



SMAUG-T

HEΛΛN
CRYPTO LAB



Defense Counterintelligence
Command

격자기반 양자내성 KEM 알고리즘 & 참조 구현 투어

KpqC Winter Camp 2026

2026.02.25.

(주) 크립토랩

김태경

오늘 발표의 구성

SMAUG-T: 격자기반 양자내성 KEM 알고리즘 & 참조 구현 투어

01

KEM이란 무엇이고, 양자내성 KEM이 필요한 이유는 무엇일까?

02

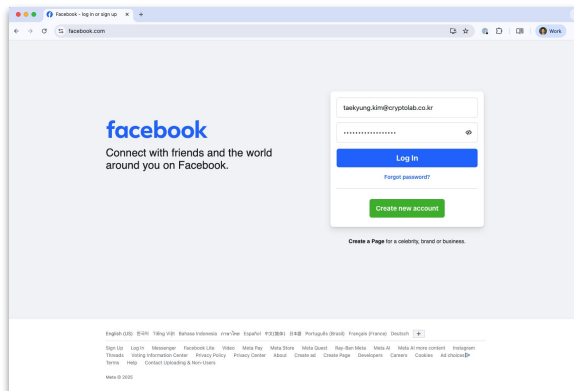
SMAUG-T 알고리즘 개괄, ML-KEM과는 어떤 차이가 있을까?

03

참조코드 walk-through: KeyGen, Encapsulation, Decapsulation API

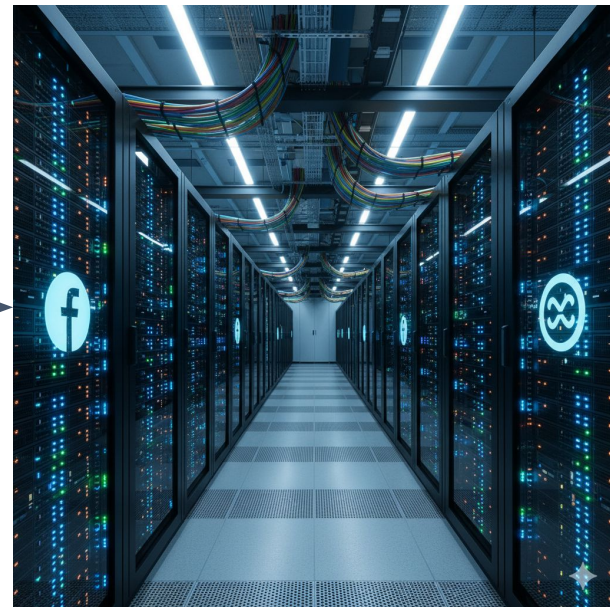
04

파라미터와 실제 사용 가이드라인



```
{ UserID: "taekyung.kim@cryptolab.co.kr",  
  Password: "Strong-and-Complicated" }
```

⚠
누군가 중간에 통신한 내용을
고스란히 가져갈 수 있음.



대칭키 암호

{ UserID: "taekyung.kim@cryptolab.co.kr",
Password: "Strong-and-Complicated" }

87A48727ECC3B1482BCB38B01D2438035A07DC75EA076A9
931D8802237737319D771CCF82BB48232D457700F776D1B
023E50DF0E1DA9C9EA5460BE7546999D581E80340B3D708
854D2A69B3E64932D01C1BAE3BD4AA4834F4D196B7A8132
ECFA

중간에서 정보 유출 불가능,
그러나 사전에 클라이언트와
서버가 키를 합의해야 함!

{ UserID: "taekyung.kim@cryptolab.co.kr",
Password: "Strong-and-Complicated" }

양자내성암호연구단

kpqc.or.kr

양자내성암호연구단

산학연이 함께 만들어가는 암호 대전환 KpqC-X

양자내성암호의 국내 기술력 제고, 국내 PQC 기술 저변 확대 및 경쟁력 강화,
산·학·관·연이 협력하여 선제적 기술개발, 국내 양자내성암호 알고리즘 공모 및 선정

VIEW MORE

공지사항

공모전	워크숍	세미나	교육	기타
KpqC 공모전 최종	2025 양자내성	[2022 KpqC 기술	2026 KpqC	[유관 양자내성

Elements Console Privacy and security x >>

1

Privacy

- Controls
- Third...

Security

- Overvi...
- Main origin
 - Reload to...

Security overview

This page is secure (valid HTTPS).

- Certificate - valid and trusted
 - The connection to this site is using a valid, trusted server certificate issued by Sectigo Public Server Authentication CA DV R36.
 - View certificate
- Connection - secure connection settings
 - The connection to this site is encrypted and authenticated using TLS 1.2, ECDHE_RSA with P-256, and AES_256_GCM.
- Resources - all served securely
 - All resources on this page are served securely.

전자서명, 지금은 안전할까?

현재 TLS 키 교환 흐름

Client Hello

Server Hello + ECDH pubkey

Client: ECDH 계산 → 공유 비밀 K

Server: ECDH 계산 → 공유 비밀 K



양자컴퓨터 → K 복구 가능!

01

이산로그 / RSA 취약

Shor 알고리즘으로 다항식 시간 내 해독 가능

02

Harvest Now, Decrypt Later

현재 트래픽 수집해두고 양자컴퓨터로 나중에 복호화.

03

규제 대응 필요

국내 및 미국 등 주요국가 가이드라인:
2030년대 PQC 전환 권고.

Key Encapsulation Mechanism



Alice

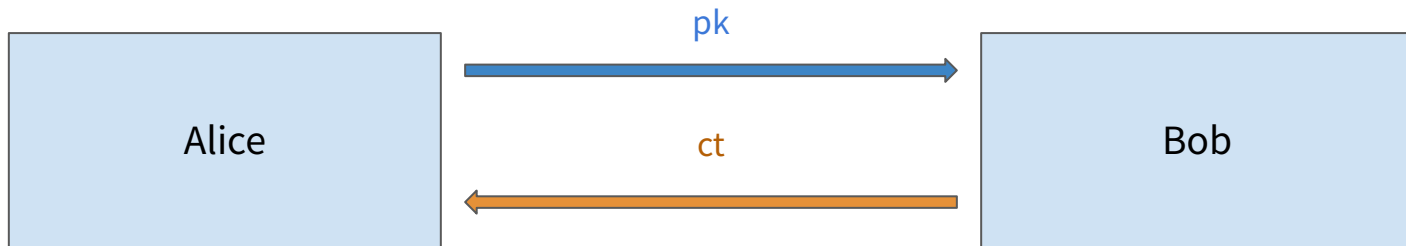


Bob

IDEA

비대칭 암호화 방식을 사용하여, 공개키를 가지고 있는 쪽에서 대칭키 용 비밀을 만들어 비밀키가 있는 쪽에 보낸다.

KEM과 SMAUG-T



$K \leftarrow \text{Decap}(\text{sk}, \text{ct})$

$\text{ct}, K \leftarrow \text{Encap}(\text{pk})$

ML-KEM (Crystals-Kyber)

NIST 표준 (FIPS 203)

SMAUG-T

KpqC 한국 표준 후보,
국내 공공기관·금융·의료
섹터 등에 최적화 (인증 등)

주요 공통점과 차이점

공통점

- MLWE

차이점

- (SMAUG-T) MLWR 기반 암호문
- 키 크기 최적화



SMAUG-T

HEAΛΛN
CRYPTO LAB



Defense Counterintelligence
Command

크립토랩 & 서울대학교 & 국군방첩사령부 & 국방부



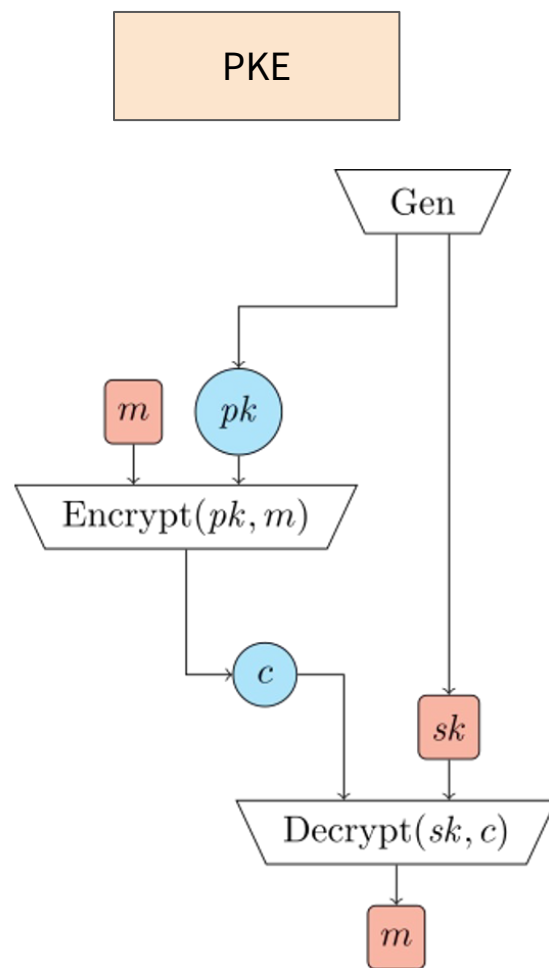
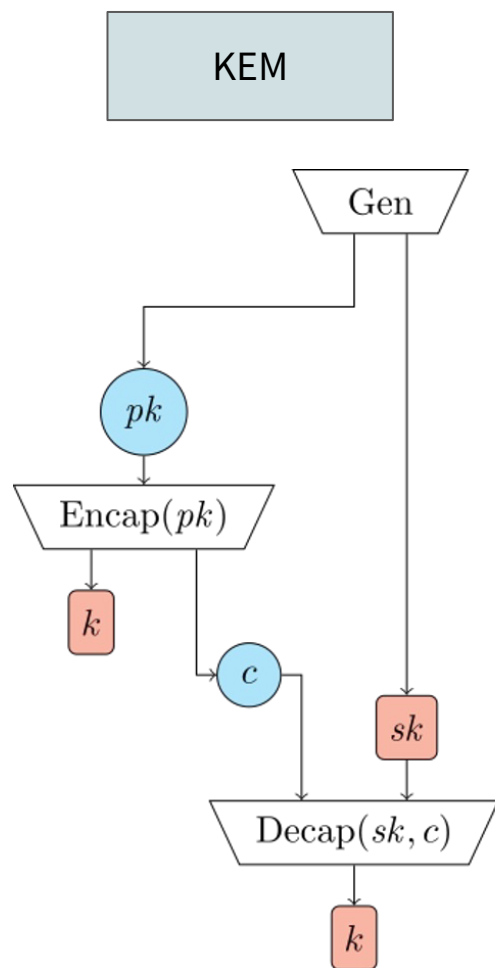
서울대학교
SEOUL NATIONAL UNIVERSITY



국군방첩사령부
Defense Counterintelligence Command

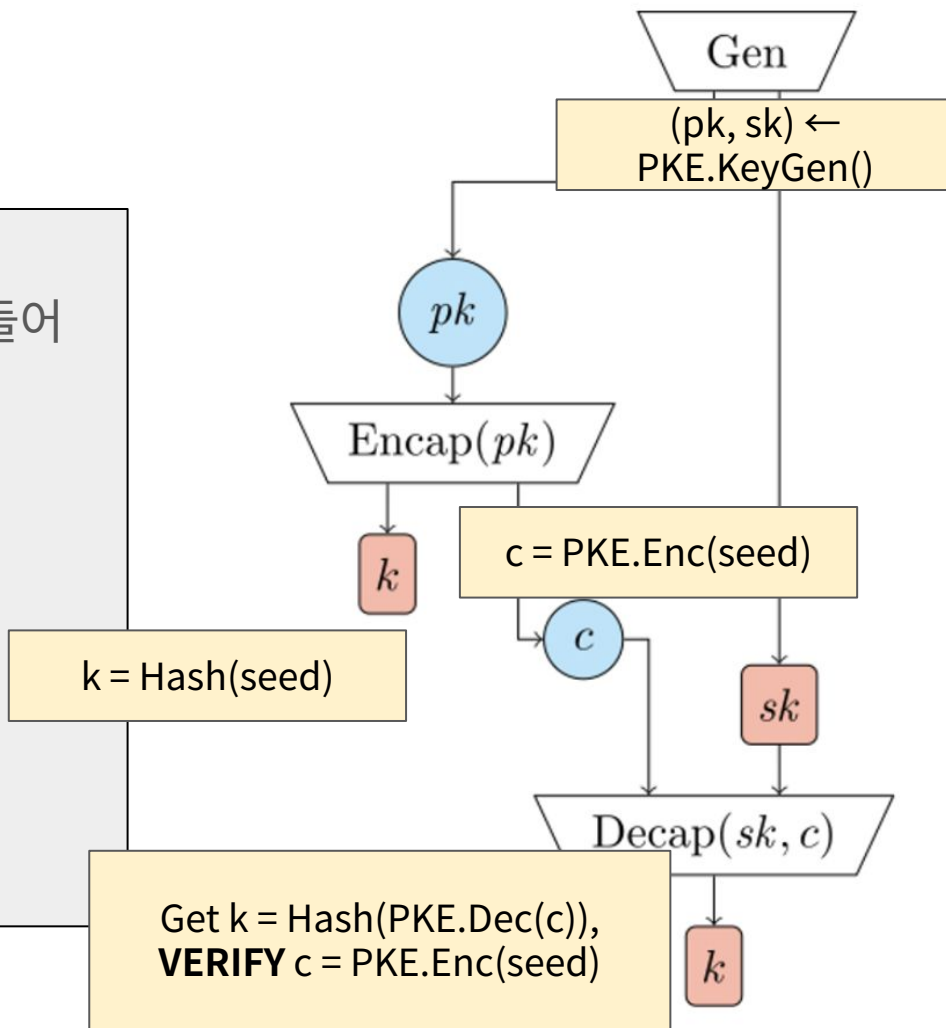


SMAUG-T (& ML-KEM) 의 수학적 기반

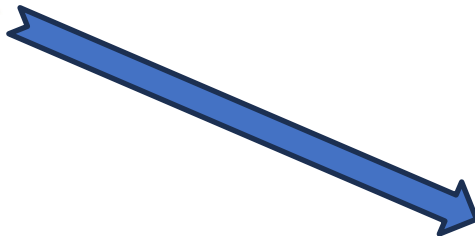
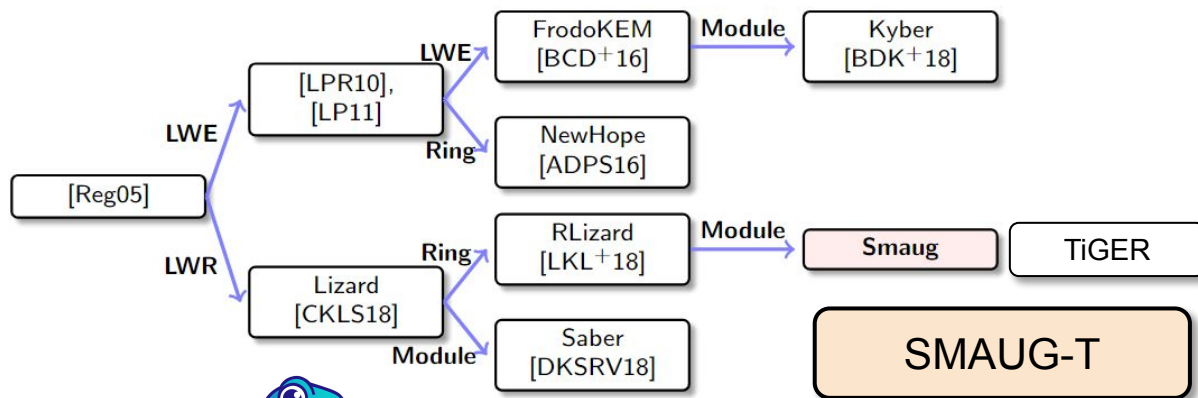


KEM from PKE

- 일반적인 PKE로부터, 비밀을 암호화하는 방식으로 KEM을 만들어냄.
- 안전성
 - 일반적 PKE: **IND-CPA**
 - KEM: **IND-CCA2** 안전성 필요.
- 이를 위해 **Fujisaki-Okamoto Transform (FO)** 수행.



결국, PKE를 제대로 construct하는 것이 핵심!
→ KEM은 FO를 통해 공짜로 얻어짐.

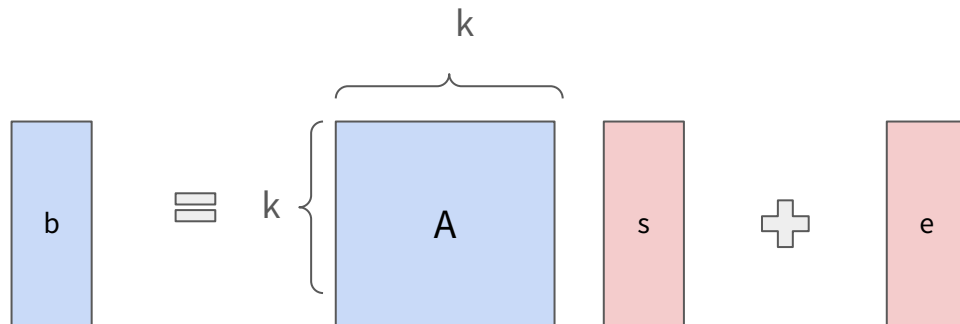


Module-LWE (MLWE) 문제

$$R_q = \mathbb{Z}_q[X] / (X^N + 1)$$

- 행렬 $A \in R_q^{k \times k}$
- 벡터 $s \in R_q^k$ (계수가 작은 다항식들)
- 벡터 $e \in R_q^k$ (계수가 작은 다항식들)

이 주어져 있을 때, $b = A \cdot s + e$ 와 A 를 공개해도 s 와 e 를 알아내기는 어려움.



PKE.KeyGen and PKE.Encryption

MLWE Public Key
Generation

pk: $\left(\begin{matrix} \overset{k}{\underbrace{\quad}} \\ \underbrace{\quad}_k \end{matrix} \begin{bmatrix} \text{A} \end{bmatrix}, \mathbf{b} = \begin{bmatrix} \text{A} \end{bmatrix} \begin{bmatrix} \mathbf{s} \end{bmatrix} + \begin{bmatrix} \mathbf{e} \end{bmatrix} \right), \text{sk: } \begin{bmatrix} \mathbf{s} \end{bmatrix}$

Each elt. is polynomial in R_q

$\mathbf{A} \in R_q^{k \times k}, \mathbf{b} \in R_q^k \text{ for } k = 2, 3, 4$

MLWR Encryption

$\left(\left[\frac{p}{q} \cdot \begin{bmatrix} \mathbf{r} \end{bmatrix} \begin{bmatrix} \text{A} \end{bmatrix} \right], \left[\frac{p'}{q} \cdot \begin{bmatrix} \mathbf{r} \end{bmatrix} \begin{bmatrix} \mathbf{b} \end{bmatrix} + \frac{p'}{q} \cdot \begin{bmatrix} \Delta \\ \mathbf{M} \end{bmatrix} \right)$

$\mathbf{r} \in R_q^k$

PKE.Decryption

$$\left(\left[\frac{p}{q} \cdot \boxed{\mathbf{r}} \quad \boxed{\mathbf{A}} \right], \left[\frac{p'}{q} \cdot \boxed{\mathbf{r}} \quad \boxed{\mathbf{b}} + \frac{p'}{q} \cdot \boxed{\begin{smallmatrix} \Delta \\ \mathbf{M} \end{smallmatrix}} \right] \right)$$

c1 c2

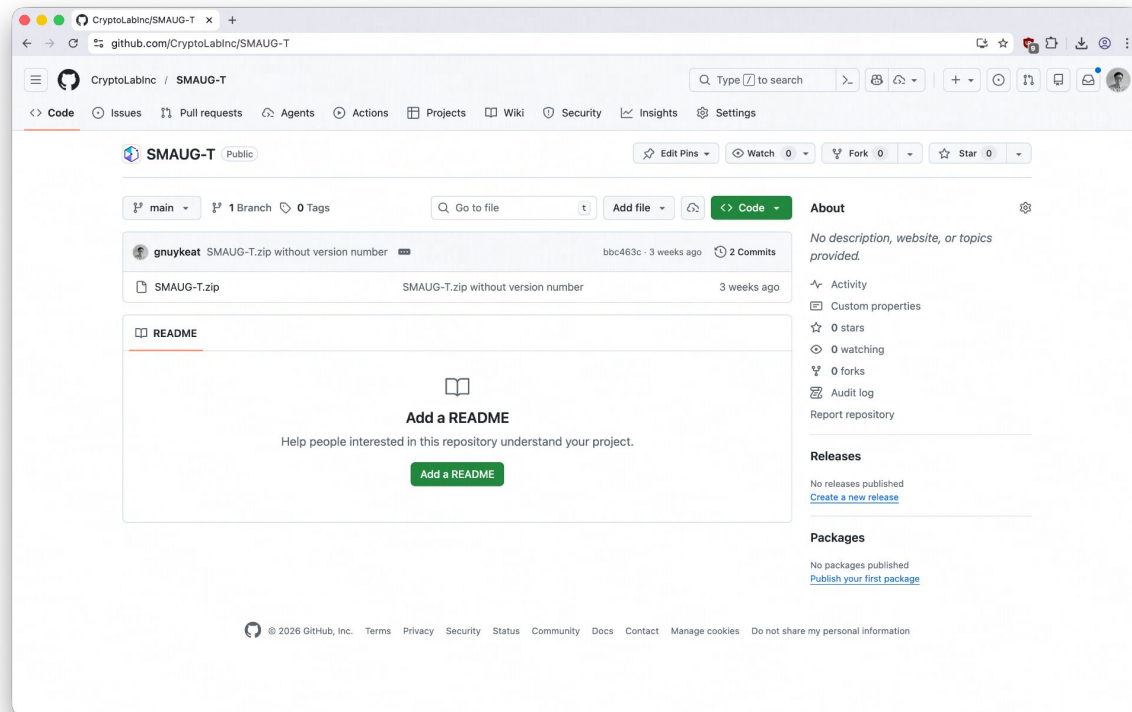
- $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$
- 복호화는 기본적으로 - $\mathbf{c1} \cdot \mathbf{s} + \mathbf{c2}$ 를 계산.
- 반올림에서 발생하는 에러를 control하지 못하면 복호화 실패 발생!

다양한 랜덤 샘플링 기법 활용

Component	Distribution	Description	Benefit
Long-Term Secret	Sparse Ternary, Fixed Hamming Weight Sampling (HWT)	A ternary polynomial vector where the number of non-zero elements is fixed to a specific Hamming weight (h_s)	<ul style="list-style-type: none">• Smaller decryption failure rates.• Constant-time execution.
Ephemeral Secret	Sparse Ternary, Sparse Centered Binomial Distribution (spCBD)	A sparse ternary polynomial vector whose entries are following a binomial distribution.	<ul style="list-style-type: none">• Smaller ciphertext size & lower decryption failure rate while maintaining security (compared to HWT).
Noise	Discrete Gaussian Distribution (dGaussian)	A discrete Gaussian distribution with a small standard deviation ($\sigma \approx 1.06$).	<ul style="list-style-type: none">• Conservative security.

SMAUG-T 참조 코드 walk-through

<https://github.com/CryptoLabInc/SMAUG-T>



핵심 API

KEYGEN

```
int crypto_kem_keypair(uint8_t* pk, uint8_t* sk)
```

공개키/비밀키 쌍 생성

ENCAPSULATION

```
int crypto_kem_enc(uint8_t *ctxt,  
                  uint8_t *ss, const uint8_t *pk);
```

공유 비밀 캡슐화 (클라이언트)

DECAPSULATION

```
int crypto_kem_dec(uint8_t *ss, const uint8_t *ctxt,  
                  const uint8_t *sk)
```

공유 비밀 복호화 (서버)

```
int kem_test() {
    uint8_t pk[SMAUGT_PUBLICKEY_BYTES] = {0};
    uint8_t sk[SMAUGT_KEM_SECRETKEY_BYTES] = {0};

    crypto_kem_keypair(pk, sk);

    uint8_t ctxt[SMAUGT_CIPHERTEXT_BYTES] = {0};
    uint8_t ss[SMAUGT_CRYPT_BYTES] = {0}, ss2[SMAUGT_CRYPT_BYTES] = {0};
    crypto_kem_enc(ctxt, ss, pk);

    int res = crypto_kem_dec(ss2, ctxt, sk);

    if (memcmp(ss, ss2, SMAUGT_CRYPT_BYTES) != 0) {
        printf("\n");
        for (int i = 0; i < m_size; ++i) {
            printf("0x%2hx ", ss[i]);
        }
        printf("\n");

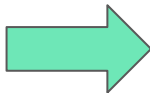
        for (int i = 0; i < m_size; ++i) {
            printf("0x%2hx ", ss2[i]);
        }
        printf("\n");
    }

    return res;
}
```

기존 ECDH → HAETAE 교체 예시

BEFORE (ECDH, OpenSSL)

```
// ECDH 키 교환
EC_KEY *key = EC_KEY_new()
EC_KEY_generate_key(key);
// pubkey 전송, DH 연산...
ECDH_compute_key(
    shared, len,
    peer_pubkey, key, NULL
);
// 양자 취약! ⚠
```



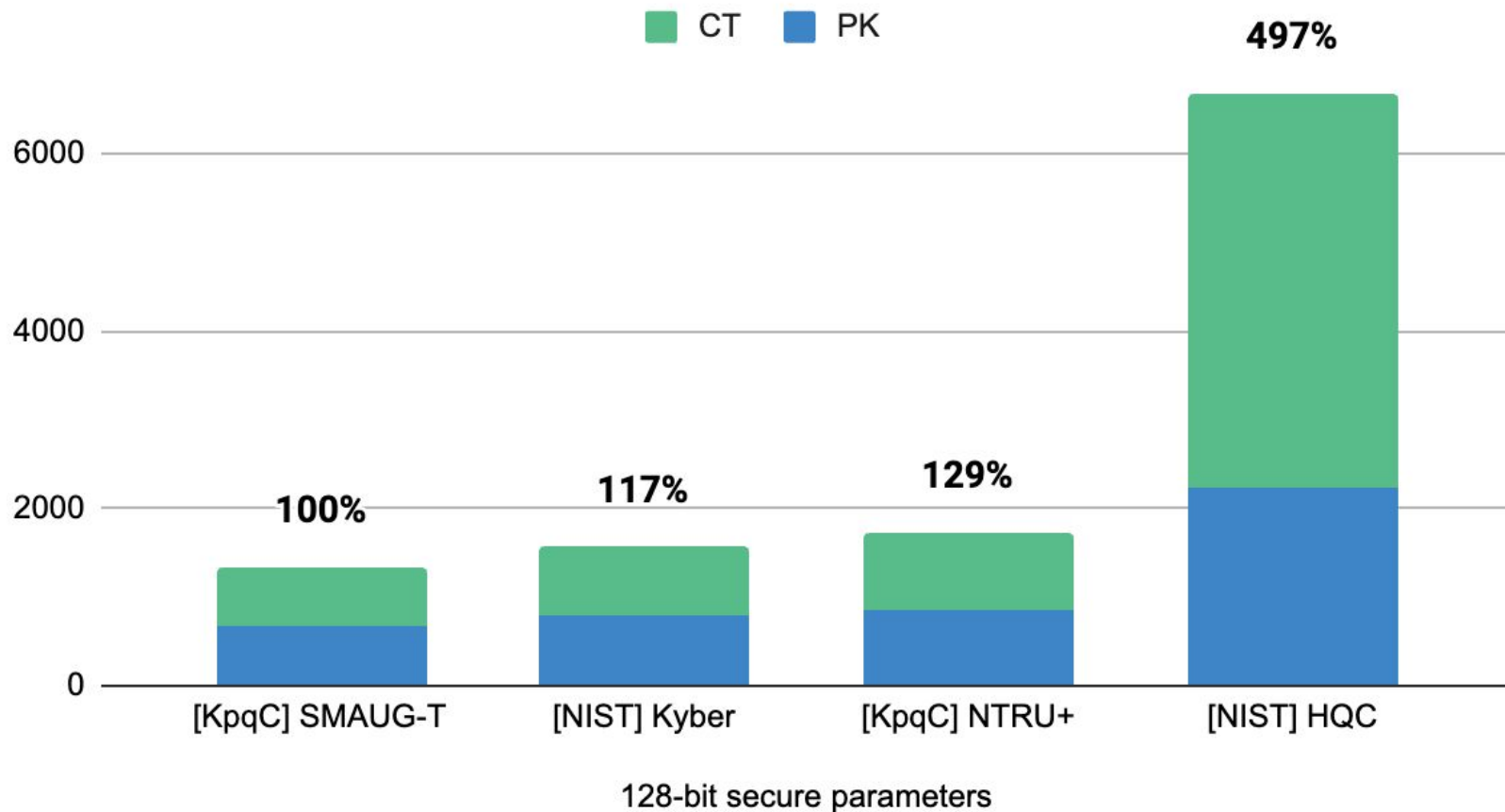
AFTER (SMAUG-T)

```
// SMAUG-T KEM 교체
crypto_kem_keypair(pk, sk);
// Encapsulation
crypto_kem_enc(
    ct, shared, pk
);
// Decapsulation
crypto_kem_dec(shared, ct, sk);
// 양자 안전 ✓
```

파라미터, 실제 사용 가이드라인

Parameters	MODE = 1	MODE = 3	MODE = 5	MODE = TiMER
Polynomial Dimension (N)	256			
Module Rank (K)	2	3	4	2
PK Modulus Size (LogQ)	10	11	11	10
ct1 Modulus Size (LogP1)	8	9	9	8
ct2 Modulus Size (LogP2)	5	4	7	3
Encapsulation Key Size (Bytes)	672	1,088	1,440	672
Decapsulation Key Size (Bytes)	832	1,312	1,728	832
Ciphertext Size (Bytes)	672	992	1,376	608
Shared Secret Size (Bytes)	32			
Decryption Failure Rate	2^{-118}	2^{-179}	2^{-194}	2^{-161}

Size in bytes



감사합니다!

Contact: 김태경
(taekyung.kim@cryptolab.co.kr)

