Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.28, NO.6, Dec. 2018 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2018.28.6.1329

LWE와 LWR을 이용한 효율적인 다중 비트 암호화 기법*

장 초 롱,[1†] 서 민 혜,[2] 박 종 환[1‡]
[1]상명대학교, [2]고려대학교

Efficient Multi-Bit Encryption Scheme Using LWE and LWR*

Cho Rong Jang,[1†] Minhye Seo,[2] Jong Hwan Park[1‡]
[1]Department of Computer Science, Sangmyung University,
[2]Graduate School of Information Security, Korea University

요 약

최근 양자 컴퓨터에 대한 개발이 활발히 진행되면서, 기존에 널리 사용되고 있는 RSA와 타원곡선 암호 알고리즘의 안전성에 대한 문제가 제기되고 있다. 이에 대응하기 위하여 미국 표준기술연구원(NIST)은 양자 컴퓨팅 환경에서도 안전한 공개키 암호 기법에 대한 표준화 작업을 진행하고 있다. 대표적인 포스트 양자 암호(post-quantum cryptography, PQC) 기법으로는 격자기반 암호(lattice-based cryptography)가 있으며, NIST의 PQC 표준화 공모에도 다양한 격자기반 암호 기법들이 제안되었다. 이 중 EMBLEM은 기존의 LWE (learning with errors) 가정을 기반으로 하여 설계된 암호 기법들과는 달리, 더 직관적이고 효율적으로 암/복호화가 가능한 새로운 다중 비트 암호화 방법을 제안하였다. 본 논문에서는 LWR(learning with rounding) 가정을 추가적으로 사용하여 더 효율적으로 동작하는 다중 비트 암호화 기법을 제안한다. 그리고 제안하는 기법의 안전성을 증명하고, EMBLEM 및 R.EMBLEM과의 비교를 통해 효율성을 분석한다.

ABSTRACT

Recent advances in quantum computer development have raised the issue of the security of RSA and elliptic curve cryptography, which are widely used. In response, the National Institute of Standards and Technology(NIST) is working on the standardization of public key cryptosystem which is secure in the quantum computing environment. Lattice-based cryptography is a typical post-quantum cryptography(PQC), and various lattice-based cryptographic schemes have been proposed for NIST's PQC standardization contest. Among them, EMBLEM proposed a new multi-bit encryption method which is more intuitive and efficient for encryption and decryption phases than the existing LWE-based encryption schemes. In this paper, we propose a multi-bit encryption scheme with improved efficiency using LWR assumption. In addition, we prove the security of our schemes and analyze the efficiency by comparing with EMBLEM and R.EMBLEM.

Keywords: Post-quantum cryptography, Lattice-based cryptography, Learning with errors, Learning with rounding

I. 서 론

1994년 Shor는 양자 컴퓨팅 환경에서 인수분해(factorization) 문제와 이산대수(discrete logarithm) 문제를 효율적으로 해결할 수 있는 알고리즘을 제안하였다[1]. 하지만 그 당시에는 양자 컴퓨팅

Received(09. 03. 2018), Modified(10. 12. 2018)
Accepted(10. 13. 2018)
* 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.
† 주저자, sweatpotato13@gmail.com
‡ 교신저자, jhpark@smu.ac.kr(Corresponding author)

환경의 실현 가능성이 매우 낮았기 때문에, RSA나 ECC와 같이 양자 컴퓨팅 환경에서 안전성을 제공하지 못하는 공개키 알고리즘들이 현재까지 표준으로 사용되어 왔다. 하지만 최근 양자 컴퓨터의 개발이 빠르게 진행되면서 양자 컴퓨팅 환경의 실현 가능성이 높아지고 있으며, 그에 따라 RSA나 ECC와 같은 공개키 암호 알고리즘을 대체할 수 있는 양자 컴퓨팅 환경에서도 안전한 포스트 양자 암호(post-quan tum cryptography, PQC)에 대한 연구가 활발히 진행되고 있다.

격자기반 암호(lattice-based cryptography)는 포스트 양자 암호 중 하나로, 오랫동안 연구가 진행되었기 때문에 안전성이 충분히 검증되었고, 아이디 기반 암호(identity-based encryption), 동형 암호(homomorphic encryption), 함수 암호(functional encryption) 등 다양한 차세대 암호 기법을 설계하는데 사용될 수 있다는 장점이 있다[2, 3, 4]. 또한 구현 속도 측면에서 기존에 사용하던 공개키 암호 알고리즘과 비교할 수 있을 만큼 효율적이기 때문에, 현재 암호 기법 설계와 안전성 분석, 구현 방법에 대한 다양한 연구가 진행되고 있다.

격자기반 암호 기법은 주로 learning with errors(LWE) 가정을 기반으로 하여 설계된다. LWE 문제는 2005년 Regev에 의해 처음 소개되었으며[5], LWE 문제를 푸는 것은 NP-hard에 속하는 다양한 worst-case 격자 문제들을 푸는 것만큼 어렵다는 것이 증명되었다. 이후 다양한 종류의 암호 프리미티브들이 LWE 가정을 기반으로 설계되어 왔으며, 변형된 형태의 LWE 문제들도 정의되고 분석되어 왔다[6, 7]. LWE 문제는 오류(error)를 더해주는 방식으로 정보를 숨기는데, 이로 인해 LWE 가정을 기반으로 설계된 암호 기법들은 복호화 과정에서 노이즈(noise)가 추가된 결과를 얻게 된다. 따라서 이를 해결하기 위해 메시지를 특정한 방법으로 인코딩을 해야 하며, 대부분의 LWE 기반 암호 기법들은 Regev가 제안한 인코딩 방식을 사용한다.

LWE 문제의 변형된 형태 중 하나인 learning with rounding(LWR) 문제는 오류를 더하는 대신 라운딩 연산을 수행함으로써 정보를 숨기는 방식이다. LWR 문제는 2012년 Banerjee 등에 의해 처음 소개되었으며[8], 이후 LWE 문제와 동일한 안전성을 보장하면서 효율적으로 파라미터를 설정할 수 있는 방법에 대한 연구들이 진행되었다[9, 10].

최근 미국 표준기술연구원(NIST)에서 포스트 양자 암호에 대한 표준화 공모를 진행하였으며[11], 1라운드를 통과한 82개의 기법 중 28개(제안된 기법 중 약 34%)가 격자기반 암호/서명 기법이었다. 이 중 EMBLEM은 LWE 가정을 이용한 암호화 알고리즘으로, 새롭게 개발한 메시지 인코딩 방법을 적용하여 (Regev가 제안한 방법에 비해) 직관적이고 더 효율적으로 다중 비트를 암호화하는 기법이다[12].

본 논문에서는 EMBLEM 및 R.EMBLEM 기법에 LWR 가정을 추가적으로 사용하여 효율성을 향상시킨 기법을 제안한다. LWR 문제에서는 이산 가우시안 분포(discrete Gaussian distribution)로부터 오류를 선택하지 않고 (라운딩을 위해) 비트 쉬프트(shift)를 통해 하위 비트들을 버리는 연산만 수행하기 때문에, 연산량 측면에서 효율적이다. 또한 LWE 문제에 비해 LWR 문제의 샘플 크기가 작기 때문에, 전송량 측면에서도 효율성이 증가한다. 본 논문에서 제안하는 기법은 EMBLEM 및 R.EMBLEM과 비교하여 암호문의 크기가 줄어들었으며, 암/복호화 단계의 수행 시간이 감소하였다.

본 논문의 구성은 다음과 같다. 제2장에서는 기법 설계 및 안전성 증명을 위한 배경지식을 설명한다. 제3장에서는 LWE와 LWR 가정을 기반으로 하는 암호 기법을 제안하고, 제안한 기법의 안전성을 증명한다. 제4장에서는 링 구조에서 기법을 설계하고, RLWE(ring LWE)와 RLWR(ring LWR) 가정을 기반으로 하여 안전성을 증명한다. 제5장에서는 제안한 기법과 EMBLEM의 구현 결과를 비교하고, 그 효율성을 분석한다. 마지막으로 제6장에서 결론을 맺는다.

II. 배경지식

본 장에서는 공개키 암호 기법 및 KEM 기법의 알고리즘 및 안전성 모델을 정의하고, 제안하는 기법이 기반으로 하는 암호학적 가정들을 설명한다.

2.1 공개키 암호

2.1.1 알고리즘

공개키 암호는 세 개의 알고리즘 ($KeyGen, Enc, Dec$)로 구성 되어 있다.

- **키생성** $KeyGen(1^{\lambda})$: 보안 상수(security parameter) 1^{λ}을 입력으로 받아 암호화에 사용

되는 공개키(pk)와 복호화에 사용되는 비밀키(sk)를 출력한다.

- **암호화** $Enc(pk,m)$: 공개키 pk와 메시지 m을 입력으로 받아 암호문 c를 출력한다.
- **복호화** $Dec(sk,c)$: 비밀키 sk와 암호문 c를 입력으로 받아 메시지 m을 출력하거나 실패를 나타내는 $\perp$를 출력한다.

정확성(correctness). 모든 메시지 M에 대하여

$$\Pr[(pk,sk)\leftarrow_R Keygen(1^\lambda);\ C\leftarrow_R Encrypt(pk,M):\ Decrypt(sk,C)=M] > 1-\epsilon(\lambda)$$

이 성립한다. 여기서 ϵ은 무시할 수 있는 함수이다.

2.1.2 안전성 모델

공개키 암호 기법의 안전성 모델은 공격자 A와 챌린저 S가 수행하는 다음의 실험 $\mathrm{Exp}_{A,\Pi}^{CPA}(1^\lambda)$으로 정의된다.

$\Pi=(KeyGen,Enc,Dec)$를 공개키 암호 기법이라 하자. 공격자 A는 보안상수 1^λ에 대해 챌린저 S와 다음과 같이 가상의 게임(game)을 수행한다.

- **Setup**: S는 $KeyGen(1^\lambda)$을 수행하여 공개키 pk와 비밀키 sk를 생성하고, A에게 공개키 pk를 전송한다.
- **Challenge**: A는 동일한 길이의 메시지 쌍 (m_0,m_1)을 S에게 전송한다. S는 임의의 비트 $b\in\{0,1\}$을 선택하고, $c^*\leftarrow Enc(pk,m_b)$를 생성하여 A에게 전송한다.
- **Guess**: A는 $b'\in\{0,1\}$을 출력한다. 만약 $b=b'$인 경우 S는 1을 출력하고, 그렇지 않으면 0을 출력한다.

공격자 A가 위의 실험 $\mathrm{Exp}_{A,\Pi}^{CPA}(1^\lambda)$에서 얻는 이점(advantage)은 다음과 같이 정의된다.

$$Adv_{\Pi}^{CPA}(A)=\left|\Pr\left[\mathrm{Exp}_{A,\Pi}^{CPA}(1^\lambda)=1\right]-\frac{1}{2}\right|$$

정의 1. 공개키 암호 기법에 대한 임의의 다항식 시간 공격자 A에 대하여 공격자의 이점 $Adv_{\Pi}^{CPA}(A)$이 의미 없는(negligible) 값이라면, 공개키 암호 기법 $\Pi=(KeyGen,Enc,Dec)$은 선택 평문 공격에 안전(indistinguishability under chosen plaintext attack, IND-CPA)하다.

2.2 KEM(Key Encapsulation Mechanism)

2.2.1 알고리즘

KEM은 세 개의 알고리즘 $(Setup,Encap,Decap)$로 구성 되어 있다.

- **설정** $Setup(1^\lambda)$: 보안 상수(security parameter) 1^λ을 입력으로 받아 캡슐화에 사용되는 공개키(pk)와 디캡슐화에 사용되는 비밀키(sk)를 출력한다.
- **캡슐화** $Encap(pk)$: 공개키 pk를 입력으로 받아 암호문 c와 키 K를 출력한다.
- **디캡슐화** $Decap(sk,c)$: 비밀키 sk와 암호문 c를 입력으로 받아 키 K를 출력하거나 실패를 나타내는 $\perp$를 출력한다.

정확성(correctness). KEM은

$$\Pr[(pk,sk)\leftarrow_R Setup(1^\lambda);\ (C,K)\leftarrow_R Encap(pk):\ Decap(sk,C)=K] > 1-\epsilon(\lambda)$$

이 성립한다. 여기서 ϵ은 무시할 수 있는 함수이다.

2.2.2 안전성 모델

KEM 기법이 선택 암호문 공격(chosen cipher text attacks, CCA)에 대해 안전하다는 것은 임의의 공격자가 자신이 선택한 암호문에 대한 키를 (디캡슐화 오라클을 통해) 획득할 수 있더라도 챌린지 암호문에 대한 키 정보를 획득할 수 없어야 함을 의미한다. KEM 기법의 안전성 모델은 공격자 A와 챌린저 S가 수행하는 다음의 실험 $\mathrm{Exp}_{A,\Pi}^{CCA}(1^\lambda)$으로 정의된다.

$\Pi=(Setup,Encap,Decap)$을 KEM 기법이라 하자. 공격자 A는 보안상수 1^λ에 대해 챌린저 S와 다음과 같이 가상의 게임을 수행한다.

- **Setup**: S는 $KeyGen(1^\lambda)$을 수행하여 공개키 pk와 비밀키 sk를 생성하고, A에게 공개키 pk를 전송한다.
- **Query1**: A는 S에게 임의의 암호문 c를 질의

하고, S는 메시지 $K \leftarrow Decap(sk, c)$을 생성하여 A에게 전송한다.

- **Challenge**: S는 랜덤한 키 K_0^*를 선택하고 $(c^*, K_1^*) \leftarrow Encap(pk)$를 생성한다. S는 임의의 비트 $b \in \{0,1\}$을 선택하고, (c^*, K_b^*)를 A에게 전송한다.
- **Query2**: A는 S에게 암호문 $c(\neq c^*)$를 질의하고, S는 메시지 $K \leftarrow Decap(sk, c)$을 생성하여 A에게 전송한다.
- **Guess**: A는 $b' \in \{0,1\}$을 출력한다. 만약 $b = b'$인 경우 S는 1을 출력하고, 그렇지 않으면 0을 출력한다.

공격자 A가 위의 실험 $\mathrm{Exp}_{A,\Pi}^{CCA}(1^\lambda)$에서 얻는 이점(advantage)은 다음과 같이 정의된다.

$$Adv_{\Pi}^{CCA}(A) = \left| \Pr\left[\mathrm{Exp}_{A,\Pi}^{CCA}(1^\lambda) = 1\right] - \frac{1}{2} \right|$$

정의 2. KEM 기법에 대한 임의의 다항식 시간 공격자 A에 대하여 공격자의 이점 $Adv_{\Pi}^{CCA}(A)$이 의미 없는(negligible) 값이라면, 공개키 암호 기법 $\Pi = (Setup, Encap, Decap)$은 선택 암호문 공격에 안전(indistinguishability under chosen ciphertext attack, IND-CCA)하다.

2.3 암호학적 가정

2.3.1 Learning with errors (LWE)

정의 3 (LWE 샘플). 주어진 비밀 벡터 $s \in Z_q^n$에 대하여, 랜덤한 $a \in Z_q^n$와 분포 D_e에서 선택한 오류 $e \in D_e$를 이용하여 출력한 $(a, b = \langle a, s \rangle + e \pmod q)$를 LWE 샘플이라고 한다.

정의 4 (LWE 문제). m개의 샘플 $(A, B) \in Z_q^{m \times n} \times Z_q^{m \times k}$가 주어졌을 때, 이것이 LWE 샘플인지, 즉, $(A, B = AS + E \pmod q)$를 만족하는 $S \in Z_q^{n \times k}$가 존재하는지, 아니면 랜덤하게 선택된 $B \in Z_q^{m \times k}$인지를 구분하는 문제를 결정 LWE 문제(decisional LWE problem)라고 한다[5].

[5] 에서는 벡터 형식의 LWE 문제를 제시 하였으나 본 논문에서는 행렬 형태의 LWE 문제를 고려한다. [13]에서는 행렬 형태의 LWE 문제가 벡터 형식의 LWE 문제 보다 쉽지 않다는 것을 보였다.

링(ring) 구조에서도 LWE 문제를 정의할 수 있다[14]. 링 구조에서는 벡터가 다항식(polynomial)으로 표현되며, 여기서는 차수가 n인 기약 다항식 $f(x) \in Z[x]$에 대해 $R_q = Z_q[x]/f(x)$를 모듈로 q에서의 링으로 사용한다.

정의 5 (RLWE 문제). 샘플 $(a,b) \in R_q^2$가 주어졌을 때, 분포 D_e^n에서 선택된 값들을 계수로 가지는 다항식 e에 대하여, $b = a \cdot s + e$를 만족하는 비밀 다항식 $s \in R_q$가 존재하는지, 아니면 랜덤하게 선택된 $b \in R_q$인지를 구분하는 문제를 결정 RLWE 문제(decisional Ring LWE problem)라고 한다.

본 논문에서는 결정 (R)LWE 문제를 (R)LWE 문제라고 표현한다. (R)LWE 가정은 (R)LWE 문제를 풀 확률이 무시할 만큼 낮다(negligible)는 것을 의미한다. (R)LWE 문제의 변형으로 비밀 벡터 (혹은 다항식의 계수 벡터) s를 $[-B, B]^n$와 같은 작은 범위에서 선택할 수 있으며, n의 크기를 증가시킴에 따라 Z_q^n에서 선택했을 때와 동일한 안전성을 제공할 수 있다[15][16].

2.3.2 Learning with rounding (LWR)

정의 6 (LWR 샘플). 주어진 비밀 벡터 $s \in Z_q^n$에 대하여, 랜덤한 $a \in Z_q^n$와 모듈러스 p, q를 이용하여 출력한 $(a, b = \lfloor (p/q)(\langle a, s \rangle \bmod q) \rceil)$를 LWR 샘플이라고 한다.

정의 7 (LWR 문제). m개의 샘플 $(A, B) \in Z_q^{m \times n} \times Z_p^{m \times k}$가 주어졌을 때, 이것이 LWR 샘플인지, 즉, $(A, B = \lfloor (p/q)(A \cdot S \bmod q) \rceil)$를 만족하는 $S \in Z_q^{n \times k}$가 존재하는지, 아니면 랜덤하게 선택된 $B \in Z_p^{m \times k}$인지를 구분하는 문제를 결정 LWR 문제(decisional LWR problem)라고 한다[8].

결정 LWR 문제도 결정 LWE 문제와 동일하게 링 구조에서 정의할 수 있으며, 이를 결정 RLWR 문제(decisional Ring LWR problem)라고 한다

[8]. 또한 본 논문에서는 결정 (R)LWR 문제를 (R)LWR 문제라고 표현한다. (R)LWR 가정은 (R)LWR 문제를 풀 확률이 무시할 만큼 낮다는 것을 의미한다.

III. LWE와 LWR을 이용한 다중 비트 암호화 기법

3.1 메시지 인코딩

LWE와 LWR을 이용하여 암호 기법을 설계하기 위해서는 메시지 인코딩을 통해 비트 문자열을 행렬(matrix) 형태로 변형해야 한다. 메시지 인코딩을 함으로써 복호화 과정에서 라운딩 없이 최상위 비트를 떼어내는 것으로 간단하게 메시지를 획득 할 수 있다.

256-bit의 메시지가 t-bit 크기의 $M_{(i,j)}$로 이루어진 $v\times k$ 행렬 $\mathbf{M}=\{M_{(i,j)}\}$ (여기서 $t\times v\times k=256$)로 표현된다고 가정할 때 오차 $|R^TE-f_2+f_1X|<2^d$ 를 무시할 수 있는 확률로 만드는 매개 변수 d를 예측할 수 있고 인코딩된 메시지의 길이는 $\lfloor \log_2(q) \rfloor = t+1+d$ 로 결정된다.

l-bit 길이의 메시지 msg, 불록 사이즈 t ,에러의 상한을 결정하는 d, $l/t=v\times k$를 만족하는 양수 v,k 그리고 모듈러스(modulus) q에 대하여, 인코딩 함수 **encode**와 디코딩 함수 **decode**는 다음과 같이 동작한다. 디코딩 함수 **decode**는 인코딩 함수 **encode**의 정 반대의 과정을 수행한다.

encode(msg, t, d, v, k, q):
① 메시지 msg을 t-bit 단위로 나눈다. (여기서 t는 l을 나눌 수 있다고 가정한다.) 그리고 $i'\in[1,l/t]$인 메시지 블록 $\{M_{i'}\}$을 생성한다.
② $i'\in[1,l/t]$에 대해서 $i=\lceil i'/k\rceil\in[1,v]$ 이고 $j=i'-\lfloor i'/k\rfloor\cdot k\in[1,k]$ 인 $M_{(i,j)}=M_{i'}$로 변환한다.
③ $i=[1,v], j=[1,k]$ 에 대해서 $\mathrm{M}[i,j]\leftarrow M_{(i,j)}\|1\|0^d$인 $v\times k$ 행렬 $\mathrm{M}=\{\mathrm{M}[i,j]\}$ 을 출력한다.

decode(M, t, q):
① $i=[1,v], j=[1,k]$ 에 대해서 $i'=(i-1)k+j\in[1,l/t]$인 $M_{i'}=[\mathrm{M[i,j]}]_t$을 계산한다.
② l-bit 문자열 $m=M_1\|\cdots\|M_{l/t}$을 출력한다.

3.2 선택 평문 공격(IND-CPA)에 안전한 기법

먼저 기법에서 사용되는 샘플링 함수 **Sam**을 정의한다. 샘플링 함수 **Sam**은 임의의 256-bit 시드(seed) r을 입력으로 받아 행렬 $R\in[-\beta,\beta]^{m\times v}$을 출력한다. 여기서 R은 랜덤하게 생성되며, 동일한 입력 r에 대해서는 항상 동일한 값 R이 출력된다. 따라서 의사 난수 함수(pseudorandom function)을 사용하여 함수 **Sam**을 설계할 수 있다.

시스템 파라미터로 사용되는 값들은 다음과 같이 생성된다. 먼저 양의 정수 m,n,k,t,v,d와 모듈러스 q와 p를 선택한다. 그리고 이산 가우시안 분포(discrete Gaussian distribution) GD_s의 표준 편차 $\sigma=s/\sqrt{2\pi}$ 와 양의 정수 $\beta<\sigma$ 를 선택한다. 시스템 파라미터 $params=(m,n,k,t,v,d,q,p,\beta,GD_s)$가 주어졌을 때, 선택 평문 공격에 안전한 공개키 암호 알고리즘 (**KeyGen**, **Encrypt**, **Decrypt**)은 다음과 같이 정의된다.

KeyGen(1^λ):
① 임의의 행렬 $A\in Z_q^{m\times n}$와 $X\in[-\beta,\beta]^{n\times k}$를 랜덤하게 선택한다.
② 오류 행렬 $E\in GD_s^{m\times k}$를 랜덤하게 선택한다.
③ 행렬 $B=AX+E\pmod q$ 를 계산한다. 공개키 pk와 비밀키 sk는 다음과 같다:
$$pk=(A,B),\ sk=(X).$$

Encrypt(pk, msg):
① 행렬 $\mathrm{M}\leftarrow$ **encode**(msg, t, d, v, k, q)을 생성한다.
② 임의의 시드 $r\in\{0,1\}^{256}$을 선택한다.
③ **Sam**(r)을 계산하여 행렬 $R\in[-\beta,\beta]^{m\times v}$을 생성한다.
④ (C_1, C_2)를 다음과 같이 계산한다.
$$\begin{cases}C_1=\lfloor (p/q)R^TA\rceil\\ C_2=\lfloor (p/q)R^TB\rceil+\mathrm{M}\end{cases}$$
⑤ 암호문 $C=(C_1,C_2)$를 출력한다.

Decrypt(sk, C):

① 암호문 $C=(C_1, C_2)$에 대하여, $(C_1', C_2')=((q/p)C_1, (q/p)C_2)$를 계산한다.

② $\mathrm{M}= C_2' - C_1' X$을 계산한다.

③ $msg \leftarrow$ **decode**(M, t, q)을 출력한다.

정리 1 (Correctness).

$R\in[-\beta,\beta]^m$, $E\in GD_s^n$, $X\in[-\beta,\beta]^n$일 때 $i\in[1,k]$에 대하여 아래와 같이 정의한다.

- $\mathbf{E}=\{E^{(i)}\}$에서 $E^{(i)}\in GD_s^m$은 $\mathbf{E}$의 i번째 열
- $\mathbf{X}=\{X^{(i)}\}$에서 $X^{(i)}\in[-\beta,\beta]^n$은 $\mathbf{X}$의 i번째 열

$d= \lfloor \log_2(q) \rfloor -(t+1)$ 에 대하여 ϵ을 다음과 같이 정의하자.

$$\epsilon=\Pr\left[\max_{i\in[1,k]}\left\{\left|\langle R,E^{(i)}\rangle\right|+|f_2| +\left|\langle f_1,X^{(i)}\rangle\right|\right\}\geq 2^d\right]$$

이 때, 3.2의 기법의 정확성은 $(1-\epsilon)$이다.

증명. 복호화 과정은 다음과 같이 진행된다.

$$\begin{aligned}\mathrm{M} &= C_2' - C_1' X\\ &=(q/p)C_2-(q/p)C_1X\\ &=(R^TB+M-f_2)-(R^TA-f_1)X\\ &=(R^T(AX+E)+M-f_2)-(R^TAX-f_1X)\\ &=(R^TE+M-f_2)+f_1X\\ &=M+(R^TE-f_2+f_1X)\end{aligned}$$

$f_1\leftarrow C_1' \pmod{q/p}$, $f_2\leftarrow C_2' \pmod{q/p}$ 이다. 만약 $q=2^{24}, p=2^{22}$인 경우, f_i는 $\lfloor (p/q)*C_i \rceil$ 의 연산을 수행하여 -3에서 3 사이의 값을 가질 것이라고 예상할 수 있다. 하지만 실제로 동작하는 과정에서는 이보다 좁은 범위의 값으로 생성된다. 여기서 f_i는 실제로 $[-2^{k-1}+1, 2^{k-1}]$의 값이 균일하게 발생된다.

만약 $|R^TE-f_2+f_1X|$ 이 2^d보다 작다면, (복호화 과정에서 발생하는 전체 오류의 부호에 상관없이) 실제 메시지 블록에는 영향을 주지 못한다. 인코딩을 통해 메시지 블록 m은 $m\|1\|0^d$ 형태로 변형되는데, 이 때 중간에 위치한 비트 1이 오류가 확산되는 것을 막아주기 때문이다. □

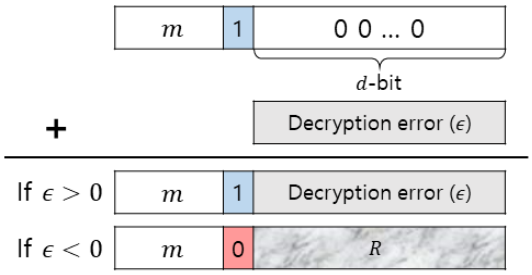

Fig. 1. Error propagation in the decryption phase

3.2.1 안전성 증명

정리 2 (Security). 만약 LWE와 LWR 가정이 성립한다면, 3.2절에서 소개한 공개키 암호 기법은 선택 평문 공격에 안전(IND-CPA secure)하다.

증명. 우리는 다음의 과정을 통해 $|m_0|=|m_1|$ 인 두 메시지 m_0, m_1에 대하여, m_0에 대한 암호문과 m_1에 대한 암호문이 구분 불가능함을 증명한다.

Game 0. 이 게임에서는 기법과 동일한 방법으로 공개키와 m_0에 대한 암호문이 생성된다. 여기서 $M_0\leftarrow$ **encode**(m_0, t, d, v, k, q)을 의미한다. **Game 0**의 분포 D_0는 아래와 같다.

$$D_0:\begin{cases}pk\leftarrow(A, B=AX+E),\\ C_1=\lfloor (p/q)(R^TA)\rceil,\\ C_2=\lfloor (p/q)(R^TB)+M_0\rceil\end{cases}$$

Game 1. 이 게임에서는 공개키 B를 $Z_q^{m\times k}$에서 랜덤하게 선택하고, 그 외에는 **Game 0**에서와 동일하게 진행된다. **Game 1**의 분포 D_1은 아래와 같다.

$$D_1:\begin{cases}pk\leftarrow(A, B\leftarrow U(Z_q^{m\times k})),\\ C_1=\lfloor (p/q)(R^TA)\rceil,\\ C_2=\lfloor (p/q)(R^TB)+M_0\rceil\end{cases}$$

D_0와 D_1은 공개키 B를 생성하는 방법에서 차이가 있다. D_0에서는 공개키 B가 LWE 구조이며 D_1에서는 랜덤한 값이기 때문에, LWE 가정이 성립한다면 D_0와 D_1은 구분 불가능하다.

Game 2. 이 게임에서는 암호문 (C_1, C_2)가 $Z_q^n \times Z_q^k$에서 랜덤하게 선택된 (U_1, U_2)를 이용하여 생성되고, 그 외에는 **Game 1**에서와 동일하게 진행된다. **Game 2**의 분포 D_2는 아래와 같다.

$$D_2 : \begin{cases} pk \leftarrow (A, B \leftarrow U(Z_q^{n\times k})), \\ C_1 = U_1, C_2 = U_2 + M_0 \end{cases}$$

D_1과 D_2는 암호문 (C_1, C_2)를 생성하는 방법에서 차이가 있다. D_1에서는 암호문 (C_1, C_2)가 LWR 구조이며 D_2에서는 랜덤한 값이기 때문에, LWR 가정이 성립한다면 D_1과 D_2는 구분 불가능하다.

Game 3. 이 게임에서는 M_0가 $M_1 \leftarrow$ **encode** (m_1, t, d, v, k, q)으로 바뀌고, 그 외에는 **Game 2**에서와 동일하게 진행된다. **Game 3**의 분포 D_3는 아래와 같다.

$$D_3 : \begin{cases} pk \leftarrow (A, B \leftarrow U(Z_q^{n\times k})), \\ C_1 = U_1, C_2 = U_2 + M_1 \end{cases}$$

D_2와 D_3는 다른 메시지에 대한 암호문을 생성한다. D_2와 D_3에서는 one-time pad 형태로 메시지가 암호화되기 때문에, D_2와 D_3는 통계적으로(statistically) 구분 불가능하다.

Game 4. 이 게임에서는 암호문 (C_1, C_2)가 LWR 구조로 복원되고, 그 외에는 **Game 3**에서와 동일하게 진행된다. **Game 4**의 분포 D_4는 아래와 같다.

$$D_4 : \begin{cases} pk \leftarrow (A, B \leftarrow U(Z_q^{n\times k})), \\ C_1 = \lfloor (p/q)(R^T A) \rceil, \\ C_2 = \lfloor (p/q)(R^T B) + M_1 \rceil \end{cases}$$

Game 2에서와 같이, LWR 가정이 성립한다면 D_3와 D_4는 구분 불가능하다.

Game 5. 이 게임에서는 공개키 B가 LWE 구조로 복원되고, 그 외에는 **Game 4**에서와 동일하게 진행된다. **Game 5**의 분포 D_5는 아래와 같다.

$$D_5 : \begin{cases} pk \leftarrow (A, B = AX + E), \\ C_1 = \lfloor (p/q)(R^T A) \rceil, \\ C_2 = \lfloor (p/q)(R^T B) + M_1 \rceil \end{cases}$$

Game 1에서와 같이, LWE 가정이 성립한다면 D_4와 D_5는 구분 불가능하다.

결과적으로, LWE와 LWR 가정이 성립한다면 m_0를 암호화하는 **Game 0**와 m_1을 암호화하는 **Game 5**는 구분 불가능하다. 따라서 3.2절에서 소개한 공개키 암호 기법은 IND-CPA에 안전하다.□

3.3 선택 암호문 공격(IND-CCA)에 안전한 기법

본 절에서는 양자 랜덤 오라클 모델(quantum random oracle model, QROM)에서 IND-CCA에 안전한 KEM(key encapsulation mechanism) 기법을 제시한다. 제안하는 기법은 3.2절에서 소개한 IND-CPA에 안전한 공개키 암호 기법에 FO 변형(Fujisaki-Okamoto transformation) 방법을 적용한 것이다[17].

메시지 공간은 $\{0,1\}^{256}$이며, 시스템 파라미터는 3.2절의 공개키 암호 기법과 동일하다. FO 변형 방법을 적용하기 위해 다음과 같이 3가지 해시함수가 추가적으로 사용되며, 이는 안전성 증명 과정에서 랜덤 오라클로 모델링된다. (해시함수 $\hat{H}$은 일반 랜덤 오라클 모델에서 IND-CCA에 안전한 기법을 설계할 때에는 사용되지 않는다.)

- 해시함수 G: $\{0,1\}^* \rightarrow \{0,1\}^{256}$
- 해시함수 H: $\{0,1\}^* \rightarrow \{0,1\}^{256}$
- 해시함수 $\hat{H}$: $\{0,1\}^* \rightarrow \{0,1\}^{256}$

3.2절에서 제시한 공개키 암호 알고리즘 (**KeyGen**, **Encrypt**, **Decrypt**)이 주어졌을 때, 선택 암호문 공격에 안전한 KEM 알고리즘 (**Setup**, **Encap**, **Decap**)은 다음과 같이 정의된다.

Setup(1^λ):
① 3.2절의 **KeyGen**과 동일하다.

Encap(pk):
① 임의의 문자열 $\delta \in \{0,1\}^{256}$를 랜덤하게 선택하고, $r = G(\delta)$ 을 계산한다.
② $C_1 \leftarrow$ **Encrypt**$(pk, \delta; r)$과 $C_2 = \hat{H}(\delta)$ 를 계산한다.
③ $K = H(\delta, C_1, C_2)$를 계산한다.
④ 암호문 $C = (C_1, C_2) \in Z_q^{v\times(n+k)} \times \{0,1\}^{256}$ 와 키 $K \in \{0,1\}^{256}$를 출력한다.

Decap(sk, C):
① 암호문 $C = (C_1, C_2)$에 대하여, $\delta \leftarrow$ **Decrypt** (sk, C_1)를 계산한다.
② $r = G(\delta)$ 를 계산한다.
③ $e \leftarrow$ **Encrypt**$(pk, \delta; r)$, $d = \hat{H}(\delta)$ 를 계산한다.

• 만약 $e \neq C_1$ 또는 $d \neq C_2$이면, $\perp$를 출력한다.
④ 그렇지 않다면, $K = H(\delta, C_1, C_2)$를 출력한다.

3.3.1 안전성 증명

3.3절에서 제안한 기법은 **(정리 3)** 랜덤 오라클 모델(ROM)에서는 타이트(tight)하게 IND-CCA에 안전하며, **(정리 4)** 양자 랜덤 오라클 모델(QROM)에서는 타이트하지 않게 IND-CCA에 안전하다.

정리 3 (Theorem 3.1 and 3.2 in [17]). 3.2절에서 제안된 기법이 δ-정확성을 가진다고 가정하자. q_D번의 복호화 질의, q_G번의 랜덤 오라클 G 질의, q_H번의 랜덤 오라클 H 질의를 수행하는 임의의 IND-CCA 공격자 B에 대하여, 아래의 이점을 가지는 IND-CPA 공격자 A가 존재한다.

$$Adv^{CCA}(B) \leq q_H{}^{*}\delta + \frac{q_H + 2q_G + 1}{2^{256}} + 3^{*}Adv^{CPA}(A)$$

정리 4 (Theorem 4.4 and 4.6 in [17]). 3.2절에서 제안된 기법이 δ-정확성을 가진다고 가정하자. q_D번의 복호화 질의, q_G번의 양자 랜덤 오라클 G 질의, q_H번의 양자 랜덤 오라클 H 질의, $q_{\hat{H}}$번의 양자 랜덤 오라클 $\hat{H}$ 질의를 수행하는 임의의 IND-CCA 공격자 B에 대하여, 아래의 이점을 가지는 IND-CPA 공격자 A가 존재한다.

$$Adv^{CCA}(B) \leq \left(2q_H + q_{\hat{H}}\right) \cdot \sqrt{8^{*}\delta(q_G+1)^2 + (1+2q_G)\sqrt{Adv^{CPA}(A)}}$$

IV. RLWE와 RLWR을 이용한 다중 비트 암호화 기법

이 장에서는 링 구조에서 설계한 Ring-LWE (RLWE) 및 Ring-LWR (RLWR) 기반의 공개키 암호 기법을 소개한다.

4.1 메시지 인코딩

RLWE와 RLWR을 이용하여 암호 기법을 설계하기 위해서는 메시지 인코딩을 통해 비트 문자열을 다항식(polynomial) 형태로 변형해야 한다. l-bit 길이의 메시지 m, 블록 사이즈 t, 에러의 상한을 결정하는 d, 그리고 모듈러스(modulus) q에 대하여, 인코딩 함수 **R.encode**는 다음과 같이 동작한다.

R.encode(m, t, q, d):
① 메시지 m을 t-bit 단위로 나눈다. (여기서 t는 l을 나눌 수 있다고 가정한다.) 그리고 $i \in [1, l/t]$에 대해 메시지 블록 $\{M_i\}$을 생성한다.
② $i \in [1, l/t]$인 i에 대하여, 메시지 블록 $\{M_i\}$를 $\lfloor \log_2(q) \rfloor$ bit 길이의 비트 문자열 $\widehat{m_i} = M_i \| 1 \| 0^d$로 변환한다.
③ 계수가 $(\widehat{m_1}, \cdots, \widehat{m_{l/t}}, 0, \ldots, 0)$인 $(n-1)$차 다항식 $\hat{m}$을 출력한다.

디코딩 함수 **R.decode**는 **R.encode**의 반대 순서로 동작한다. 다항식 $\hat{m}$, 블록 사이즈 t, 그리고 모듈러스 q에 대하여, **R.decode**는 다음과 같이 동작한다.

R.decode$(\hat{m}, t, q)$:
① $i \in [1, l/t]$인 i에 대하여, $M_i = [\widehat{m_i}]_t$를 계산한다. 여기서 $\widehat{m_i}$는 다항식 $\hat{m}$의 i번째 계수이다.
② l-bit 문자열 $M = M_1 \| \cdots \| M_{l/t}$를 출력한다.

4.2 선택 평문 공격(IND-CPA)에 안전한 기법

원분다항식(cyclotomic polynomial) $f(x) = x^n + 1 \in Z[x]$에 대하여, $R_q = Z_q[x]/\langle f(x) \rangle$는 모듈로 $f(x)$와 모듈로 q에서의 정수 다항식으로 구성된 링(ring)이다. R_q의 원소들은 계수가 $\{0, \ldots, q-1\}$에서 선택되는 n차 미만의 다항식으로 표현된다. 3장에서 소개한 기법과 마찬가지로, 본 절 장에서도 링 구조에서 사용되는 샘플링 함수 **R.Sam**을 정의한다. **R.Sam**은 임의의 256-bit 시드(seed) z를 입력으로 받아 $[-B, B]$의 값들을 계수로 가지는 다항식 $r \in R_q$을 출력한다.

전송 효율성을 높이기 위해서 **Trunc** 함수가 추가적으로 사용된다. 임의의 n차 다항식 $a = a_0 + a_1 X + \cdots + a_{n-1} X^{n-1} \in R_q$와 $1 \leq l \leq n$에 대하여, **Trunc** 함수는 다음과 같이 정의된다.

$$\mathbf{Trunc}(a,l)=a_0+a_1X+\cdots+a_{l-1}X^{l-1}$$

시스템 파라미터는 다음과 같이 생성된다. 먼저 양의 정수 n,t와 모듈러스 q와 p를 선택한다. 그리고 이산 가우시안 분포 GD_s의 표준 편차 $\sigma=s/\sqrt{2\pi}$ 와 양의 정수 $B<\sigma$ 를 선택한다. 시스템 파라미터 $params=(n,t,q,p,B,R_q,GD_s)$가 주어졌을 때, 선택 평문 공격에 안전한 링 구조의 공개키 암호 알고리즘 (R.Keygen, R.Encrypt, R.Decrypt)는 다음과 같이 정의된다.

R.KeyGen(1^λ):

① 임의의 다항식 $a\in R_q$를 랜덤하게 선택한다.

② $[-B,B]$에서 계수들을 랜덤하게 선택하여 다항식 $x\in R_q$를 생성한다.

③ GD_s에서 계수들을 랜덤하게 선택하여 다항식 $e\in R_q$를 생성한다.

④ 다항식 $b=ax+e\in R_q$를 계산한다. 공개키 pk와 비밀키 sk는 다음과 같다:

$$pk=(a,b),\ sk=(x).$$

R.Encrypt(pk,m):

① 다항식 $\widehat{m}\leftarrow$R.encode(m,t,q,d)을 생성한다.

② 임의의 시드 $z\in\{0,1\}^{256}$를 선택한다.

③ R.Sam(z)을 계산하여 $[-B,B]$의 값들을 계수로 가지는 다항식 r을 생성한다.

④ $c_1=r*a$, $c_2\leftarrow$Trunc$(r*b,l/t)$을 계산한다.

⑤ $(c_1',c_2')=\left(\lfloor(p/q)*c_1\rceil,\ \lfloor(p/q)*c_2\rceil+\widehat{m}\right)$을 계산한다.

⑥ 암호문 $c=(c_1',c_2')$를 출력한다.

R.Decrypt(sk,c):

① 암호문 $c=(c_1',c_2')$에 대하여, $(c_1'',c_2'')=(\lfloor(q/p)*c_1'\rceil,\lfloor(q/p)*c_2'\rceil)$를 계산한다.

② $d\leftarrow$Trunc$(c_1''*x,\ l/t)$를 계산한다.

③ $\widehat{m}=c_2''-d$ 를 계산한다.

④ $m\leftarrow$ R.decode$(\widehat{m},t,q)$를 출력한다.

정리 5 (Correctness). $r^{(i)}$, $e^{(i)}$, 그리고 $x^{(i)}$는 각각 다항식 r, e, 그리고 x의 i번째 계수이다. $d=\log_2(q)-(t+1)$ 에 대하여, ϵ을 다음과 같이 정의하자.

$$\epsilon=\Pr\left[\max_{i\in[1,l/t]}\left\{\left|r^{(i)}*e^{(i)}\right|+|f_2|+\left|f_1*x^{(i)}\right|\right\}\geq 2^d\right]$$

이 때, 4.2의 기법의 정확성은 $(1-\epsilon)$이다.

증명. 복호화 과정은 다음과 같이 진행된다.

$$\begin{aligned}&c_2''-d\\&=\mathbf{Trunc}(r*b-f_2,l/t)+\widehat{m}\\&\qquad-\mathbf{Trunc}((r*a-f_1)*x,l/t)\\&=\mathbf{Trunc}(r*(ax+e)-f_2-(r*ax-f_1x),l/t)+\widehat{m}\\&=\mathbf{Trunc}(re-f_2+f_1x,l/t)+\widehat{m}\end{aligned}$$

위의 식에서 $f_1=c_1(\mathrm{mod}\ q/p)$ 이고 $f_2=c_2(\mathrm{mod}\ q/p)$이다. 만약 $|re-f_2+f_1x|$ 이 2^d보다 작다면, 메시지 인코딩으로 인해 복호화 과정에서 발생하는 오류는 (부호에 상관없이) 실제 메시지 블록에는 영향을 주지 못한다. □

4.2.1 안전성 증명

정리 6 (Security). 만약 RLWE와 RLWR 가정이 성립한다면, 4.2절에서 소개한 링 구조의 공개키 암호 기법은 선택 평문 공격에 안전(IND-CPA secure)하다.

증명. 우리는 다음의 과정을 통해 $|m_0|=|m_1|$ 인 두 메시지 m_0,m_1에 대하여, m_0에 대한 암호문과 m_1에 대한 암호문이 구분 불가능함을 증명한다.

Game 0. 이 게임에서는 기법과 동일한 방법으로 공개키와 m_0에 대한 암호문이 생성된다. 여기서 $\widehat{m_0}\leftarrow$R.encode(m_0,t,q,d)이다. Game 0의 분포 D_0는 아래와 같다.

$$D_0:\begin{cases}pk\leftarrow(a,b=a*x+e),\\c_1=\lfloor(p/q)r*a\rceil,\\c_2=\lfloor(p/q)(Trunc(r*b,l/t))\rceil+\widehat{m_0}\end{cases}$$

Game 1. 이 게임에서는 공개키 b를 R_q에서 랜덤하게 선택하고, 그 외에는 Game 0에서와 동일하게 진행된다. Game 1의 분포 D_1은 아래와 같다.

$$D_1:\begin{cases}pk\leftarrow(a,b=U(R_q)),\\c_1=\lfloor(p/q)r*a\rceil,\\c_2=\lfloor(p/q)(Trunc(r*b,l/t))\rceil+\widehat{m_0}\end{cases}$$

D_0와 D_1은 공개키 b를 생성하는 방법에서 차이

가 있다. D_0에서는 공개키 b가 RLWE 구조이며 D_1에서는 랜덤한 값이기 때문에, RLWE 가정이 성립한다면 D_0와 D_1은 구분 불가능하다.

Game 2. 이 게임에서는 R_p에서 랜덤하게 선택된 u_1을 암호문 c_1으로 사용하고, 그 외에는 **Game 1**에서와 동일하게 진행된다. **Game 2**의 분포 D_2는 아래와 같다.

$$D_2 : \begin{cases} pk \leftarrow (a, b = U(R_q)), \\ c_1 = u_1, \\ c_2 = \lfloor (p/q)(Trunc(r*b, l/t)) \rceil + \widehat{m_0} \end{cases}$$

D_1과 D_2는 암호문 c_1을 생성하는 방법에서 차이가 있다. D_1에서는 암호문 c_1이 RLWR 구조이며 D_2에서는 랜덤한 값이기 때문에, RLWR 가정이 성립한다면 D_1과 D_2는 구분 불가능하다.

Game 3. 이 게임에서는 R_p에서 랜덤하게 선택된 u_2를 이용하여 암호문 c_2를 생성하고, 그 외에는 **Game 2**에서와 동일하게 진행된다. **Game 3**의 분포 D_3는 아래와 같다.

$$D_3 : \begin{cases} pk \leftarrow (a, b = U(R_q)), \\ c_1 = u_1, \\ c_2 = \lfloor Trunc(u_2, l/t) \rceil + \widehat{m_0} \end{cases}$$

D_2과 D_3는 암호문 c_2을 생성하는 방법에서 차이가 있다. D_2에서는 암호문 c_2가 RLWR 구조이며 D_3에서는 랜덤한 값이기 때문에, RLWR 가정이 성립한다면 D_2과 D_3는 구분 불가능하다.

Game 4. 이 게임에서는 $\widehat{m_0}$가 $\widehat{m_1} \leftarrow$ **R.encode**(m_1, t, q, d)으로 바뀌고, 그 외에는 **Game 3**에서와 동일하게 진행된다. **Game 4**의 분포 D_4는 아래와 같다.

$$D_4 : \begin{cases} pk \leftarrow (a, b = U(R_q)), \\ c_1 = u_1, \\ c_2 = \lfloor Trunc(u_2, l/t) \rceil + \widehat{m_1} \end{cases}$$

D_3와 D_4는 다른 메시지에 대한 암호문을 생성한다. D_3와 D_4에서는 one-time pad 형태로 메시지가 암호화되기 때문에, D_3와 D_4는 통계적으로(statistically) 구분 불가능하다.

Game 5. 이 게임에서는 암호문 c_2가 RLWR 구조로 복원되고, 그 외에는 **Game 4**에서와 동일하게 진행된다. **Game 5**의 분포 D_5는 아래와 같다.

$$D_5 : \begin{cases} pk \leftarrow (a, b = U(R_q)), \\ c_1 = u_1, \\ c_2 = \lfloor (p/q)(Trunc(r*b, l/t)) \rceil + \widehat{m_1} \end{cases}$$

Game 3에서와 같이, RLWR 가정이 성립한다면 D_4와 D_5는 구분 불가능하다.

Game 6. 이 게임에서는 암호문 c_1이 RLWR 구조로 복원되고, 그 외에는 **Game 5**에서와 동일하게 진행된다. **Game 6**의 분포 D_6는 아래와 같다.

$$D_6 : \begin{cases} pk \leftarrow (a, b = U(R_q)), \\ c_1 = \lfloor (p/q) r*a \rceil, \\ c_2 = \lfloor (p/q)(Trunc(r*b, l/t)) \rceil + \widehat{m_1} \end{cases}$$

Game 2에서와 같이, RLWR 가정이 성립한다면 D_5와 D_6는 구분 불가능하다.

Game 7. 이 게임에서는 공개키 b가 RLWE 구조로 복원되고, 그 외에는 **Game 6**에서와 동일하게 진행된다. **Game 7**의 분포 D_7는 아래와 같다.

$$D_7 : \begin{cases} pk \leftarrow (a, b = a*x + e), \\ c_1 = \lfloor (p/q) r*a \rceil, \\ c_2 = \lfloor (p/q)(Trunc(r*b, l/t)) \rceil + \widehat{m_1} \end{cases}$$

Game 1에서와 같이, RLWE 가정이 성립한다면 D_6와 D_7은 구분 불가능하다.

결과적으로, RLWE와 RLWR 가정이 성립한다면 m_0를 암호화하는 **Game 0**와 m_1을 암호화하는 **Game 7**은 구분 불가능하다. 따라서 4.2절에서 소개한 KEM 기법은 IND-CPA에 안전하다. □

4.3 선택 암호문 공격(IND-CCA)에 안전한 기법

본 절에서는 양자 랜덤 오라클 모델(QROM)에서 IND-CCA에 안전한 KEM 기법을 제시한다. 제안하는 기법은 4.2절에서 소개한 IND-CPA에 안전한 공개키 암호 기법에 FO 변형 방법을 적용한 것으로, 추가적으로 사용되는 3가지 해시함수는 3.3절에서와 동일하게 정의된다.

4.2절에서 제시한 공개키 암호 알고리즘 (**R.KeyGen**, **R.Encrypt**, **R.Decrypt**)이 주어졌을 때, 선택 암호문 공격에 안전한 KEM 알고리즘 (**R.Setup**, **R.Encap**, **R.Decap**)은 다음과 같이 정의된다.

R.Setup(1^{λ}):
① 4.2절의 **R.Keygen**과 동일하다.

R.Encap(pk):
① 임의의 문자열 $\delta \in \{0,1\}^{256}$를 랜덤하게 선택하고, $z = G(\delta)$ 를 계산한다.
② $(c_1, c_2) \leftarrow$ **R.Encrypt**$(pk, \delta; z)$와 $c_3 = \hat{H}(\delta)$를 계산한다.
③ $K = H(\delta, c_1, c_2, c_3)$를 계산한다.
④ 암호문 $c = (c_1, c_2, c_3)$와 키 $K \in \{0,1\}^{256}$를 출력한다.

R.Decap(sk, c):
① 암호문 $c = (c_1, c_2, c_3)$에 대하여, $\delta \leftarrow$ **R.Decrypt**$(sk, (c_1, c_2))$를 계산한다.
② $z = G(\delta)$를 계산한다.
③ $(d_1, d_2) \leftarrow$**R.Encrypt**$(pk, \delta; z)$와 $d_3 = \hat{H}(\delta)$를 계산한다.
- 만약 $(d_1, d_2) \neq (c_1, c_2)$ 또는 $d_3 \neq c_3$이면, $\perp$를 출력한다.

④ 그렇지 않다면, $K = H(\delta, c_1, c_2, c_3)$를 출력한다.

4.3.1 안전성 증명

4.3절에서 제안한 기법은 **(정리 7)** 랜덤 오라클 모델(ROM)에서는 타이트(tight)하게 IND-CCA에 안전하며, **(정리 8)** 양자 랜덤 오라클 모델(QROM)에서는 타이트하지 않게 IND-CCA에 안전하다.

정리 7 (Theorem 3.1 and 3.2 in [17]). 4.2절에서 제안된 기법이 δ-정확성을 가진다고 가정하자. q_D번의 복호화 질의, q_G번의 랜덤 오라클 G 질의, q_H번의 랜덤 오라클 H 질의를 수행하는 임의의 IND-CCA 공격자 B에 대하여, 아래의 이점을 가지는 IND-CPA 공격자 A가 존재한다.

$$Adv^{CCA}(B) \leq q_H{}^{*}\delta + \frac{q_H + 2q_G + 1}{2^{256}} + 3^{*} Adv^{CPA}(A)$$

정리 8 (Theorem 4.4 and 4.6 in [17]). 4.2절에서 제안된 기법이 δ-정확성을 가진다고 가정하자. q_D번의 복호화 질의, q_G번의 양자 랜덤 오라클 G 질의, q_H번의 양자 랜덤 오라클 H 질의, $q_{\hat{H}}$번의 양자 랜덤 오라클 $\hat{H}$ 질의를 수행하는 임의의 IND-CCA 공격자 B에 대하여, 아래의 이점을 가지는 IND-CPA 공격자 A가 존재한다.

$$Adv^{CCA}(B) \leq (2q_H + q_{\hat{H}}) \cdot \sqrt{8^{*}\delta(q_G+1)^2 + (1+2q_G)\sqrt{Adv^{CPA}(A)}}$$

V. 구현 결과

본 장에서는 3장 및 4장에서 제안한 공개키 암호 기법과 KEM 기법을 구현하고 그 성능을 측정하였다. 또한 LWE 및 RLWE를 기반으로 설계된 암호 기법인 EMBLEM 및 R.EMBLEM과 성능을 비교 분석하였다[12]. 실험은 Windows 10 64bit 운영체제에서 Visual Studio 2017을 이용하여 진행하였으며, AMD Ryzen™ 7 1700 (3.5GHz) 프로세서를 사용하였다.

Table 1은 3장에서 제안한 기법들과 EMBLEM 기법의 구현에 사용된 파라미터를 나타낸다. 메시지 길이는 모든 기법에서 동일하게 256-bit로 설정하였다. Table 2는 LWE 및 LWR 가정을 기반으로 설계된 기법들의 공개키, 비밀키, 그리고 암호문 크기를 나타낸다. EMBLEM.CPA와 본 논문의 3.2절에서 소개된 기법(§3.2)은 IND-CPA에 안전한 기법이며, EMBLEM과 본 논문의 3.3절에서 소개된 기법(§3.3)은 IND-CCA에 안전한 기법이다. 4가지 기법 모두 공개키와 비밀키의 크기는 동일하며, 본 논문에서 제안한 기법이 EMBLEM과 비교했을 때

Table 1. Parameters for LWE-based schemes

Secret distribution $[-B, B]$	[-2,2]
(m, n, k)	(1016, 784, 4)
$(\log_2(q), \log_2(p))$	(24, 22)
σ	25
(t, v)	(8, 8)

Table 2. Comparison of (PK, SK, CT) sizes with EMBLEM

	|PK| (bytes)	|SK| (bytes)	|CT| (bytes)
EMBLEM.CPA	12,224	32	18,912
§3.2			17,336
EMBLEM			18,944
§3.3			17,368

암호문의 크기 측면에서 약 8.3% 더 효율적이다.

Fig. 2는 위의 4가지 기법들의 알고리즘 별 수행 시간을 나타낸다. KeyGen 알고리즘의 경우, 3.3절(3.2절)에서 소개한 기법이 EMBLEM(EMBLEM.CPA)에 비해 수행 시간이 17.24% 감소하였다. IND-CPA에 안전한 기법의 경우, 3.2절에서 소개한 기법이 EMBLEM.CPA에 비해 Encrypt와 Decrypt 알고리즘 수행 시간이 각각 42.4%, 16.3% 감소하였다. IND-CCA에 안전한 기법의 경우, 3.3절에서 소개한 기법이 EMBLEM에 비해 Encap과 Decap 알고리즘 수행 시간이 각각 40%, 37.4% 감소하였다. 모든 알고리즘에서 본 논문에서 제안한 기법이 EMBLEM(EMBLEM.CPA)에 비해 수행 시간이 감소하였음을 확인할 수 있다.

Table 4는 4장에서 제안한 기법들과 R.EMBLEM 기법의 구현에 사용된 파라미터를 나타낸다. 메시지 길이는 모든 기법에서 동일하게 256-bit로 설정하였다. Table 5는 RLWE 및 RLWR 가정을 기반으로 설

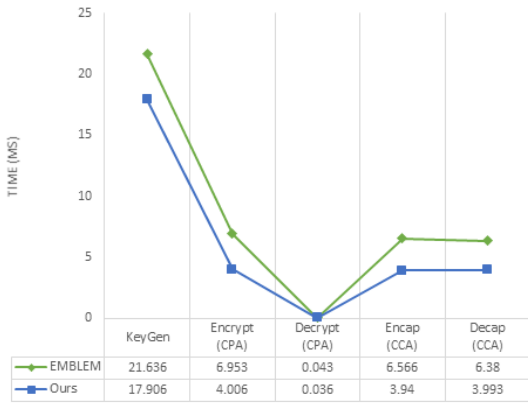

Fig. 2. Comparison of performances with EMBLEM

Table 3. Comparison of ssecurity with EMBLEM

	Primal		Dual	
	EMBLEM	§3.3	EMBLEM	§3.3
Q-Core-Sieve	96	72	112	85
Q-β-Sieve	105	80	138	110
Core-Sieve	106	80	137	105
β-Sieve	114	88	147	117
Lotus	113	72	237	161

Table 4. Parameters for RLWE-based schemes

Secret distribution $[-B,B]$	$\{-1,1\}$
n	1,024
$(\log_2(q), \log_2(p))$	(14, 12)
σ	3
t	1

계된 기법들의 공개키, 비밀키, 그리고 암호문 크기를 나타낸다. R.EMBLEM.CPA와 본 논문의 4.2절에서 소개된 기법(§4.2)은 IND-CPA에 안전한 기법이며, R.EMBLEM과 본 논문의 4.3절에서 소개된 기법(§4.3)은 IND-CCA에 안전한 기법이다. 4가지 기법 모두 공개키와 비밀키의 크기는 동일하며, 본 논문에서 제안한 기법이 R.EMBLEM과 비교했을 때 암호문의 크기 측면에서 약 14% 더 효율적이다.

Fig. 3은 위의 4가지 기법들의 알고리즘 별 수행 시간을 나타낸다. KeyGen 알고리즘의 경우, 4.3절(4.2절)에서 소개한 기법이 R.EMBLEM(R.EMBLEM.CPA)에 비해 수행 시간이 12.6% 감소하였다. IND-CPA에 안전한 기법의 경우, 4.2절에서 소개한 기법이 R.EMBLEM.CPA에 비해 Encrypt와 Decrypt 알고리즘 수행 시간이 각각 47%, 21% 감소하였다. IND-CCA에 안전한 기법

Table 5. Comparison of (PK, SK, CT) sizes with R.EMBLEM

	|PK| (bytes)	|SK| (bytes)	|CT| (bytes)
R.EMBLEM.CPA	1,824	32	2,240
§4.2			1,920
R.EMBLEM			2,272
§4.3			1,952

Table 6. Comparison of security with R.EMBLEM

	Primal		Dual	
	R.EMBLEM	§4.3	R.EMBLEM	§4.3
Q-Core-Sieve	211	190	246	220
Q-β-Sieve	221	199	256	228
Core-Sieve	233	209	271	238
β-Sieve	243	218	277	247
Lotus	359	309	396	336

의 경우, 4.3절에서 소개한 기법이 R.EMBLEM에 비해 Encap과 Decap 알고리즘 수행 시간이 각각 59%, 45.5% 감소하였다. 모든 알고리즘에서 본 논문에서 제안한 기법이 R.EMBLEM(R.EMBLEM .CPA)에 비해 수행 시간이 감소하였음을 확인할 수 있다.

VI. 결 론

본 논문에서는 (R)LWE와 (R)LWR 문제를 이용하여 효율적으로 다중 비트를 암호화하는 기법을 제안하였다. (R)LWE 가정을 기반으로 효율적으로 다중 비트를 암호화하는 기법인 (R.)EMBLEM이 NIST PQC 표준화 과제에 제안되었으나[12], 본 논문에서는 (R)LWR 가정을 추가적으로 사용하여 암호문의 크기를 줄임으로써 전송 효율성을 향상시켰다. 또한 암호화 단계에서 이산 가우시안 분포로부터 오류 벡터를 선택하는 과정이 생략되었기 때문에, 본 논문에서 제안하는 기법은 (R.)EMBLEM에 비해 알고리즘 수행 시간이 단축되었다.

향후 연구로는 LWE와 LWR 문제를 이용한 효율적인 다중 비트 암호화 방법을 함수 암호 등 다양한 암호 프리미티브에 적용하여, 양자 컴퓨팅 환경에서의 안전성도 제공하면서 더 효율적인 기법을 설계하는 것이 필요할 것이다.

References

[1] P.W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," In: Proc. of Foundations of Computer Science, pp. 124-134, Nov. 2004.

[2] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert, "Bonsai trees, or how to delegate a lattice basis," In: Proc. of the EUROCRYPT'10, LNCS 6110, pp. 523-552, 2010.

[3] C. Gentry, "Fully homomorphic encryption using ideal lattics," In: Proc. of the STOC'09, pp. 169-178, 2009.

[4] M. Abdalla, F. Bourse, A.D. Caro, and D. Pointcheval, "Simple functional encryption schemes for inner products," In: Proc. of the PKC'15, LNCS 9020, pp. 733-751, 2015.

[5] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," In: Proc. of the STOC'05, pp. 84-93, 2005.

[6] B. Applebaum, D. Cash, C. Peikert, and A. Sahai, "Fast cryptographic primitives and circular-secure encryption based on hard learning problems," In: Proc. of the CRYPTO'09, LNCS 5677, pp. 595-618, 2009.

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

[8] A. Banerjee, C. Peikert, and A. Rosen, "Pseudorandom functions and lattices," In: Proc. of the EUROCRYPT'12, LCS 7237, pp. 719-737, 2012.

[9] J. Alwen, S. Krenn, K. Pietrzak, and D. Wichs, "Learning with rounding, revisited," In: Proc. of the CRYPTO'13, LNCS 8042, pp. 57-74, 2013.

[10] A. Bogdanov, S. Guo, D. Masny, S. Richelson, and A. Rosen, "On the hardness of learning with rounding over small modulus," In: Proc. of the TCC'16, LNCS 9562, pp. 209-224, 2016.

[11] "NIST Post-Quantum Cryptography standardization,"(https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization)

[12] M. Seo, S. Kim, D.H. Lee, and J.H. Park, "EMBLEM and R.EMBLEM," (https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions)

[13] Bai,Shi, and Steven D. Galbraith. "An improved compression technique for

signatures based on learning with errors." Topics in CryptologyCT-RSA 2014. Springer International Publishing, 2014. 28-47.
〔14〕 V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," In: Proc. of the EUROCRYPT'10, LNCS 6110, pp. 1-23, 2010.
〔15〕 S. Bai and S. D. Galbraith, "Lattice decoding attacks on binary LWE," In: Proc. of the ACISP'14, LNCS 8544, pp. 322-337, 2014.
〔16〕 S. Goldwasser, Y. T. Kalai, C. Peikert, and V. Vaikuntanathan, "Robustness of the learning with errors assumption," 2010.
〔17〕 D. Hofheinz, K. Hovelmanns, and E. Kiltz, "A modular analysis of the fujisaki-okamoto transformation," In: Proc. of the TCC'17, LNCS 10677, pp. 341-371, 2017.

〈저자소개〉

장 초 롱 (Jang Cho Rong) 학생회원
2017년 8월: 상명대학교 컴퓨터과학과 졸업
2017년 9월~현재: 상명대학교 컴퓨터과학과 석사과정
〈관심분야〉 암호프로토콜

서 민 혜 (Minhye Seo) 학생회원
2012년 2월: 고려대학교 수학과 졸업
2012년 3월~현재: 고려대학교 정보보호대학원 석박사 통합과정
〈관심분야〉 암호 프로토콜, 인증 및 키 교환, 함수암호

박 종 환 (Jong Hwan Park) 정회원
1999년 2월: 고려대학교 수학과 졸업
2004년 2월: 고려대학교 정보보호대학원 석사 졸업
2008년 8월: 고려대학교 정보보호대학원 박사 졸업
2013년 9월~현재: 상명대학교 컴퓨터과학과 조교수
〈관심분야〉 함수 암호, 영지식 증명, 암호 프로토콜