**CRYSTALS-KYBER**

**1. Written Spec.**

KYBER 는 IND-CCA2-secure key encapsulation mechanism[1] (KEM) 임.

KYBER의 보안성은 module lattices(MLWE problem)[3] 의 learning with errors 문제를 푸는 것의 어려움에 기반함.

KYBER의 구성은 2가지 단계의 접근방식을 따른다.

첫 번째로 32비트 고정 길이 메시지를 암호화하는 IND-CPA-secure[4] 공개키 암호화 방식을 소개하고, 이를 KYBER.CPAPKE라 부름.

그리고 약간 뒤튼 Fujisaki-Okamoto 변환[5] 을 사용해 IND-CCA2-secure KEM 을 구성한다.

우리는 앞으로 IND-CCA2-secure KEM에 대해 이야기하고 있음을 강조하고 싶을 때 Kyber.CCAKEM 이라고 언급하도록 한다.

1.1절에서는 기본 사항을 설명하고 기호를 정의함.

1.2절에서는 Kyber.CPAPKE의 전체 사양을 제시

1.3절에서는 Kyber.CPAPKE로부터 Kyber.CCAKEM을 얻기 위해 사용하는 Kyber 변환에 대한 상세 정보를 제공

1.4절에서는 저자들이 제안하는 다른 보안 레벨에 대해 대한 파라미터 나열(lists)

1.5절에서는 Kyber 설계의 이유 설명

**1.1 기본 사항과 기호(표기법)**

Kyber에서의 바이트와 비트 배열

Kyber의 모든 API 함수의 입출력은 바이트 배열임.

표기법을 간단히 하기 위해 다음과 같이 정의함

* B : 8비트 부호 없는 정수(바이트)의 집합, 즉 {0, …, 255} 까지의 수
* : 길이가 k인 바이트 배열의 집합
* : 임의 길이 (혹은 바이트 스트림)의 길이의 집합
* (a || b) : a와 b라는 두 바이트 배열을 연결한다. 연결 시에는 a가 먼저, b가 뒤에 오도록 연결
* a + k : 하나의 바이트 배열 a의 k 번째 바이트부터 시작하는 배열을 뜻함.
  + 예시 : a 를 길이가 l 인 바이트 배열이라고 할 때, b를 다른 바이트 배열이라 두고,   
    c를 (a || b)로 둘 때, a와 b의 연결은 b = a + l 임.(a의 l번째 바이트부터 시작하는 배열 ?)

바이트 배열보다 비트 배열로 작업하는 것이 더 편하다면, BytesToBits 함수를 사용해 이 변환을 명시적으로 수행함. 이 함수는 길이가 l인 바이트 배열을 입력으로 받고 8l 비트의 배열을 출력함.

출력 비트 배열의 i 번째 비트 βi는 입력 배열의 계산 결과 인 위치가 i / 8인 바이트 로부터 얻을 수 있다.

Polynomial rings and vectors

이 문서에서 n, n’, q 가 각각 고정되어 있음

n = 256

n’ = 9

q = 3329

R, R\_q, 볼드체 lower-case 문자는 R이나 R\_q의 계수를 가지는 벡터를 나타냄.

볼드체 upper-case 문자는 행렬임.(matrices)

Rounding

Q에 속하는 원소 x에 대해 는 가장 가까운 정수로 반올림

Sizes of elements

Z\_q에 속하는 원소 w에 대해,

우리는  라고 한다.

그리고 우리는 ,  의 크기를  라 함.

텍스트, 폰트, 스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명

Sets and Distributions

집합 S에 대해, s ← S 라고 쓰면 s가 S에서 균등하게 무작위로 선택되었음을 나타낸다. S가 확률 분포 라면, s가 그 분포 S에 따라 선택되었음을 나타냅니다.

Compression and Decompression

이제 우리는

Compress\_q(x, d) 라는 함수를 정의한다.

이는 Z\_q에 속하는 원소 x를 받고, 정수 {0, …, 2^d – 1}를 출력한다. 

더 나아가

Decompress\_q 라는 함수를 정의한다.

 - (1)

 - (2)

이 두 함수는 다음을 만족한다.



Compress\_q 와 Decompress\_q 를 사용할 때  인 경우, 각 계수에 개별적으로 절차를 적용한다.

이 두 함수를 사용하는 이유는 복호화 정확도에 큰 영향을 미치지 않는 암호문의 low-order(?) 비트를 일부 버릴 수 있도록 하여 암호문의 크기를 줄이기 위함

또한 이 두 함수는 압축 외에도 암호화 및 복호화 과정에서 일반적인 LWE(learning with errors) 오류 수정을 수행하는데 사용

더 자세히 살펴보면, 암호화 알고리즘 5의 20번 라인에서 Decompress\_q 함수가 메시지 비트를  로 보내는 방식으로 오차 허용 범위를 만든다.

이후, 복호화 알고리즘의 라인 4번에서, Compress\_q 함수가 사용된다. (v – s^tu 가 q/2 에 가까운 경우 1, 다른 경우 0으로 복호화 함)

Symmetric primitives(대칭 원시 값)

Kyber 의 설계는 pseudorandom function 을 사용하게 만든다.

 과 늘어날 수 있는 출력 함수 .

KYBER 는 또한 두개의 해시 함수를 사용하게 만든다.



NTTs, Multiplication, and bitreversed order

R\_q의 곱셈을 효율적으로 하는 방법은 NTT(number-theoretic transform) 라고 불림.

소수 q = 3329 (q – 1 = 2^8 \* 13)에 대해,

base field 인 Z\_q는 원시 256번째 roots of unity 를 포함하지만 원시 512번째 roots 는 포함하지 않는다.

그러므로, 다항식  

라고 정의함.

(R의 X^256 + 1는 mod q 연산에서 2차 방정식 128개로 인수분해된다.)

(R\_q에 속하는 다항식 f의 NTT는 1차 다항식 128개의 벡터로 표현됨)

우리는 NTT를 비트 반전 순서로 출력값을 재배열하지 않고 직접 구현하는 방식을 사용한다.

구체적으로,  를 첫번째 256번째 root of unity modulo q라고 하면  이다.

다항식 X^256 + 1은 다음과 같이 쓰여질 수 있다.

텍스트, 폰트, 화이트, 라인이(가) 표시된 사진

자동 생성된 설명

br\_7(i) (i = 0, …, 127) 는 부호 없는 7비트 정수 i 의 비트 반전값을 나타냄.

AVX 명령어의 특징과 호환성을 위해 이러한 비트 반전 순서를 사용함.

그러므로 R\_q에 속하는 f의 NTT는 다음과 같음.

 (3)

선형 다항식 벡터를 표준적인(canonical) 방식으로  벡터로 직렬화한다.

그리고, 추가적인 데이터 유형을 도입하지 않고 NTT 알고리즘의 직접 구현을 용이하게 하기 위해,  를 다음과 같이 정의한다.

* R\_q 에 속하는 f 를 입력으로 받는다.
* 앞서 언급한 계수 벡터를 갖는 다항식을 출력값으로 반환한다.



폰트, 텍스트, 화이트, 도표이(가) 표시된 사진

자동 생성된 설명 (4), (5)

라는 결과 도출.

우리는 f^(f hat) 을 R\_q에 속하는 다항식으로 표기하지만, 실제로는 대수적인 의미가 없음

f^ 는 NTT(f) 연산의 결과이지만, (3)의 식과 같이 1차항 128개로 이루어진 벡터로 표현하는 것이 더 자연스러운 대수적 표현임.

(4) 식과 (5)식에서 정의된  값들을 사용하여 128개의 1차 다항식으로 표현하는 것이 더 정확한 의미를 담고 있음. 이는,  임.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

이건 무슨 내용인지..

Uniform sampling in R\_q

KYBER 은 R\_q 링의 원소를 균일한 랜덤 분포에 근접한 값으로 결정론적 방식으로 샘플링함.

이 샘플링을 위해  함수를 사용함.

Parse 함수

* 바이트 스트림  를 입력으로 받음(바이트 스트림은 0과 1로 구성)
* 받은 바이트 스트림을 이용해 NTT 표현   을 연산함

다음은 Parse 알고리즘(q = 3329)

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

Parse 함수에 대한 직관적인 이해는 다음과 같다. 만약 입력 바이트 배열이 통계적으로 균일 난수 바이트 배열과 가까우면, 출력 다항식은 R\_q의 균일 난수 원소로 통계적으로 가까워진다.

그리고 이것은 만약 입력 다항식의 계수가 균일하게 랜덤하게 선택되었다면, 전단사 함수인 NTT를 통과한 후에도 출력 다항식의 계수 역시 균일하게 랜덤한 상태를 유지하게 된다는 것을 보여준다.

Sampling from a binomial distribution

KYBER의 잡음은 중심 이항 분포(centered binomial distribution)[6] 에서 샘플링 됨 

은 다음을 따르도록 정의함.

텍스트, 폰트, 화이트, 도표이(가) 표시된 사진

자동 생성된 설명

R\_q에 속하는 다항식 f 또는 이런 다항식으로 이루어진 벡터가  에서 샘플링된다고 표기할 때는 각 다항식의 계수가 모두 에서 샘플링되었음을 의미한다.

KYBER의 경우, 위와 같은 샘플링을 pseudorandom 함수의  바이트 출력값을 이용해 결정적론적인 방법으로 수행해야 한다.(설명에서는 n = 256으로 고정)

이것은 CBD(centered binomial distriution) 함수에 의해 완료된다.

텍스트, 폰트, 스크린샷, 번호이(가) 표시된 사진

자동 생성된 설명

Encoding and decoding

KYBER 는 두 가지 데이터 타입을 바이트 배열로 직렬화 해야 함.

* 바이트 배열
* 다항식(벡터 포함)

바이트 배열은 본래 바이트 배열과 동일하게 직렬화되고, 이 경우 다항식의 직렬화 및 역직렬화 방법을 정의해야 함.

알고리즘 3에서 함수 Decode 에 대한 pseudocode 설명에서, 이 함수는 바이트 배열을 다항식 에 각각의 계수를 로 역직렬화한다.

우리가 Encode 를 다항식 벡터로 적용할 때(?, apply) 우리는 각각의 다항식을 개별적으로 인코딩하고 출력 바이트 배열을 연결한다.

텍스트, 영수증, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

1.2 Specification of KYBER.CPAPKE

Kyber.CPAPKE는 Lyuba shevsky, Peikert, Regev 세 명의 연구자에 의해 Eurocrypt 2010 컨퍼런스 발표 와 논문 전체에서 소개된 LPR[7] 암호화 방식 (Ring-LWE[8]에 사용)과 유사하다.

이 scheme의 뿌리(기본, 기반)는 Regev가 처음으로 발표한 LWE 기반 암호화 방식에 도달한다.

이것과 Kyber.CPAPKE의 가장 큰 차이점은 KYBER는 Z\_q 라는 ring을 사용하지 않는 것과 비밀 키와 에러 벡터가 작은 계수를 가지고 있다는 것이다.

NTRU 암호 체계에서 처음으로 Z\_q 대신 다항식 링을 사용하는 아이디어가 제시되었다.

반면 비밀 벡터와 잡음(에러)의 대칭성은 이전의 논문에서 매우 유사한 암호화 방식에서도 이미 사용되었지만, 다른 연구를 통해 이러한 대칭성을 사용하는 방식의 안정성이 더욱 엄격하게 분석되었다.

LRP 암호화 방식과 다른 가장 큰 차이점은 Ring-LWE를 쓰는 것 대신 Module-LWE 를 사용하는 것임.

또한, Alkım, Ducas, Pöppelmann, Schwabe의 연구에서 제안된 방식을 이용해 공개 행렬 A를 생성함.

더 나아가, learning-with-rounding 기반 방식에서 사용되는 것과 유사하게 암호문의 하위 비트를 반올림해 암호문의 크기를 줄인다. 이런 낮은 비트 제거는 LWE 기반 암호화 방식에서도 일반적인 기법이며 암호문 크기를 줄이는 효과가 있음

Parameters

KYBER.CPAPKE는 정수 를 이용해 매개변수화 된다.

시작 전에, 문서에서 언급했던 것 처럼 n = 256, q = 3329를 가진다.

1.1 절의 표기법을 사용해 알고리즘 4, 5, 6 에 KYBER.CPAPKE 공개 키 암호화 방식의 키 생성, 암호화, 복호화 알고리즘을 사용한다. 각 알고리즘에 대한 상세 설명은 주석으로 제공된다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 영수증, 폰트이(가) 표시된 사진

자동 생성된 설명

Specification of KYBER.CCAKEM

우리는 이전 하위 절에서 설명한 IND-CPA-secure 공개 키 암호화 방식을 기반으로 KYBER.CCAKEM IND-CCA2-secure KEM을 구축한다. 이 과정에서 약간 수정된 Fujisaki-Okamoto 변환이 사용된다.

알고리즘 7, 8, 9는 KYBER.CCAKEM의 키 생성, 캡슐화, 분해과정을 정의한다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

KYBER 매개변수 집합

우리는 KYBER512, 768, 1024 세 개의 매개변수 집합을 정의했다.

표에 유도된 매개변수  이것은 유효한 KYBER.CCAKEM 암호문을 복호화할 때 실패할 확률이다.

텍스트, 스크린샷, 폰트, 영수증이(가) 표시된 사진

자동 생성된 설명

n은 256으로 설정된다.

목표가 256 비트 엔트로피를 가진 키를 캡슐화하는 것이기 때문

(CPAPKE.Enc 에서 평문 크기를 256비트를 사용함)

더 작은 n은 여러 키 비트를 하나의 다항 계수에 인코딩해야 하고, 이 경우 더 낮은 노이즈 수준이 필요하며, 이는 보안성을 약화시킴.

더 큰 n은 매개변수 k를 통해 보안 수준을 쉽게 조절하기 어렵다.

따라서, n=256일 때 다음 두 가지의 균형을 유지함

* 256비트 키의 효율적인 캡슐화
* 매개변수 k를 통해 조절 가능한 양호한 수준의 보안 유지

q 값에 대한 설명임.

q 선택 기준 : q는 키 크기인 n보다 작은 소수여야 하며, 빠른 NTT 기반 곱셈을 위한 특정 조건(n | (q-1)) 을 만족해야 함.

적합한 소수로, 257과 769가 있지만 이 값들은 CCA(선택 암호문 공격) 보안에 필요한 무시할 수 있는 실패 확률을 달성하기 어려움.

그래서 다음으로 큰 소수인 3329를 선택함.

k값에 대한 설명임

k는 격자의 한 치수를 결정함. n은 항상 256으로 고정되어 있어 k 값을 조정해 격자의 전체 크기를 결정함. 즉, k는 n의 배수가 되도록 선택됨.

k를 바꾸는 것은 보안 단계를 바꾸는 KYBER의 중심이 되는 매커니즘이다.

n \* k (k=2) = 512 (KYBER512)

남은 매개변수 

이것들은 보안, 암호문 크기, 실패 확률 간의 균형을 맞추기 위해 선택됨.

세 가지 매개변수 집합 모두 2^-128 보다 작은 여유 있는 실패 확률을 달성함.

1.5절, 5.5절에서 더 자세하게 다루고 있음.

잡음(노이즈) 매개변수

 : 알고리즘 4의 s 와 e 벡터의 노이즈 레벨 정의

: 알고리즘 5의 e\_1, e\_2 벡터의 노이즈 레벨 정의

 >  (in KYBER512) 의 이유는 1.5절에서 설명한다.

실패 확률 는 Kyber.py 를 통해 계산함.

이는 <https://github.com/pq-crystals/security-estimates> 에 존재

Instantiating PRF, XOF, H, G, KDF

KYBER 사양 완성을 위한 대칭 원시 함수 인스턴스화

FIPS-202 표준은 NIST에서 발표한 표준임. 이를 기반으로 결정될 예정

일반 Kyber 프로토콜:

XOF (확장 가능한 출력 함수): SHAKE-128 알고리즘을 사용합니다.

H (해시 함수): SHA3-256 알고리즘을 사용합니다.

G (해시 함수): SHA3-512 알고리즘을 사용합니다.

PRF (가 pseudo-random function, 의사 난수 함수): SHAKE-256(s || b) 형태로 정의되며, s와 b를 연결한 값을 SHAKE-256 해시 함수에 적용합니다.

KDF (키 도출 함수): SHAKE-256 알고리즘을 사용합니다.

"90s" variant of Kyber:

일반 Kyber 프로토콜과 다른 점

XOF (확장 가능한 출력 함수): AES-256 암호 알고리즘을 CTR 모드 (Counter mode)로 사용합니다. 이때 키는 i이고 논스 (nonce)는 0으로 채워진 12바이트 값 (j)을 사용합니다. CTR 모드의 카운터는 0으로 초기화됩니다.

PRF (가 pseudo-random function, 의사 난수 함수): AES-256 암호 알고리즘을 CTR 모드로 사용합니다. 이때 키는 s이고 논스 (nonce)는 0으로 채워진 12바이트 값 (b)을 사용합니다. CTR 모드의 카운터는 0으로 초기화됩니다.

텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명

[1] IND-CCA2-secure key encapsulation(KEM) : 공개 키 암호화 방식을 사용하며, 선택 암호문 공격(CCA2)[2]에 대한 안정성과 양자 컴퓨팅 공격에 대한 안전성을 가지는 키 전송 메커니즘

By Gemini

[2] CCA2(Chosen Ciphertext Attack) : 선택 암호문 공격, 임의로 선택된 암호문과 일치하는 평문으로부터 암호 키를 알아내기 위해 시도하는 공격이다. CCA1(런치타임), CCA2(적응형 선택 암호문 공격)으로 나뉜다.

By Gemini, [한국정보통신기술협회 정보통신용어사전](https://terms.tta.or.kr/dictionary/dictionaryView.do?word_seq=038476-1)

[3] Adeline Langlois and Damien Stehlé, Worst-Case to Average-Case Reductions for Module Lattices, Laboratoire LIP (U. Lyon, CNRS, ENS Lyon, INRIA, UCBL), 46 Allée d’Italie, 69364 Lyon Cedex 07, France 의 논문에서 ML 상에서 어려움을 보여주고 있음

[4] IND-CPA-secure public key encryption : Indistinguishability under Chosen-Plaintext Attack, 평문을 선택하면 대응하는 암호문을 얻을 수 있는 상황에서의 공격. 공격자가 한꺼번에 선택한 평문들에 대한 암호문이 주어진다는 가정 하에 복호화 키를 찾는 공격

By [해시넷 위키](http://wiki.hash.kr/index.php/%EC%84%A0%ED%83%9D%ED%8F%89%EB%AC%B8%EA%B3%B5%EA%B2%A9)

[5] Fujisaki-Okamoto transform : 공개 키 암호화 방식(PKE) 의 안전성을 증명하는 데 사용되는 기법.

IND-CPA secure PKE 를 IND-CCA secure PKE 로 변환하는 데 사용함. 이 때 ‘랜덤 오라클(해시 함수)’을 사용

By Gemini

[6] 중심 이항 분포 : 일반적인 이항 분포는 성공 확률 p와 시행 횟수 n을 기반으로 합니다. 하지만 중심 이항 분포는 이를 변형하여 평균이 0이 되도록 만들어진 분포입니다. 즉, 성공과 실패 확률이 서로 반이 되도록 조정됩니다. (예: 성공 확률이 1/2, 실패 확률이 1/2)

By Gemini

[7] LPR : LPR은 Lyuba shevsky, Peikert, Regev의 이름을 딴 암호화 방식으로, Ring-LWE (링 학습 암호) 기반으로 작동한다.

[8] Ring-LWE : 격자 암호의 일종, 특정 수학적 구조인 링을 이용해 암호화를 수행한다.

Kyber.CPAPKE는 LPR 방식과 비슷한 개념을 사용하지만, 몇 가지 중요한 차이점이 있습니다. 주된 차이점은 Kyber.CPAPKE가 Ring-LWE 대신 Module-LWE[9] 라는 다른 격자 암호 기반으로 동작한다는 점입니다. 또한 Kyber.CPAPKE는 공개 행렬 생성 방법에서도 LPR과 차별화됩니다.

[9] Module-LWE : 격자 기반 암호화에 사용되는 난해 문제(Hard problem)

격자 구조에서 특정한 구조를 가진 비밀 벡터를 찾는 문제이고, 효율적인 해결 방법이 현재까지는 알려져 있지 않음

Ring-LWE에 비해 더 강력한 보안성 제공 및 공개 키 크기가 작아 효율적임.

+) NTT : 다항식 계산을 빠르게 수행하는 알고리즘