KpqC 공모전 격자기반 알고리즘 증명 가능한 안전성 분석 기술 연구

2023.07.13.

Sangmyung University. Dept. of Computer Science

Jong Hwan Park

KpqC 공모전

❖ 국내 양자내성암호 국가공모전(KpqC)

◆ 1차 라운드 암호화 기법 7개, 서명 기법 9개 제출 (22.10)

기법	기반문제	알고리즘
KEM	Lattice	NTRU+
	Lattice	SMAUG
	Lattice	TiGER
	Code	REDOG
	Code	PALOMA
	Code	Layered ROLLO-I
	Graph PDF	IPCC
Signature	Lattice	GCKSign
	Lattice	HAETAE
	Lattice	NCC_Sign
	Lattice	Peregrine
	Lattice	SOLMAE
	Code	Enhanced pqsigRM
	Isogeny	FIBS
	Hash	AIMER
	Multivariate	MQ-Sign

[KpqC 공모전 주요 일정(안)]

시기	내용	비고
'22. 2. 18.	'개발 계획서' 접수 마감	
'22. 3. 18.	'개발 계획서' 평가 완료	결과는 개별 통보 예정
'22. 7.	2022 KpqC 1차 워크숍	알고리즘 설계 현황 발표
'22. 10.	'1라운드 제안서' 접수 마감	
	- KpqC 공모전 1라운드 -	
'22. 11.	2022 KpqC 2차 워크숍	1라운드 제안 알고리즘 발표
'23. 7./11.	2023 KpqC 1/2차 워크숍	알고리즘 분석/개선 결과 공유
'23. 12.	공모전 1라운드 결과 발표	2라운드 후보 목록 공개
'24. 2.	'2라운드 제안서' 접수 마감	
	- KpqC 공모전 2라운드 -	
'24. 3.	2024 KpqC 1차 워크숍	2라운드 제안 알고리즘 발표
'24. 9.	2024 KpqC 2차 워크숍	알고리즘 분석/개선 결과 공유
	KpqC 공모전 최종 결과 발표	알고리즘 ○종 선정 예정

KpqC 공모전 알고리즘 안전성 분석 연구

❖ 격자 기반 알고리즘의 설계 원리 및 안전성 증명 논리 분석 연구

◆ 1차 라운드 제출 격자 기반 알고리즘

- KEM 기법 : SMAUG, TiGER, NTRU+
- 서명 기법 : GCKSign, HAETAE, NCC-Sign, Peregrine, SOLMAE

◆ 격자 기반 알고리즘의 증명 가능한 안전성 분석 기술 연구

- ElGamal 기반 KEM (SMAUG, TiGER) 및 NTRU 기반 KEM의 설계 원리 및 안전성 증명 논리 분석
  - 기반 난제, 안전성 증명 논리, 복호화 실패 확률 검증
- Fiat-Shamir 기반 서명 (GCKSign, HAETAE, NCC-Sign)의 설계 원리 및 안전성 증명 논리 분석
  - 기반 난제, 안전성 증명 논리 (영지식성), Rejection sampling 검증

기법	알고리즘	기반 구조
KEM	NTRU+	NTRU
	SMAUG	ElGamal
	TiGER	ElGamal
Signature	GCKSign	Fiat-Shamir
	HAETAE	Fiat-Shamir
	NCC_Sign	Fiat-Shamir

LWE-based Encryption – Basic Concept

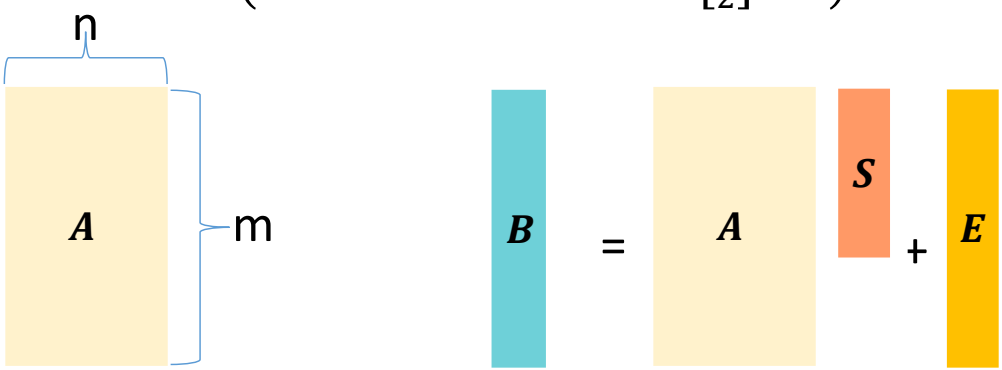
❖ LWE-based encryption scheme*

- **Public key :** $(A \in \boldsymbol{Z_q^{m \times n}},\ B = AS + E)$ **Secret key :** S
- **Ciphertext :** $(C_1, C_2) = \left(RA + E_1,\ RB + E_2 + \left\lfloor \frac{q}{2} \right\rfloor \cdot m\right)$ $(\mathbf{m} \in \{\mathbf{0}, \mathbf{1}\})$

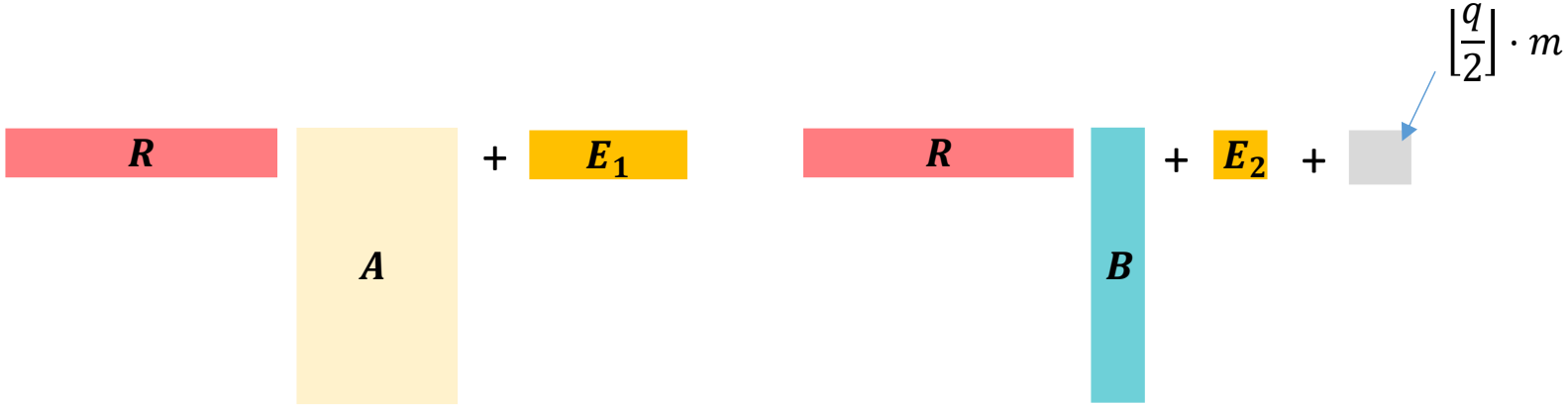
$S, E, R, E_1, E_2 \leftarrow$ 

Discrete Gaussian

PK (LWE) :



CT (LWE) :



* [LP11] Better Key Sizes (and Attacks) for LWE-Based Encryption, CT-RSA 2011.

❖ Security Proof (Sketch)

◆ IND-CPA Security Proof

- G_0 : The original IND-CPA game
- G_1 : Public key → random
- G_2 : Ciphertext → random

Decisional LWE

Decisional LWE

$$|Adv_0 - Adv_1| < Adv^{LWE}_{m,n,q,\chi}$$

$$|Adv_1 - Adv_2| < Adv^{LWE}_{m+1,n,q,\chi}$$

◆ Fujisaki-Okamoto Transform (QROM)

IND-CPA PKE

$(pk, sk) \leftarrow PKE.Gen$

$c \leftarrow PKE.Enc(pk, m; r)$

$m \leftarrow PKE.Dec(sk, c)$

Fujisaki-Okamoto Transform
(including QROM)

IND-CCA KEM

$KEM.Gen$:
$(pk, sk) \leftarrow PKE.Gen, s \leftarrow \{0,1\}^d$

$KEM.Encap(pk)$:
$c \leftarrow Enc(pk, m'; H(m)), K := F(m)$

$KEM.Decap((sk, s), c)$:
$F(m)$ if $m \neq \perp$
$F(c, s)$ if $m = \perp$

❖ Multi-user Security via pk Hashing [DHK+21]

◆ Multi-user Security

- Security against adversaries having access to multiple public keys

(n, q_C)-IND-CPA

01 **for** $j \in [n]$
02 $(pk_j, sk_j) \xleftarrow{\$}$ Gen
03 $\vec{pk} \leftarrow (pk_1, \ldots, pk_n)$
04 $b \xleftarrow{\$} \{0,1\}$
05 $b' \xleftarrow{\$} \mathcal{A}^{\mathsf{Chall}}(\vec{pk})$
06 **return** $[\![b' = b]\!]$

Chall(j, m_0, m_1) / max. q_C queries
07 **return** Enc(pk_j, m_b)

◆ FO Transform with pk Hashing

IND-CPA PKE

$(pk, sk) \leftarrow PKE.Gen$

$c \leftarrow PKE.Enc(pk, m; r)$

$m \leftarrow PKE.Dec(sk, c)$

Fujisaki-Okamoto Transform
with pk hashing
(including QROM)

IND-CCA KEM

$KEM.Gen$:
$(pk, sk) \leftarrow PKE.Gen, s \leftarrow \{0,1\}^d$

$KEM.Encap(pk)$:
$(K, r) \leftarrow F(H(pk), m), c \leftarrow Enc(pk, m'; r)$

$KEM.Decap((sk, s), c)$:
$F(H(pk), m)$ if $m \neq \perp$
$F(H(pk), s, c)$ if $m = \perp$

[DHK+21] Julien Duman et al., "Faster Lattice-Based KEMs via a Generic Fujisaki-Okamoto Transform Using Prefix Hashing", ACM CCS 2021

❖ Multi-user Security via pk Hashing [DHK+21]

◆ Advantages in Multi-user Setting

FO variant	$\mathrm{Adv}^{(n,q_C)\text{-IND-CCA}}_{\mathrm{KEM}}$ (ROM)	$\mathrm{Adv}^{(n,q_C)\text{-IND-CCA}}_{\mathrm{KEM}}$ (QROM)
Include $ID(pk)$ in hash: $\mathrm{FO}^{\not\perp}_{\mathrm{ID}(pk),m}$ (Th. 3.1+3.2)	$\mathrm{Adv}^{(n,q_C)\text{-IND-CPA}}_{\mathrm{PKE}} + q_F\delta(n) + \frac{n^2}{2^\ell}$	$\sqrt{q_F \mathrm{Adv}^{(n,q_C)\text{-IND-CPA}}_{\mathrm{PKE}}} + q_F^2\delta(n) + \frac{n^2}{2^\ell}$
Include pk in hash: $\mathrm{FO}^{\not\perp}_{pk,m}$ (Th. 3.1+3.2)	$\mathrm{Adv}^{(n,q_C)\text{-IND-CPA}}_{\mathrm{PKE}} + q_F\delta(n)$	$\sqrt{q_F \mathrm{Adv}^{(n,q_C)\text{-IND-CPA}}_{\mathrm{PKE}}} + q_F^2\delta(n)$
Original FO transform: $\mathrm{FO}^{\not\perp}_{m}$ (Th. 3.1+[6, 21])	$\mathrm{Adv}^{(n,q_C)\text{-IND-CPA}}_{\mathrm{PKE}} + n \cdot q_F\delta(1)$	$nq_C \cdot (\sqrt{q_F \mathrm{Adv}^{\text{IND-CPA}}_{\mathrm{PKE}}} + q_F^2\delta(1))$

- Required minimal bound:

$$\frac{\text{Time(A)}}{\text{Adv(A)}} \geq 2^{\gamma} \quad \text{for } \gamma\text{–bit security}$$

- For secret keys with the same norm, we have $\delta(n) \approx \delta(1)$
  → good news for TiGER and SMAUG, but . . .
- To achieve γ-bit security (in multi-user setting)

$$\frac{\text{Time(A)}}{\text{Adv(A)}} \geq \frac{q_F}{q_F\delta(n)} = \frac{1}{\delta(1)} \geq 2^{\gamma} \rightarrow \delta(1) \leq 2^{-\gamma}$$

[DHK+21] Julien Duman et al., "Faster Lattice-Based KEMs via a Generic Fujisaki-Okamoto Transform Using Prefix Hashing", ACM CCS 2021

❖ When H is a (real) hash function

◆ Minimum Min-Entropy of message

- By Leftover Hash Lemma (LHL), for
  - Want l-bit secret K — $h: \{0,1\}^* \to \{0,1\}^l$
  - Min-entropy of message space M — $H_\infty(M)$
  - Statistical dist. between hash output and uniform — ϵ

$$l \leq H_\infty(M) - 2\log\left(\frac{1}{\epsilon}\right)$$

- (e.g.) If $l = 128$ and $\epsilon = 2^{-64}$, then we need

$$256 \leq H_\infty(M).$$

That is, message should be uniformly random in $\{0,1\}^{256}$

$KEM.Encap(pk)$:

$c \leftarrow Enc(pk, m'; G(m))$,
$K := H(m)$
return c, K

❖ When H is modeled as a random oracle

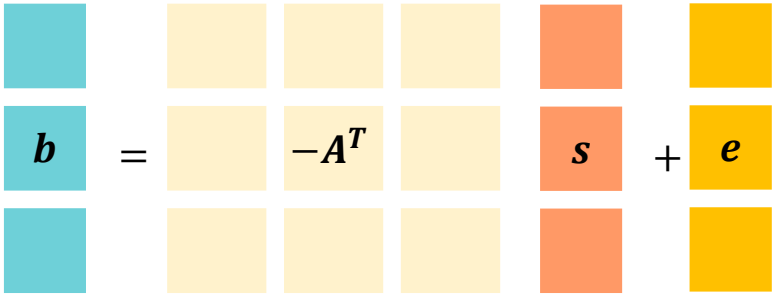
◆ $l \approx H_\infty(M)$

❖ IND-CPA PKE scheme

MLWE MLWR

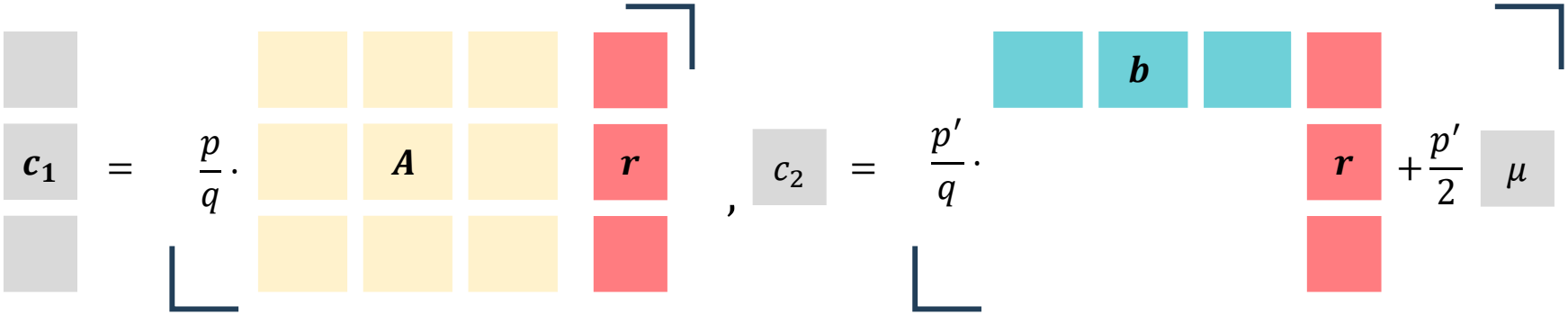
- **Public key :** $(A,\ b = -A^T s + e) \in R_q^{k\times k} \times R_q^k$ **Secret key :** s
- **Ciphertext :** $(c_1, c_2) = (\lfloor p/q \cdot A \cdot r \rceil,\ \lfloor p'/q \cdot \langle b, r\rangle + p'/t \cdot \mu \rceil) \in R_p^k \times R_{p'}$

PK (MLWE):



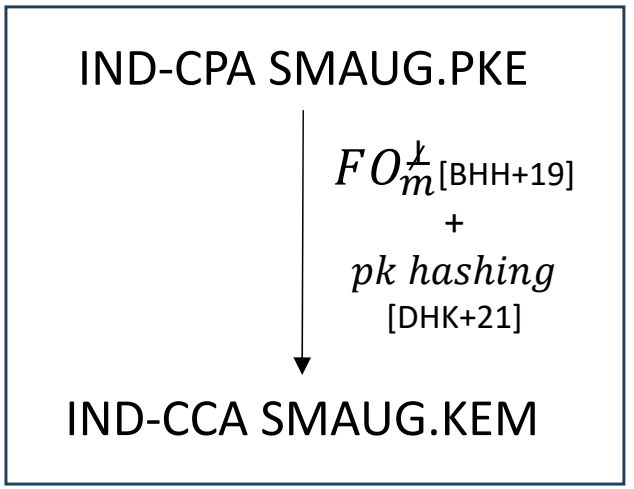
$s, r \leftarrow$ ternary, fixed hamming weight
$e \leftarrow$ discrete Gaussian

CT (MLWR):



Dec : $\mu' = \lfloor 2/p \cdot \langle c_1, s\rangle + 2/p' \cdot c_2 \rceil$

❖ IND-CCA KEM scheme



◆ Key Generation

1. $(pk, sk') \leftarrow PKE.KeyGen(1^\lambda)$
2. $d \leftarrow \{0,1\}^{256}$
3. **return** $pk, sk = (sk', d)$

◆ Encapsulation

1. $\mu \leftarrow \{0,1\}^{256}$
2. $(K, seed) \leftarrow G(\mu, H(pk))$ — pk hashing : prevents multi-user attack
3. $ct \leftarrow PKE.Enc(pk, \mu; seed)$ de-randomization
4. **return** ct, K

◆ Decapsulation

1. $\mu' = PKE.Dec(sk', ct)$
2. $(K', seed') \leftarrow G(\mu, H(pk))$
3. $ct' = PKE.Enc(pk, \mu'; seed)$ re-encryption
4. if $ct \neq ct'$ then

   $(K', \cdot) \leftarrow G(d, H(ct))$ implicit rejection
5. **return** K'

[BHH+18] Nina Bindel et al., "Tighter proofs of CCA security in the quantum random oracle model", TCC 2019
[DHK+21] Julien Duman et al., "Faster Lattice-Based KEMs via a Generic Fujisaki-Okamoto Transform Using Prefix Hashing", ACM CCS 2021

❖ Decryption Failure Probability (DFP)

◆ Rounding Errors

- $\boldsymbol{e}_1 = \frac{q}{p}\left\lfloor \frac{p}{q} \cdot \boldsymbol{A} \cdot \boldsymbol{r} \right\rceil - \boldsymbol{A} \cdot \boldsymbol{r}$
- $e_2 = \frac{q}{p'}\left\lfloor \frac{p'}{q} \cdot \langle \boldsymbol{b}, \boldsymbol{r} \rangle \right\rceil - \langle \boldsymbol{b}, \boldsymbol{r} \rangle$

◆ DFP = Prob. of at least 1 Bit Error

- $\delta = \Pr\left[\| \langle \boldsymbol{b}, \boldsymbol{r} \rangle + \langle \boldsymbol{e_1}, \boldsymbol{s} \rangle + e_2 \|_\infty > \frac{q}{4} \right]$

Parameter Sets	DFP (log_2)	
	SMAUG spec.	Our results
SMAUG128	-119.6	-120.1
SMAUG192	-136.1	-136.9
SMAUG256	-167.2	-167.9

❖ IND-CPA PKE scheme

RLWR

RLWER(+ECC)

- **Public key :** $(a,\ \boldsymbol{b} = \lfloor p/q \cdot \boldsymbol{a} * \boldsymbol{s} \rceil) \in \boldsymbol{R_q} \times \boldsymbol{R_p}$ **Secret key :** $(\boldsymbol{s})$
- **Ciphertext :** $(\boldsymbol{c_1}, c_2) = (\lfloor k_1/q \cdot \boldsymbol{a} * \boldsymbol{r} + \boldsymbol{e_1} \rceil,\ \lfloor k_2/q \cdot (q/p \cdot \boldsymbol{b} * \boldsymbol{r} + \boldsymbol{e_2} + q/2 \cdot \boldsymbol{\mu_{ecc}}) \rceil) \in \boldsymbol{R_{k_1}} \times \boldsymbol{R_{k_2}}$

$\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{e_1}, \boldsymbol{e_2} \leftarrow$ ternary, fixed hamming weight

PK (RLWR):

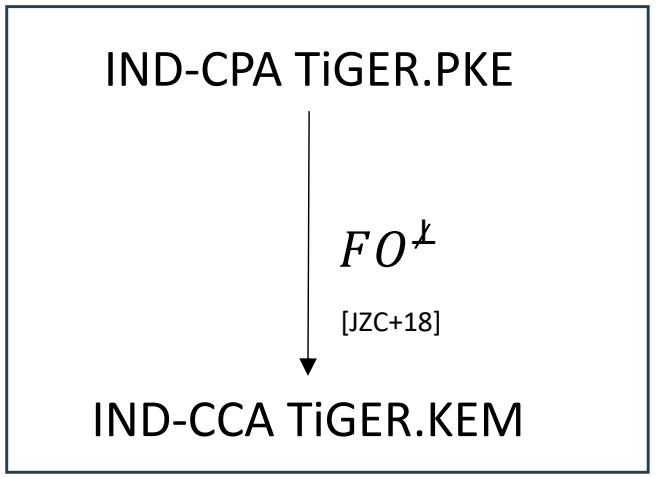
$$\boldsymbol{b} = \left\lfloor \frac{p}{q} \cdot \boldsymbol{a} * \boldsymbol{s} \right\rceil$$

CT (RLWE):

$$\boldsymbol{c_1} = \left\lfloor \frac{k_1}{q} \cdot \left(\boldsymbol{a} * \boldsymbol{r} + \boldsymbol{e_1} \right) \right\rceil, \quad c_2 = \left\lfloor \frac{k_2}{q} \cdot \left(\frac{q}{p} \cdot \boldsymbol{b} * \boldsymbol{r} + \boldsymbol{e_2} + \frac{q}{2} \cdot \mu_{ecc} \right) \right\rceil$$

Dec : $\mu'_{ecc} = \lfloor 2/k_2 \cdot \boldsymbol{c_2} - 2/q \cdot (q/k_1 \cdot \boldsymbol{c_1}) * \boldsymbol{s} \rceil$

❖ IND-CCA KEM scheme



◆ Key Generation

1. $(pk, sk') \leftarrow PKE.KeyGen(1^{\lambda})$
2. $\mathbf{u} \leftarrow R_2$
3. **return** $pk, sk = (sk', u)$

◆ Encapsulation

$pk\ hashing$: prevents multi-user attack

1. $\mu \leftarrow \{0,1\}^d$
2. $ct \leftarrow PKE.Enc(pk, \mu; H(\mu, H(pk)))$ — $H(\mu, H(pk))$: de-randomization
3. $K \leftarrow G(H(ct), \mu)$
4. **return** $c, t\ K$

◆ Decapsulation

1. $\mu' = PKE.Dec(sk', ct)$
2. $ct' = PKE.Enc(pk, \mu'; H(\mu', H(pk)))$ — re-encryption
3. if $ct = ct'$ then

   $K' \leftarrow G(H(ct), \mu')$
4. else $K' \leftarrow G(H(ct), u)$ — implicit rejection
5. **return** K'

[JZC+18] Haodong Jiang et al., "IND-CCA-Secure Key Encapsulation Mechanism in the Quantum Random Oracle Model, Revisited", CRYPTO 2018

❖ Decryption Failure Probability (DFP)

◆ Rounding Errors

- $\boldsymbol{u}_A = \frac{q}{p}\left\lfloor \frac{p}{q} \cdot \boldsymbol{a} * \boldsymbol{s} \right\rceil - \boldsymbol{a} * \boldsymbol{s}$ (public key rounding)
- $\boldsymbol{u}'_B = \frac{q}{k_1}\boldsymbol{c_1} - (\boldsymbol{a} * \boldsymbol{r} + \boldsymbol{e_1})$ (ciphertext compression)
- $\boldsymbol{u}''_B = \frac{q}{k_2}\boldsymbol{c_2} - \left(\left(\frac{q}{p} \cdot \boldsymbol{b}\right) * \boldsymbol{r} + \boldsymbol{e_2}\right)$ (ciphertext compression)

◆ In TiGER, DFP = Prob. of at least $f + 1$ bits errors

f : error correction capacity

- D2 Encoding + Xef **Error Correction Codes**
- (1 bit error prob.) $\delta = \Pr\left[||(-\boldsymbol{s} * (\boldsymbol{e_1} + \boldsymbol{u'_B}) + \boldsymbol{u_A} * \boldsymbol{r} + \boldsymbol{e_2} + \boldsymbol{u''_B})_i||_\infty > \frac{q}{2}\right]$

◆ Independency model for calculating DFP

- Assumes that bit errors are independent (in TiGER, too)

$$DFP = 1 - \Sigma_{i=0}^{f} \binom{n}{i} \cdot \delta^i \cdot (1-\delta)^{n-i}$$

- However, [DVV19] showed that bit errors are *positively correlated* due to ring structure

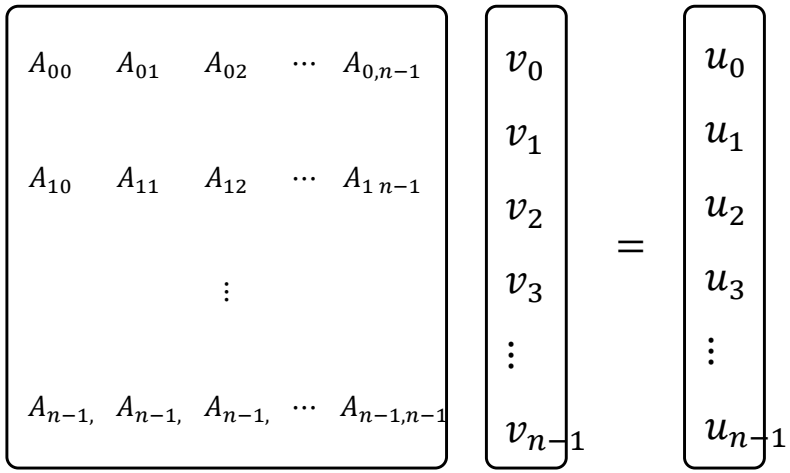
[DVV19] D'Anvers J-P., Vercauteren F. and Verbauwhede I., "The Impact of Error Dependencies on Ring/Mod-LWE/LWR Based Schemes", PQCrypto 2019

❖ Matrix-Vector Multiplication vs. Poly.-Poly. Multiplication

◆ Matrix Multiplication

- u_i and u_j are independent

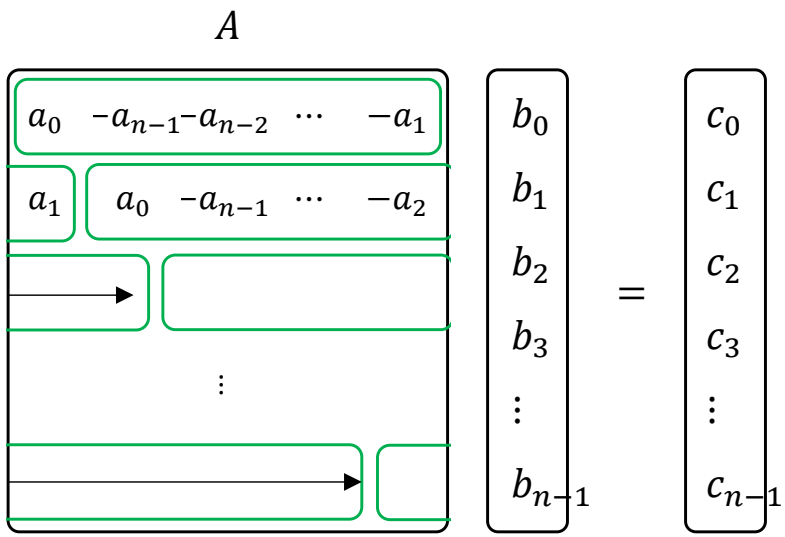
$$Av = u$$



◆ Polynomial Multiplication

- c_i and c_j are *dependent*
- This causes error dependencies

$$a * b = c$$
Polynomial Multiplication



About DFP of TiGER

❖ Problem of Independency Model

- Probability of multiple errors are larger in experimental data than in independency model
- Errors are positively correlated

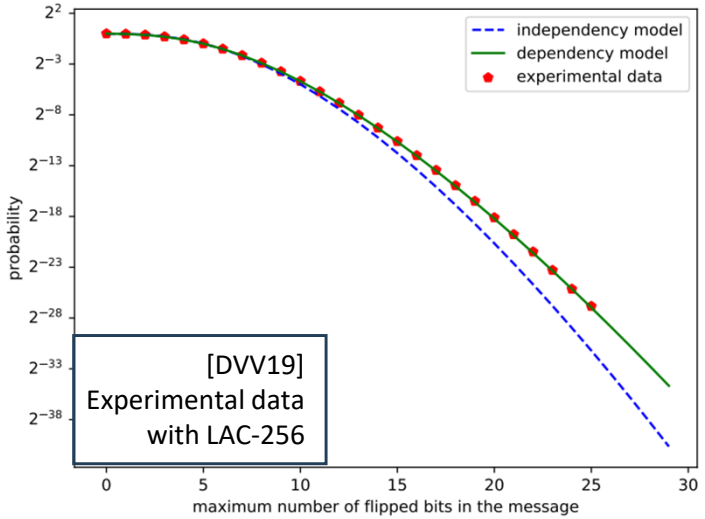


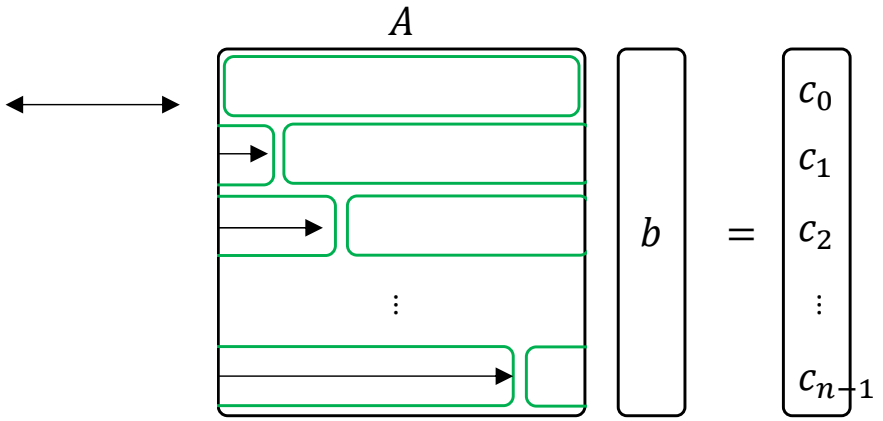
Figure 2: Probability of failure for various error correction capabilities of `ecc_enc`

❖ Dependency Model for DFP

◆ [DVV19]

- c_i is large
  ⇒ $\|a\|_2, \|b\|_2$ might be large
  ⇒ c_j might be large, too

$$a * b = c$$
Polynomial Multiplication



- Compute each bit error probability conditioned on norm of the errors

[DVV19] D'Anvers J-P., Vercauteren F. and Verbauwhede I., "The Impact of Error Dependencies on Ring/Mod-LWE/LWR Based Schemes", PQCrypto 2019

❖ [DVV19] DFP of LAC-v1

◆ LAC(proposed in NIST PQC)

- RLWE-based PKE scheme with error correction codes (BCH)

	LAC-128	LAC-256
Independency model	2^{-233}	2^{-114}
Dependency model	2^{-185}	2^{-92}
Overestimation factor	2^{48}	2^{22}

DFP increase

- Maximum overestimation factor of 2^{48}

◆ Can also be applied to TiGER (currently in progress)

[DVV19] D'Anvers J-P., Vercauteren F. and Verbauwhede I., "The Impact of Error Dependencies on Ring/Mod-LWE/LWR Based Schemes", PQCrypto 2019

A Variant of NTRU PKE (1)

❖ $R_q = \mathbb{Z}_q[x]/< x^n - 1 >,\ \ p = 3$

- **Public key :** $\boldsymbol{h} = p\,(\boldsymbol{g} \cdot \boldsymbol{f}^{-1}) \in R_q$ **Secret key :** $\boldsymbol{f}$
  - $\boldsymbol{f} = 3\boldsymbol{f}' + 1$, where $\boldsymbol{f}'$ chosen in R_p
- **Ciphertext :** $\boldsymbol{c} = \boldsymbol{r} \cdot \boldsymbol{h} + \boldsymbol{m} \in R_q$

$$f = 3f' + 1$$

PK : $\boldsymbol{h} = \lceil p\,\boldsymbol{g} \cdot \boldsymbol{f}^{-1} \rfloor_{R_q}$ SK : $\boldsymbol{f}$

CT : $\boldsymbol{c} = \lceil \boldsymbol{r} \cdot \boldsymbol{h} + \boldsymbol{m} \rfloor_{R_q}$

- Coefficients of f' are chosen from $D_{f'}$

A Variant of NTRU PKE (2)

❖ $R_q = \mathbb{Z}_q[x]/< x^n - 1 >,\ \ p = 3$

- **Public key :** $\boldsymbol{h} = p\,(\boldsymbol{g} \cdot \boldsymbol{f}^{-1}) \in R_q$ **Secret key :** $\boldsymbol{f}$
  - $\boldsymbol{f} = 3\boldsymbol{f}' + 1$, where $\boldsymbol{f}'$ chosen in R_p
- **Ciphertext :** $\boldsymbol{c} = \boldsymbol{r} \cdot \boldsymbol{h} + \boldsymbol{m} \in R_q$
- **Decryption:** $\boldsymbol{m} = \boldsymbol{c} \cdot \boldsymbol{f}\ (in\ R_q)$ **(mod** p**)**

$f = 3f' + 1$

$$\boldsymbol{m} = \lfloor \boldsymbol{c}\ \boldsymbol{f} \rfloor_{R_q} = \lfloor \boldsymbol{r}\ p\ \boldsymbol{g}\ \boldsymbol{f}^{-1}\ \boldsymbol{f} + \boldsymbol{m}\ \boldsymbol{f} \rfloor_{R_q} = \lfloor p\ \boldsymbol{r}\ \boldsymbol{g} + \boldsymbol{m}\ \boldsymbol{f} \rfloor_{R_q}$$

$$= \lfloor p\ \boldsymbol{r}\ \boldsymbol{g} + p\ \boldsymbol{m}\ \boldsymbol{f}' + \boldsymbol{m} \rfloor_{R_q}$$

$This\ part = 0\ \ (mod\ \ p)\ \ if\ \ |p(\boldsymbol{r} \cdot \boldsymbol{g} + \boldsymbol{m} \cdot \boldsymbol{f}') + \boldsymbol{m}|_\infty < q/2$

- Decryption works only when all coefficients of $p(\boldsymbol{r} \cdot \boldsymbol{g} + \boldsymbol{m} \cdot \boldsymbol{f}') + \boldsymbol{m} \in [-q/2,\ q/2]$
- Distributions D_f, D_g, D_r, D_m are important for correctness of NTRU

❖ When using FO-transform

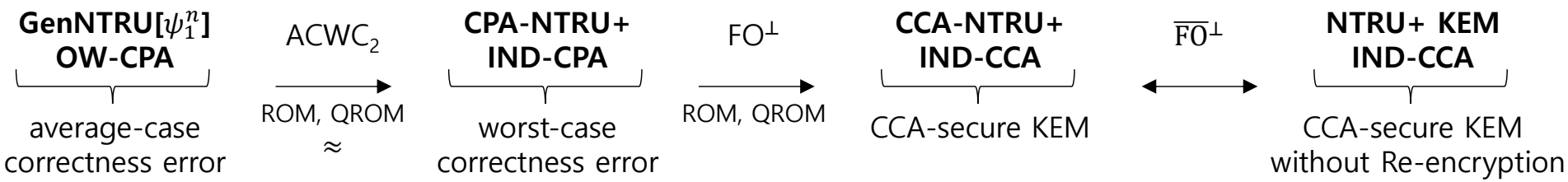
◆ Hard to control r, but adversary can control m

◆ Need to achieve negligible worst-case correctness error

- **Solution 1 – <u>perfect correctness error</u>**
  - All (m, r) tuples do not make NTRU decryption fail
  - Adopted by **Finalist NTRU**

- **Solution 2 - <u>worst-case correctness error ≈ average-case correctness error</u>**
  - Using an encoding method that forces m (as well as r) to be sampled honestly
  - Adopted by **NTRU+**

❖ Overview


GenNTRU[ψ_1^n]
OW-CPA
average-case
correctness error
ACWC$_2$
ROM, QROM
≈
CPA-NTRU+
IND-CPA
worst-case
correctness error
FO$^\perp$
ROM, QROM
CCA-NTRU+
IND-CCA
CCA-secure KEM
$\overline{\text{FO}}^\perp$
NTRU+ KEM
IND-CCA
CCA-secure KEM
without Re-encryption

❖ GenNTRU[ψ_1^n]

- **Gen**(1^λ)
  - $(pk, sk) = \mathbf{Gen}(1^\lambda)$
    - $\mathbf{f}', \mathbf{g} \leftarrow \boldsymbol{\psi_1^n}$
    - $\mathbf{f} = 3\mathbf{f}' + 1$
    - check if $\mathbf{f}$ and $\mathbf{g}$ are invertible
    - $\boldsymbol{pk} = \mathbf{h} = 3\mathbf{g}\mathbf{f}^{-1}$, $\boldsymbol{sk} = \mathbf{f}$

- **Enc**$(\boldsymbol{pk}, \mathbf{m} \leftarrow \boldsymbol{\psi_1^n}; \mathbf{r} \leftarrow \boldsymbol{\psi_1^n})$
  - $\mathbf{c} = \mathbf{h}\mathbf{r} + \mathbf{m}$

- **Dec**$(\mathbf{sk}, \mathbf{c})$
  - $\boldsymbol{m} = (\mathbf{c}\mathbf{f} \bmod q) \bmod^{\pm} 3$

- **Recover**$^r(\mathbf{h}, \mathbf{c}, \mathbf{m})$
  - $\mathbf{r} = (\mathbf{c} - \mathbf{m})\mathbf{h}^{-1}$

❖ CPA-NTRU+ from $ACWC_2$

◆ $\mathbf{Gen'}(1^\lambda)$

- $(pk, sk) = \mathbf{Gen}(1^\lambda)$
  - $\mathbf{f}', \mathbf{g} \leftarrow \boldsymbol{\psi}_1^n$
  - $\mathbf{f} = 3\mathbf{f}' + 1$
  - check if $\mathbf{f}$ and $\mathbf{g}$ are invertible
  - $(pk, sk) = (\mathbf{h} = 3\mathbf{g}\mathbf{f}^{-1}, \mathbf{f})$

◆ $\mathbf{Enc'}(pk, m; \mathbf{r} \leftarrow \psi_1^n)$

- $\mathbf{m} = \mathbf{SOTP}(m, \mathbf{G}(\mathbf{r}))$
  - $(u_0, u_1) = \mathbf{G}(\mathbf{r})$
  - $\mathbf{m} = (m \oplus u_0) - u_1$
- $\mathbf{c} = \mathbf{Enc}(\mathbf{h}, \mathbf{m}; \mathbf{r})$
  - $\mathbf{c} = \mathbf{h}\mathbf{r} + \mathbf{m}$

◆ $\mathbf{Dec'}(sk, \mathbf{c})$

- $\boldsymbol{m}' = \mathbf{Dec}(\mathbf{f}, \mathbf{c})$
  - $\boldsymbol{m}' = (\mathbf{c}\mathbf{f} \bmod q) \bmod^{\pm} 3$
- $\mathbf{r}' = \mathbf{Recover}^r(\mathbf{h}, \mathbf{c}, \mathbf{m}')$
  - $\mathbf{r}' = (\mathbf{c} - \mathbf{m}')\mathbf{h}^{-1}$
- $m = \mathbf{Inv}(\mathbf{m}', \mathbf{G}(\mathbf{r}'))$
  - $(u_0, u_1) = \mathbf{G}(\mathbf{r}')$
  - $m = (\mathbf{m}' + u_1) \oplus u_0$

❖ CCA-NTRU+ from $\mathbf{FO}^{\perp}$

◆ KeyGen(1^{λ})

- $(pk, sk) = \mathbf{Gen}(1^{\lambda})$
  - $\mathbf{f}', \mathbf{g} \leftarrow \boldsymbol{\psi}_1^n$
  - $\mathbf{f} = 3\mathbf{f}' + 1$
  - check if $\mathbf{f}$ and $\mathbf{g}$ are invertible
  - $(pk, sk) = (\mathbf{h} = 3\mathbf{g}\mathbf{f}^{-1}, \mathbf{f})$

◆ Encap(pk)

- $m \leftarrow \{0,1\}^n$
- $(K, \mathbf{r}) = \mathbf{H}(m)$
- $\mathbf{c} = \mathbf{Enc}'(pk, m; \mathbf{r})$
  - $\mathbf{m} = \mathbf{SOTP}(m, \mathbf{G}(\mathbf{r}))$
  - $\mathbf{c} = \mathbf{Enc}(\mathbf{h}, \mathbf{m}; \mathbf{r})$

◆ Decap$(sk, \mathbf{c})$

- $m' = \mathbf{Dec}'(sk, \mathbf{c})$
  - $\mathbf{m}' = \mathbf{Dec}(\mathbf{f}, \mathbf{c})$
  - $\mathbf{r}' = \mathbf{Recover}^{\mathbf{r}}(\mathbf{h}, \mathbf{c}, \mathbf{m}')$
  - $m' = \mathbf{Inv}(\mathbf{m}', \mathbf{G}(\mathbf{r}'))$
- $(K', \mathbf{r}'') = \mathbf{H}(m')$
- If $\boldsymbol{c} = \mathbf{Enc}'(pk, m'; \mathbf{r}'')$
  - Return K'
  - Else, return $\perp$

❖ CCA-NTRU+ from $\overline{\mathbf{FO}}^{\perp}$

◆ KeyGen(1^{λ})

- $(pk, sk) = \mathbf{Gen}(1^{\lambda})$
  - $\mathbf{f}', \mathbf{g} \leftarrow \boldsymbol{\psi}_1^n$
  - $\mathbf{f} = 3\mathbf{f}' + 1$
  - check if $\mathbf{f}$ and $\mathbf{g}$ are invertible
  - $(pk, sk) = (\mathbf{h} = 3\mathbf{g}\mathbf{f}^{-1}, \mathbf{f})$

◆ Encap(pk)

- $m \leftarrow \{0,1\}^n$
- $(K, \mathbf{r}) = \mathbf{H}(m)$
- $\mathbf{c} = \mathbf{Enc}'(pk, m; \mathbf{r})$
  - $\mathbf{m} = \mathbf{SOTP}(m, \mathbf{G}(\mathbf{r}))$
  - $\mathbf{c} = \mathbf{Enc}(\mathbf{h}, \mathbf{m}; \mathbf{r})$

◆ Decap$(sk, \mathbf{c})$

- $m' = \mathbf{Dec}'(sk, \mathbf{c})$
  - $\mathbf{m}' = \mathbf{Dec}(\mathbf{f}, \mathbf{c})$
  - $\mathbf{r}' = \mathbf{Recover}^{\mathbf{r}}(\mathbf{h}, \mathbf{c}, \mathbf{m}')$
  - $m' = \mathbf{Inv}(\mathbf{m}', \mathbf{G}(\mathbf{r}'))$
- $(K', \mathbf{r}'') = \mathbf{H}(m')$
- If $\mathbf{r}' == \mathbf{r}''$
  - Return K'
  - Else, return $\perp$

❖ [Lee23] Attack Against NTRU+

- $M = \mathbf{SOTP}(m, \mathbf{G}(\mathbf{r}) = (u_1, u_2))$
  - $M = (m \oplus u_1) - u_2$
- $m = \mathbf{Inv}(M, \mathbf{G}(\mathbf{r}) = (u_1, u_2))$
  - $m = (M + u_2) \oplus u_1$

- **Inv**$(y, \boldsymbol{u} = (u_0, u_1))$
  - $t = y + u_1$
  - If $t \notin \{0,1\}^n$, **return** $\perp$.
  - Else, $m = t \oplus u_0$
  - **return** $m \in \{0,1\}^n$

- $\mathbf{c} = \mathbf{Enc}(\mathbf{h}, \mathbf{SOTP}(m, \mathbf{G}(\mathbf{r})); \mathbf{r})$
  - $M = \mathbf{SOTP}(m, \mathbf{G}(\mathbf{r}))$
  - $[M]_1 = -1, [u_2]_1 = 1$
  - $[M]_1 \oplus [u_2]_1 = -1 + 1 = 0$
- $\mathbf{c}' = \mathbf{Enc}(\mathbf{h}, \mathbf{SOTP}(m, \mathbf{G}(\mathbf{r})); \mathbf{r}) + (2,0,\cdots,0)$
  - $M' = \mathbf{SOTP}(m, \mathbf{G}(\mathbf{r})) + (2,0,\cdots,0)$
  - $[M']_1 = 1, [u_2]_1 = 1$
  - $[M']_1 \oplus [u_2]_1 = 1 + 1 = 2 \equiv 0 \pmod 2$

Same value

➔ Recover same m

➔ Compute same $(K', \mathbf{r}') = \mathbf{H}(m)$

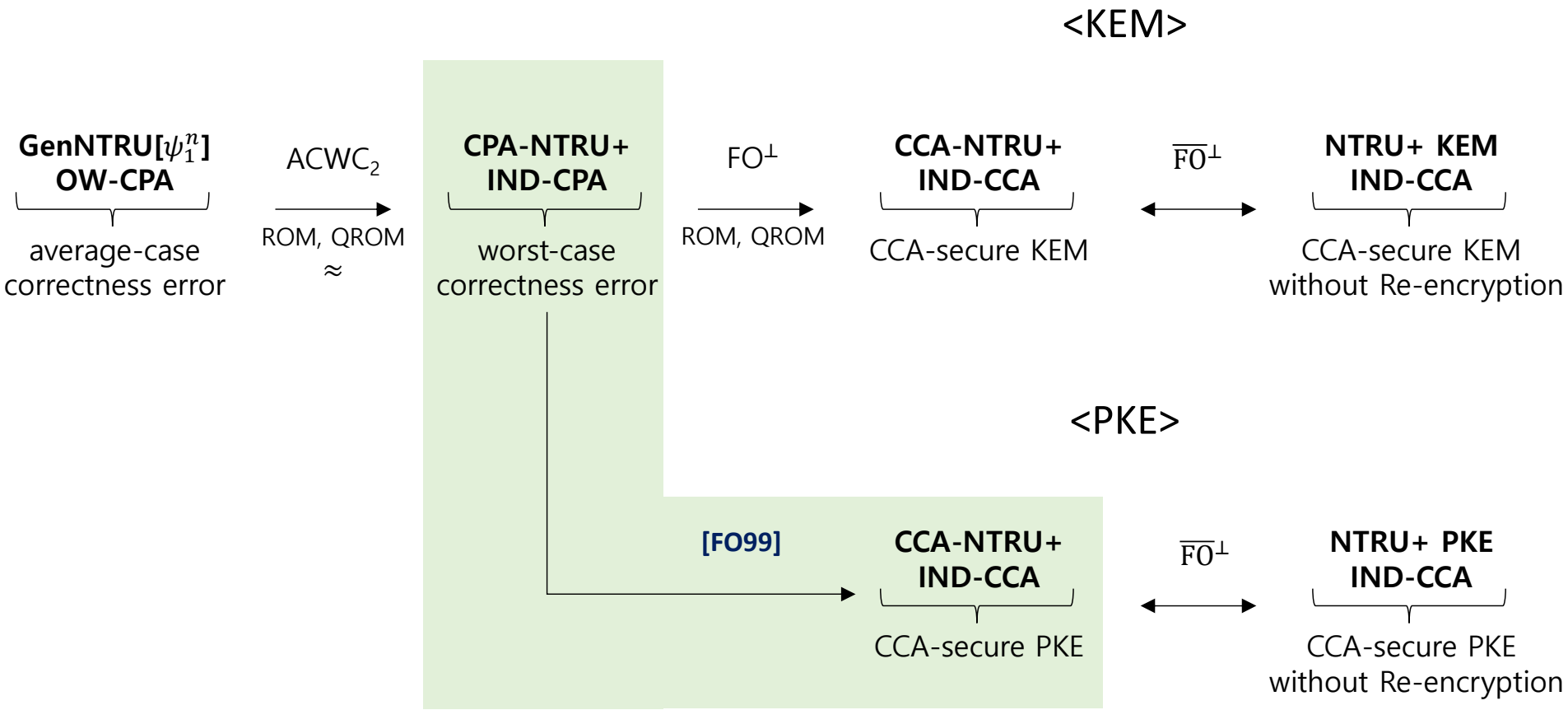
➔ Pass validity check $\mathbf{r} == \mathbf{r}'$

➔ $\mathbf{Decap}(sk, \mathbf{c}) = \mathbf{Decap}(sk, \mathbf{c}') \rightarrow K = K'$

< CCA against NTRU+ >

[Lee23] https://groups.google.com/g/kpqc-bulletin/c/cxnTcy1oJK8

❖ IND-CCA secure PKE from NTRU+



[FO99] Fujisaki et al. “How to Enhance the Security of Public-Key Encryption at Minimum Cost“, PKC’99

❖ IND-CCA secure PKE from IND-CPA secure PKE [FO99]

$\mathbf{PKE}' = (\mathbf{Gen}', \mathbf{Enc}', \mathbf{Dec}')$ → $\mathbf{PKE}'' = (\mathbf{Gen}'', \mathbf{Enc}'', \mathbf{Dec}'')$

IND-CPA → IND-CCA

◆ **Enc"**$(pk, m; r)$

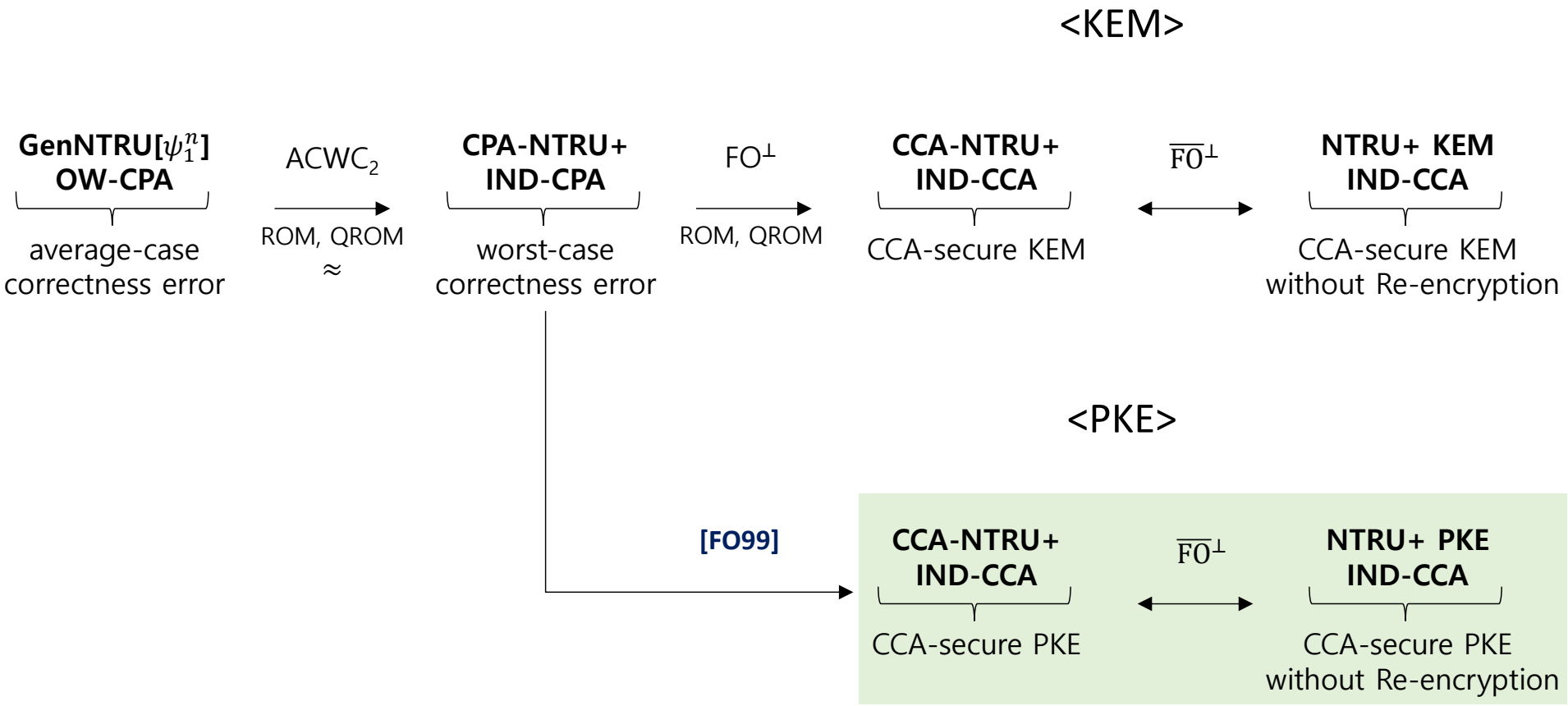
- $\boldsymbol{c} = \mathbf{Enc}'\left(pk, m||r; \mathbf{H}(m||r)\right)$
  - $\boldsymbol{m} = \mathbf{SOTP}\left(m||r, \mathbf{G}\left(\boldsymbol{r} = \mathbf{H}(m||r)\right)\right)$
  - $\boldsymbol{c} = \mathbf{Enc}(\boldsymbol{h}, \boldsymbol{m}; \boldsymbol{r})$
- **return** $\boldsymbol{c}$

◆ **Dec"**(sk, c)

- $m'||r' = \mathbf{Dec}'(sk, c)$
  - $\boldsymbol{m}' = \mathbf{Dec}(sk, c)$
  - $\boldsymbol{r}' = \mathbf{RRec}(\boldsymbol{h}, \boldsymbol{c}, \boldsymbol{m}')$
  - $m'||r' = \mathbf{Inv}\left(\boldsymbol{m}', \mathbf{G}(\boldsymbol{r}')\right)$
- $c' = \mathbf{Enc}'\left(pk, m'||r'; \mathbf{H}(m'||r')\right)$
- If $c = c'$, return m'. Else, return $\perp$.

[FO99] Fujisaki et al. "How to Enhance the Security of Public-Key Encryption at Minimum Cost", PKC'99

❖ IND-CCA secure PKE from NTRU+



[FO99] Fujisaki et al. “How to Enhance the Security of Public-Key Encryption at Minimum Cost“, PKC’99

❖ IND-CCA secure PKE′ from IND-CPA secure PKE [FO99]

$\mathbf{PKE}' = (\mathbf{Gen}', \mathbf{Enc}', \mathbf{Dec}')$ → $\mathbf{PKE}'' = (\mathbf{Gen}'', \mathbf{Enc}'', \mathbf{Dec}'')$

IND-CPA → IND-CCA

◆ **Enc"**$(pk, m; r)$

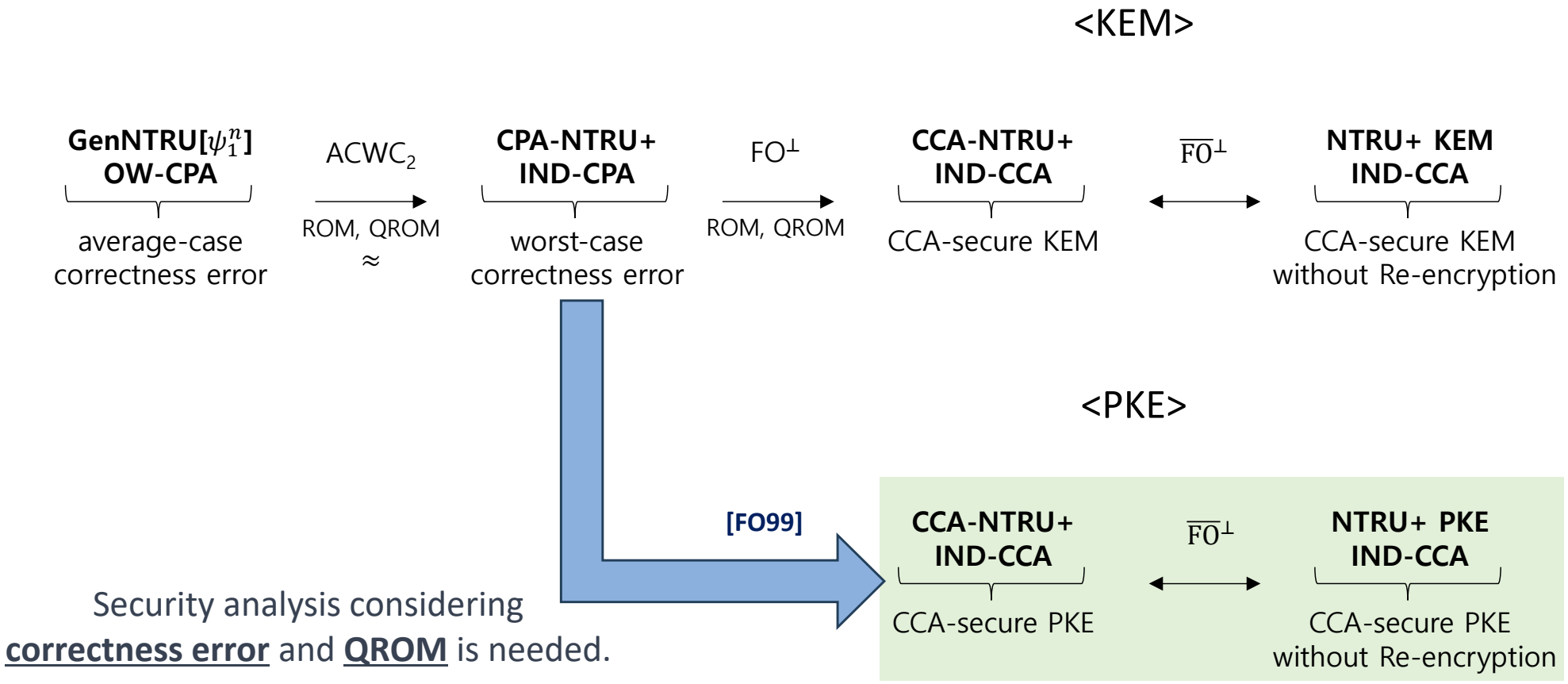
- $\boldsymbol{c} = \mathbf{Enc}'\left(pk, m||r; \mathbf{H}(m||r)\right)$
  - $\boldsymbol{m} = \mathbf{SOTP}\left(m||r, \mathbf{G}\left(\boldsymbol{r} = \mathbf{H}(m||r)\right)\right)$
  - $\boldsymbol{c} = \mathbf{Enc}(\boldsymbol{h}, \boldsymbol{m}; \boldsymbol{r})$
- return $\boldsymbol{c}$

◆ **Dec"**(sk, c)

- $m'||r' = \mathbf{Dec}'(sk, c)$
  - $\boldsymbol{m}' = \mathbf{Dec}(sk, c)$
  - $\boldsymbol{r}' = \mathbf{RRec}(\boldsymbol{h}, \boldsymbol{c}, \boldsymbol{m}')$
  - $m'||r' = \mathbf{Inv}\left(\boldsymbol{m}', \mathbf{G}(\boldsymbol{r}')\right)$
- ~~$c' = \mathbf{Enc}'\left(pk, m'||r'; \mathbf{H}(m'||r')\right)$~~
- ~~If $c = c'$, return m'. Else, return $\perp$.~~
- If $\mathbf{H}(m'||r') = \boldsymbol{r}'$, return m. Else, return $\perp$

[FO99] Fujisaki et al. "How to Enhance the Security of Public-Key Encryption at Minimum Cost", PKC'99

❖ IND-CCA secure PKE from NTRU+



[FO99] Fujisaki et al. “How to Enhance the Security of Public-Key Encryption at Minimum Cost“, PKC’99

비교

Security Level		Parameter Sets	n	k	q	pk (Bytes)	ct (Bytes)	sk (Bytes)	pk + ct (Bytes)	Log $_2\delta$
I	118	KYBER512	256	2	3329	800	768	1,632	1,568	139
	120	SMAUG128	256	2	1024	672	672	176	1,344	120
	130	TiGER128	512	-	256	480	640	528	1,120	128*
	115	NTRU+576	576	-	3457	864	864	1,728	1,728	487
	161	NTRU+768	768	-	3457	1152	1152	2304	2,304	379
III	182	KYBER768	256	3	3329	1,184	1,088	2,400	2,272	164
	180	SMAUG192	256	3	2048	1,088	1,024	236	2,112	136
	200	TiGER192	1024	-	256	928	1,024	528	1,952	154*
	188	NTRU+864	864	-	3457	1,296	1,296	2,592	2,592	340
V	256	KYBER1024	256	4	3329	1,568	1,568	3,168	3,136	174
	260	SMAUG256	256	5	2048	1,792	1,472	218	3,264	167
	263	TiGER256	1024	-	256	928	1,152	1,056	2,080	200*
	264	NTRU+1152	1152	-	3457	1,728	1,728	3,456	3,456	260

분석 결과 요약

❖ SMAUG

- **Module LWE** 및 **Module LWR** 기반 안전성 증명
- **FO 변환**을 통해 IND-CCA에 안전한 KEM으로 변환함
- 안전성 비도 레벨에 맞도록 **복호화 실패율을 더 낮출 필요가 있음**

❖ TiGER

- **Module LWR** 및 **Module LWER** 기반 안전성 증명
- **FO 변환**을 통해 IND-CCA에 안전한 KEM으로 변환함
- **ECC**(오류정정코드)를 기반으로 설계되어 **비트 간 복호화 오류 의존성**으로 인해 **복호화 실패율이 상승**한다는 문제점을 가짐[DVV19] → 명확한 추가 분석 필요

❖ NTRU+

- **NTRU** 및 **LWE** 기반 안전성 증명
- **ACWC2 변환 후 FO 변환**을 통해 IND-CCA에 안전한 KEM으로 변환함
- [Lee23]의 공격에 안전하도록 **SOTP 알고리즘**이 수정된 spec 및 코드가 필요함
- **Muti-user setting**에도 안전하도록 기법의 수정이 필요함
- **NTRU+PKE** 기법으로의 확장을 추가 제출하는 것도 필요함

Q&A