파라미터는 7차 다항식 역원을 128(=1024/8)번 구하는 데 반해, 864는 2차 다항식의 역원을 288(=863/3)번, 1152는 3차 다항식의 역원을 288(=1152/4)번 구해야 했다는 점에서 제안 파라미터의 함수 호출량의 이점이 발생한 것으로 보인다. 제안 파라미터의 경우 동일 연산의 반복 횟수가 적어 Montgomery 리덕션 함수가 상대적으로 덜 호출되었기 때문으로 해석할 수 있다. 7차 다항식 연산이 비효율적일 것으로 예상되었으나, 4.1절에서 언급된 바와 같이 이항식 환에서 $q=3329$일 때 Montgomery 리덕션 알고리즘의 최대 입력 가능한 곱셈의 수에 7차 다항식 곱셈 시 발생하는 곱셈 수가 포함되어 최적화가 가능했다. 따라서 제안 파라미터는 7차 다항식 연산을 수행해야 하므로 기존 삼항식 파라미터보다 연산이 복잡하지만, 최적화 요인 중 하나인 Montgomery 리덕션 함수 호출량이 상대적으로 작다는 점에서 의미가 있다.

Table 10.은 NTRU+KEM 기법에서 사용되는 주요 연산 함수들의 CPU 사이클을 분석한 결과를 보여준다. Table 10.에 따르면 NTT 연산과 다항식 역원을 구하는 과정에서 제안 이항식 파라미터가 기존 삼항식 파라미터 864보다 낮은 CPU 사이클을 기록했다. 반면 다항식의 곱셈 시에는 삼항식 파라미터 1152보다 높은 CPU 사이클이 측정되었다. 이러한 결과는 Table 9.에서 리덕션 함수 호출량 분석과 유사한 경향을 보여주며, 리덕션 함수가 CPU 사이클에 주요한 영향을 미친다는 것을 확인할 수 있다. 특히 7차 다항식 연산 실행 결과가 비효율적일 것으로 예상되었으나 리덕션 함수 호출량을 줄인 점과 이항식 환의 구조적 특징은 일부 연산에서 효율성을 높이는 데 기여한 것을 실험적 결과를 통해 알 수 있다. 또한 2.5절에서 언급된 다항식을 분할하여 다항식의 역원을 구하는 방법이 고차 다항식의 역원을 구하는 데에 효율적으로 동작함을 확인할 수 있다.

본 절은 이항식 환의 구조적 특징과 리덕션 함수를 활용하여 NTRU+KEM의 알고리즘들을 효율적으로 구현하였음을 보였으며, Table 8.의 CPU 사이클 비교 결과에서 확인할 수 있다. 본 논문에서 통신 효율성을 높이기 위해 다항식 환을 7번만 인수분해 할 수 있는 모듈러스를 선정함에 따라 기존 파라미터들에 비해 연산 과정이 복잡한 7차 다항식의 연산을 구현해야 했다. 그럼에도 리덕션 알고리즘의 최대 입력 가능 범위를 분석하여 7차 다항식 연산에 충분히 대응했다. 이는 Table 9.의 Montgomery 리덕션 함수 호출량에서 뚜렷하게 드러난다. 또한, 가장 연산 과정이 복잡할 것으로 예상되었던 7차 다항식의 역원을 구하기 위해 2.5절의 다항식 분할 방법을 이용하였고, 기존의 삼항식 파라미터에 비해 고차 다항식의 연산을 수행함에도 상대적으로 적은 값의 CPU 사이클이 측정되었다. 이는 Table 10.에서 확인할 수 있었다. 이러한 분석 결과들은 제안 파라미터가 이론적으로 효율적일 뿐만 아니라 실제 구현에서도 성능상의 이점을 제공함을 입증한다.

V. 결 론

본 논문에서는 이항식 환 기반 NTRU+KEM의 새로운 파라미터 $n=1024$, $q=3329$를 제안한다. 3.1절에서 키 및 암호문 크기를 비교한 결과, 제안

된 파라미터는 기존의 파라미터 NTRU+KEM864, NTRU+KEM1152 사이의 적절한 균형을 보였다. 특히 3.2절의 안전성 분석에서 제안 파라미터의 비트 안전성은 클래식한 환경에서 218로 측정되었으며, 기존의 NTRU+KEM864가 달성하지 못한 192비트 안전성을 만족하는 것을 확인할 수 있었다. 이러한 분석 결과는 본 논문에서 제안하는 이항식 파라미터가 NTRU+KEM의 새로운 파라미터 선택지로 충분히 고려될 수 있음을 보여준다.

제안 파라미터 구현 과정에서 NTT를 이항식 환에 적용할 때 발생한 효율성의 한계로 모듈러스 값을 줄이고자 7차 다항식의 연산을 수행해야 했다. 또한 NTRU+KEM의 Keygen 알고리즘을 구현하기 위해 7차 다항식의 역원을 효율적으로 구하는 방법이 요구되었다. 그럼에도 4.1절에서 이항식 환의 다항식 곱셈 시 Montgomery 리덕션 함수의 입력값 범위를 분석해 리덕션 함수 호출을 최소화할 수 있음을 보였고, 4.2절에서는 다항식 차수 감소 방법을 통해 7차 다항식의 역원 계산을 효율적으로 해결하였다. 최종적으로 4.3절에서 NTRU+KEM864와 근소한 차이의 CPU 사이클이 측정되었고, 특히 Keygen 알고리즘에서는 약 4.3% 향상된 CPU 사이클을 가지는 것을 확인할 수 있었다.

본 논문은 NTRU+KEM의 새로운 파라미터를 제안함으로써 안전성을 고려한 파라미터 선택의 자유도를 높이는 데 중점을 두었다. 이항식 파라미터를 구현하고 성능을 분석한 결과, 기존에 제안되었던 NTRU+KEM864에서는 달성할 수 없었던 192비트 안전성을 성공적으로 달성하였다. 이는 기존 파라미터와 비교해 키와 암호문 크기가 증가했음에도 불구하고 효율성 측면에서 큰 손실 없이 근소한 CPU 사이클 차이만을 보였다는 점에서 중요한 의의를 가진다. 따라서, 본 논문에서 제안한 파라미터는 192비트 안전성을 요구하는 환경에 더욱 적합한 대안임을 입증하였다. 이를 통해 NTRU+KEM의 파라미터 선택의 폭을 확장하고 구현 효율성을 향상하는 데 기여할 수 있음을 실증하였다.

References

[1] P. Schwabe, R. Avanzi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J.M. Schanck, G. Seiler, D. Stehle, and J. Ding, "CRYSTALS-KYBER," Technical report, National Institute of Standards and Technology, pp. 1-43, Oct. 2020. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions

[2] S.H. Park, S. Kim, D.H. Lee, and J.H. Park, "Improved ring LWR-based key encapsulation mechanism using cyclotomic trinomials," IEEE Access, vol. 8, pp. 112585-112597, June 2020.

[3] C. Chen, O. Danba, J. Hoffstein, A. Hulsing, J. Rijneveld, J.M. Schanck, P. Schwabe, W. Whyte, Z. Zhang, T. Saito, T. Yamakawa, and K. Xagawa, "NTRU," Technical report, National Institute of Standards and Technology, pp. 1-41, Sep. 2022. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions

[4] V. Lyubashevsky and G. Seiler, "NTTRU: truly fast NTRU using NTT," Cryptology ePrint Archive, vol. 2019, no. 3, pp. 180 - 201, Feb. 2019.

[5] J. Kim and J.H. Park, "NTRU+: Compact construction of NTRU using simple encoding method," IEEE Transactions on Information Forensics and Security, vol. 18, pp. 4760-4774, July 2023.

[6] J. Zhang, D. Feng, and D. Yan, "NEV: Faster and smaller NTRU encryption using vector decoding," International Conference on the Theory and Application of Cryptology and Information Security. Singapore: Springer Nature Singapore, pp. 157-189, Dec. 2023.

[7] P.L. Montgomery, "Modular multiplication without trial division," Mathematics of Computation, vol. 44,

no. 170, pp. 519-521, April 1985.

[8] P. Barrett, "Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor," Conference on the Theory and Application of Cryptographic Techniques. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 311-323, Jan. 1986.

[9] M.R. Albrecht, R. Player, and S. Scott. "On the concrete hardness of learning with errors," Journal of Mathematical Cryptology, no. 9, vol. 3, pp. 169 - 203, Sep. 2015.

[10] M.R. Albrecht, S. Bai, and L. Ducas, "A subfield lattice attack on overstretched NTRU assumptions: : Cryptanalysis of some FHE and graded encoding schemes," Annual International Cryptology Conference. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 153-178, July 2016.

[11] J.H. Cheon, J. Jeong, and C. Lee, "An algorithm for NTRU problems and cryptanalysis of the GGH multilinear map without a low-level encoding of zero," LMS Journal of Computation and Mathematics 19(A), pp. 255-266, Aug. 2016.

[12] L. Ducas, and W.V Woerden, "NTRU fatigue: how stretched is overstretched?," Advances in Cryptology-ASIACRYPT 2021, Singapore, pp. 3-32, Dec, 2021.

〈저자소개〉

이 미 라 (Mira Lee) 학생회원
2024년 2월: 덕성여자대학교 사이버보안전공 졸업
2024년 3월~현재: 고려대학교 정보보호학과 석사과정
〈관심분야〉 암호 프로토콜, 양자 내성 암호

이 승 우 (Seungwoo Lee) 학생회원
2023년 2월: 고려대학교 물리학과/수학과 졸업
2023년 3월~현재: 고려대학교 정보보호학과 석사과정
〈관심분야〉 암호 프로토콜, 양자 내성 암호

김 종 현 (Jonghyun Kim) 정회원
2014년 2월: 성균관대학교 수학과 졸업
2024년 8월: 고려대학교 정보보호학과 석박사 통합과정 졸업
2024년 9월~현재: 고려대학교 정보보호학과 박사후 연구원
〈관심분야〉 암호 프로토콜, 양자 내성 암호

박 종 환 (Jong Hwan Park) 정회원
1999년 2월: 고려대학교 수학과 졸업
2005년 2월: 고려대학교 정보보호학과 석사
2008년 8월: 고려대학교 정보보호학과 박사
2013년 9월~2019년 8월: 상명대학교 컴퓨터과학과 조교수
2019년 9월~2024년 8월: 상명대학교 컴퓨터과학과 부교수
2024년 9월~현재: 상명대학교 컴퓨터과학과 정교수
〈관심분야〉 함수 암호, 브로드캐스트 암호, 암호 프로토콜