코드기반 KpqC 공모전 알고리즘 안전성 분석

Jihoon Hong

July. 14, 2023

Department of Mathematics
Sogang University, S. Korea

김종락 교수 연구실

KpqC 코드 기반 암호 알고리즘

1. Enhanced pqsigRM

proposed by 노종선, 조진규, 이용우, 구자현, 김영식

■ CFS(Courtois, Finiasz, and Sendrier) 서명 기법 - Goppa 코드 기반 서명 기법.

- 가장 저명한 부호 기반 전자서명 시스템
- 작은 Hamming 무게를 갖는 오류를 찾을 때 까지 반복해서 복호화 진행
- 초기 모델에는 high rate Goppa 부호 사용 - 단점 존재

개선 방안 : CFS 서명 기법에서 Goppa 부호를 Reed-Muller 부호로 대체하여 단점 개선.

■ Reed-Muller 부호

- complete decoding이 가능함 (recursive decoding)
- 다양한 정의 방법이 존재 $\rightarrow$ recursive decoding을 위해 recursive definition을 채택

■ Original pqsigRM & Enhanced pqsigRM

Table 1: Original pqsigRM 과 Enhanced pqsigRM 비교

	Original pqsigRM	Enhanced pqsigRM
키 생성	열 puncturing 및 추가	부분적 permutation, 행 추가 및 대체
복호화 과정	랜덤화 되지 않음	랜덤화됨
공격법	Hull을 통해 puncturing 찾는 공격	없음

Enhanced pqsigRM은 Reed-Muller 대신 modified Reed-Muller 부호를 사용하여 안정성을 높임.

■ 기존 Reed-Muller 부호와 modified Reed-Muller 부호의 차이

- 기존의 Reed-Muller 부호와 부분적으로 permute된 Reed-Muller 부호

σ_p^1, σ_p^2 : permutation

$G_{(r,m-2)}$	$G_{(r,m-2)}$	$G_{(r,m-2)}$	$G_{(r,m-2)}$
0	$G_{(r-1,m-2)}$	0	$G_{(r-1,m-2)}$
0	0	$G_{(r-1,m-2)}$	$G_{(r-1,m-2)}$
0	0	0	$G_{(r-2,m-2)}$

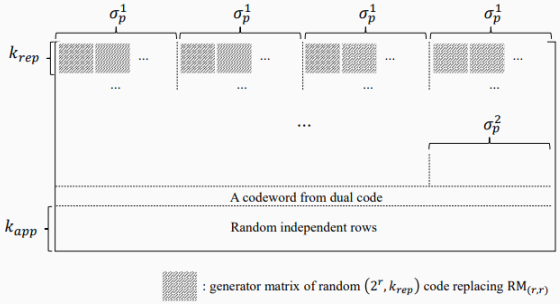
$\longrightarrow$

$G^{\sigma_p^1}_{(r,m-2)}$	$G^{\sigma_p^1}_{(r,m-2)}$	$G^{\sigma_p^1}_{(r,m-2)}$	$G^{\sigma_p^1}_{(r,m-2)}$
$G^{\sigma_p^1}_{(r,m-2)}$	$G^{\sigma_p^1}_{(r,m-2)}$	$G^{\sigma_p^1}_{(r,m-2)}$	$G^{\sigma_p^1}_{(r,m-2)}$
0	0	$G_{(r-1,m-2)}$	$G_{(r-1,m-2)}$
0	0	0	$G^{\sigma_p^2}_{(r-2,m-2)}$

- H'으로부터 H를 만들어내는 키 복구 공격 방지.
- Hull의 dimension을 크게 설정해서 hull을 이용하는 공격 방지.

1. Enhanced pqsigRM - modified Reed-Muller 부호

- 부분적으로 permute된 Reed-Muller 부호에 세 가지 변형 추가
  - $RM_(r,r)$ 들을 2^{m-r}개의 반복되는 랜덤한 $(2^r, k_{rep})$ 부호로 대체
  - k_{app} 만큼의 랜덤한 독립 행들 추가
  - Dual 부호에서 랜덤한 부호어 한 줄 추
- 위의 변형을 진행하면 다음과 같은 modified Reed-Muller 부호를 얻음.



1. Enhanced pqsigRM - modified Reed-Muller 부호

- Modified Reed-Muller 부호의 복호화는 다음과 같이 진행됨

function DECODE($\mathbf{s}; \mathbf{H}$)
  $\mathbf{r} \leftarrow$ PRANGE($\mathbf{H}, \mathbf{s}$)
  while True **do**
    $\mathbf{r} \leftarrow \mathbf{r}+$ random codeword
    $\mathbf{c} \leftarrow$ MODDEC($\mathbf{r}, r, M$)
    if $\mathrm{wt}(\mathbf{r}+\mathbf{c}) \leq w$ **then**
      Output $\mathbf{r}+\mathbf{c}$
    end if
  end while
end function

function MODDEC($\mathbf{y}, r, M$)
  $\mathbf{y} \leftarrow \mathbf{y}^{\sigma^{-1}}$
  if $r = 0$ **then**
    Output MD decoding on RM$(0, m)$
  else if $r = m$ **then**
    Output MD decoding on RM(r, r)
     or replaced $(2^r, k_{rep})$ code
  else
    $(\mathbf{y}'|\mathbf{y}'') \leftarrow \mathbf{y}$
    $\mathbf{y}^{\mathbf{v}} = \mathbf{y}' \cdot \mathbf{y}''$
    $\hat{\mathbf{v}} \leftarrow$ MODDEC($\mathbf{y}^{\mathbf{v}}, r-1, m-1$)
    $\mathbf{y}^{\mathbf{u}} \leftarrow (\mathbf{y}' + \mathbf{y}'' \cdot \hat{\mathbf{v}})/2$
    $\hat{\mathbf{u}} \leftarrow$ MODDEC($\mathbf{y}^{\mathbf{u}}, r, m-1$)
    $\mathbf{y} \leftarrow (\hat{\mathbf{u}}|\hat{\mathbf{u}} \cdot \hat{\mathbf{v}})$
  end if
  Output $\mathbf{y}^{\sigma}$
end function
$^*\sigma$ is σ_p^1 or σ_p^2 for permuted block and identity, otherwise.

1. Enhanced pqsigRM - Signature Scheme

■ Enhanced pqsigRM의 서명 기법은 다음과 같음

Key Generation :

$\mathbf{G}_{\mathcal{M}}$: $k \times n$ generator matrix of modified RM codes
$\mathbf{H}_{\mathcal{M}}$: $(n-k) \times n$ parity check matrix of modified RM codes
$\mathbf{S} \xleftarrow{\$} F_2^{(n-k)\times(n-k)}, \mathbf{Q} \xleftarrow{\$} F_2^{n\times n}$
$\mathbf{H}' \leftarrow \mathbf{S}\mathbf{H}_{\mathcal{M}}\mathbf{Q}$
$\mathbf{H}'_{\mathbf{sys}} = (\mathbf{I}|\mathbf{T})$: systematic form of $\mathbf{H}'$
Public key: $\mathbf{T}$
Secret key: $\mathbf{Q}$, σ_p^1, σ_p^2, $k_{rep} \times 2^r$ (repeated) replacing codes, $k_{app} \times n$ appending codes, and $1 \times n$ padding dual code codeword

Signing :

M: Message, $i \hookleftarrow \{0,1\}^{\lambda_0}$: Counter
$\mathbf{s} \leftarrow h(M|i)$: Syndrome
$\mathbf{s}'^T \leftarrow \mathbf{S}^{-1}\mathbf{s}^T$
$\mathbf{e}' \leftarrow \textsc{Decode}(\mathbf{s}'; \mathbf{H}_{\mathcal{M}})$
$\mathbf{e}^T \leftarrow \mathbf{Q}^{-1}\mathbf{e}'^T$
Signature: $(M, \mathbf{e}, \mathbf{i})$

Verification :

If $wt(e) \leq w$ and $H'_{sys}e^T = h(M|i)$,
return ACCEPT
Else, return REJECT

*h: hash function SHAKE-128/256
*DECODE: Decoding algorithm of modified RM codes
*$wt(a)$: Hamming weight of a vector a
*w: error correcting capability of modified RM codes

1. Enhanced pqsigRM - Security Analysis

- Enhanced pqsigRM 서명 기법
  - CFS 전자 서명 방식에 Goppa 부호 대신 modified Reed-Muller 부호의 패리티 체크 행렬을 사용함
  - 신드롬 복호화 문제의 변형인 DOOM(Decoding out of many) 문제에 기반함
  - 신드롬에 hasing을 두번 진행하여 키 교환 공격으로부터 안전함
  - 공개키가 랜덤 시퀀스와 구별 불가능함을 확인함
  - 공개키가 구별 불가능하다는 가정 하에 EUF-CMA 안전성이 증명됨
- 다음과 같은 공격들에 대해 안전하다고 증명됨.
  - Minder-Shokrollahi의 공격
  - Chizhov-Borodin의 공격
  - Square Code 공격
  - Hull을 이용한 punctured & inserted element를 찾는 공격
  - 서명의 각 element들에 대한 1의 확률을 이용한 공격
  - 최근접 최소 무게 부호어의 각 element에 대한 1의 확률 공격

- NIST PQC 경연에 3라운드에 진출한 서명 알고리즘들과의 비교

Table 2: 공개키 크기 비교 (Bytes)

security	Enhanced pqsigRM	CRYSTALS-DILITHIUM	FALCON	SPHINCS+
128	$474,445$	$1,312$	897	32
256	$2,000,000$	$2,592$	$1,793$	64

Table 3: 서명 크기 비교 (Bytes)

security	Enhanced pqsigRM	CRYSTALS-DILITHIUM	FALCON	SPHINCS+
128	512	$2,420$	666	$7,856$
256	$1,024$	$4,595$	$1,280$	$29,792$

- 부호 기반 전자 서명 중에는 공개키 크기와 서명 길이가 가장 작음
- 다른 서명들과 비교하였을 때에는 서명 길이는 가장 작지만 공개키 사이즈가 매우 큼

2. Layered ROLLO-1

proposed by 김찬기, 김영식, 노종선

■ 랭크 무게와 랭크 거리 부호

- 랭크 무게
  - 각 원소가 m차원인 길이가 n인 벡터 $v \in (\mathbb{F}_{q^m})^n$의 랭크 무게 $\left||v|\right|$는 각 원소를 $m \times 1$ 벡터에 대응한 크기로 대응한 $m \times n$ 행렬 $M(v)$에 대한 랭크 값

- 랭크 기반 부호 (최소 랭크 무게가 d인 $(n,k)\mathbb{F}_{q^m}$ 선형 부호 $\mathcal{C}$)
  - 길이가 n인 부호어 $\mathbf{c} \in \mathcal{C}$의 랭크 무게 $\left||v|\right| \geq d$
  - $(n-k) \times n$ 크기의 패리티 검사 행렬 H에 대해 $Hc^\top = 0$ 만족

Layered ROLLO-1에서는 Ideal LRPC(low-rank parity-check) 부호를 사용함. 이 부호의 패리티 검사 행렬 H는 다음과 같이 정의됨.

$$H = (H_1|H_2)$$

■ ROLLO의 주요 아이디어

- Ideal LRPC부호의 구분불가능성
  부호어 $c = [ab]$의 길이가 n인 벡터 a, b에 대해 $g = a^{-1}b$를 다음과 같이 정의하면 구분불가능성을 확보함.
  - $H \times c^{\top} = [H_1 H_2][a, b]^{\top} = 0^{\top}$가 만족하는 경우 $c = [xgx]$인 벡터 생성자로서, polynomial ring 연산인 $g \rightarrow G(X) = \frac{B[X]}{A[X]}(mod x^n - 1)$을 통해 계산 가능
  - 구분 불가능한 $g = a^{-1}b$를 공개키로 활용하여 전송

■ ROLLO-1 암호 알고리즘

- ROLLO-1은 다음과 같은 3개의 phase로 진행됨.

1. *Key generation: Select two n-tuple vectors* $\boldsymbol{x}, \boldsymbol{y} \in F$*, where* F *denotes a set of n-tuple vectors with rank weight* d*. Then, Alice construct secret key (SK) as* $(\boldsymbol{x}, \boldsymbol{y})$ *and public key (PK) as* $\boldsymbol{h} = \boldsymbol{x}^{-1}\boldsymbol{y} \mod P$*.*
2. *Encryption: Bob select two vector* $(\boldsymbol{e}_1, \boldsymbol{e}_2) \in E$*, where* E *denotes a set of an n-tuple vector with rank weight* r*. Then, derive* $\boldsymbol{c} = \boldsymbol{e}_1 + \boldsymbol{e}_2\boldsymbol{h} mod P$ *using PK* $\boldsymbol{h}$*, where* $Hash(\cdot)$ *denotes a hash function known to Alice and Bob. Then, derive* $K = Hash(E)$*. Finally, send* $\boldsymbol{c}$ *to Alice and use* $\boldsymbol{k}_1 = Hash(E)$ *as a shared secret (SS).*
3. *Decryption: Using SK* $\boldsymbol{x}$*, derive* $\boldsymbol{s} = \boldsymbol{x}\boldsymbol{c} = \boldsymbol{x}\boldsymbol{e}_1 + \boldsymbol{y}\boldsymbol{e}_2 \mod P$ *and thus, we can find the basis* E *with the ideal RSR algorithm. Using the basis* E*, derive* $Hash(E)$ *for a hash function. Finally, the remaining procedure are divided as follows. Finally, Alice validate the correctness by checking the SS* $\boldsymbol{k}_2 = Hash(E)$*. Then, Alice can validate the correctness by checking* $\boldsymbol{k}_1 = \boldsymbol{k}_2$*.*

■ BII(Block-wise interleaved ideal)-LRPC 부호를 설계하여 고속 동작하는 Layered ROLLO-1 암호화 방법 제안

- BII-LRPC 부호
  - $\mathbb{F}_{q^m}$ 안에 있는 $\mathbb{F}$-subspace인 F에서 n/b 차수 다항식 P 가정
  - F를 support로 하는 무작위한 $2b$개의 벡터 $x_1, ..., x_b$ 및 $y_1, ..., y_b$를 가정
  - 임의의 Permutation 행렬 P에 대해 BII-LRPC 부호를 $H = [PH_1 PH_2]$로 정의

BII-LRPC 부호에 기반하여 ROLLO-1을 개선한 Layered ROLLO-1 암호 알고리즘 제안

1. Key generation: Let F be a set of $\frac{n}{b}$-tuple vector with rank weight d. Alice selects two $\frac{n}{b}$-tuple random vectors $\mathbf{x}, \mathbf{y}$ satisfying $\mathbf{x}, \mathbf{y} \in F$. Also, denote random degree-$(b-1)$ primitive polynomial $P_I \in \mathbb{F}_{q^m}[X]/ < P >$ and a degree-n $P_O, P_N \in \mathbb{F}_{q^m}[X]/ < P^b >$, For $\mathbf{x}$ and $\mathbf{y}$, derive an $\frac{n}{b}$-tuple vector

$$\mathbf{z} = P_I \mathbf{x}^{-1} \mathbf{y} \mod P.$$

Finally, Alice constructs a public key (PK) as

$$\begin{aligned} P_P &= P_O P_I && \mod P^b, \\ \mathbf{h} &= P_O \mathbf{z}' + P_N P && \mod P^b, \end{aligned}$$

for $\mathbf{z}' = [\mathbf{0}, \mathbf{z}]$ and secret key (SK) as $\mathbf{x}, \mathbf{y}, P_I$, and P_O.

2. Encapsulation: Bob selects random two $\frac{n}{b}$-length vectors $(\mathbf{e}_1, \mathbf{e}_2) \in E$, where E denotes a set of $\frac{n}{b}$-tuple vector with maximum degree $\frac{n}{b} - b$ and rank weight r. Also, let length-n vectors $\mathbf{e}'_1 = [\mathbf{0}, \mathbf{e}_1]$ annd $\mathbf{e}'_2 = [\mathbf{0}, \mathbf{e}_2]$. Then, a ciphertext is generated by

$$\mathbf{c} = P_P \mathbf{e}'_1 + \mathbf{h}\mathbf{e}'_2 \mod P^b$$

using PK of P_P and $\mathbf{h}$. For a hash function $\mathrm{Hash}(\cdot)$ known to Alice and Bob, $\mathbf{k}_1 = \mathrm{Hash}(E)$ can be calculated. Finally, send $\mathbf{c}$ to Alice and use $\mathbf{k}_1$ as a shared secret (SS).

3. Decapsulation: Using SK of $\mathbf{x}$ and P_P, we have

$$\begin{aligned} \mathbf{c}' &= P_O^{-1}\mathbf{c} \mod P^b, \\ \mathbf{c}'' &= P_I^{-1}\{\mathbf{c}' \mod P\} \mod P, \\ \mathbf{x}\mathbf{c}'' &= \mathbf{x}\mathbf{e}_1 + \mathbf{y}\mathbf{e}_2 \mod P. \end{aligned}$$

Then, it is easy to check that $\mathbf{x}\mathbf{c}''$ can be decoded using the RSR algorithm, which can reconstruct E'. From $\mathbf{k}_2 = \mathrm{Hash}(E')$, Alice can validate the correctness by checking $\mathbf{k}_1 = \mathbf{k}_2$.

■ KpqC 1라운드에 제안한 Layered ROLLO-1에 대한 새로운 공격 제시

- Polynomial P_E 에 대한 공격이 진행되었음.
  - Key recovery instance
  - New combinatorial attack
  - New algebraic attack
  - Conventional Structural attack
  - Direct attack

결과적으로, 파라미터가 (q, n, m, r, d, b)인 Layered ROLLO-1은 파라미터가 $(q, n/b, m, r, d)$인 ROLLO-1로 변환될 수 있다는 것이 연구가 됨.

■ KpqC 1라운드에 제안한 Layered ROLLO-1에 대한 새로운 공격에 대한 대비

- Modified Layered ROLLO-1을 제시하여 key generation, encapsulation, decapsulation을 수정함.
  - Key generation : 두개의 modulus $P^{(1)}$와 $P^{(2)}$를 생성하여 키 생성 과정 변경
  - Encapsulation : 랭크 무게 r을 증가시키고 $P_{E,1}$와 $P_{E,2}$ 위에서 랜덤화된 다항식들을 추가
  - Decapsulation : 위에서 변경된 사항들을 반영하여 decapsulation 과정 진행

결과적으로, 파라미터가 (q, n, m, r, d, b)인 Layered ROLLO-1은 파라미터가 $(q, n/b, m, r, d)$인 ROLLO-1로 변환될 수 있다는 것이 연구가 됨.

■ Original ROLLO-1, Layered ROLLO-1, Modified Layered ROLLO-1 의 파라미터, 키 크기

Table 4: Original ROLLO-1과 Layered ROLLO-1 비교

Instances (security)	q	n	m	r	d	b	DFR	$\mathsf{PK}_{\mathsf{size}}$ (Bytes)	$\mathsf{SK}_{\mathsf{size}}$ (Bytes)	$\mathsf{CT}_{\mathsf{size}}$ (Bytes)
ROLLO-1 (128)	2	83	67	7	8	1	2^{-27}	696	40	696
ROLLO-1 (192)	2	97	79	8	8	1	2^{-33}	958	40	958
ROLLO-1 (256)	2	113	97	9	9	1	2^{-32}	1,371	40	1,371
Layered ROLLO-1 (128)	2	74	67	3	2	2	2^{-31}	1,240	120	620
Layered ROLLO-1 (192)	2	86	79	4	3	2	2^{-35}	1,699	120	850
Layered ROLLO-1 (256)	2	106	97	5	3	2	2^{-38}	2,571	120	1,286

■ Original ROLLO-1, Layered ROLLO-1, Modified Layered ROLLO-1 의 파라미터, 키 크기

Table 5: Modified Layered ROLLO-1의 파라미터

Instances (security)	q	$n^{(1)}$	$n^{(2)}$	m	r	d	$\deg(PI)$	DFR	$\mathsf{PK}_{\mathsf{size}}$ (Bytes)	$\mathsf{CT}_{\mathsf{size}}$ (Bytes)
Modified Layered ROLLO-1 (128)	2	37	61	67	6	2	11	2^{-25}	1,022	511
Modified Layered ROLLO-1 (192)	2	43	71	79	7	3	15	2^{-22}	1,403	702
Modified Layered ROLLO-1 (128)	2	53	103	97	7	3	20	2^{-32}	2,498	1,249

■ Modified Layered ROLLO-1이 Layered ROLLO-1 보다 향상된 성능을 보임

- Degenerated DFR이 감소함
- 이론적 복호화 복잡도가 향상됨
- 암호화 알고리즘 과정에 소모되는 사이클의 수가 감소함 (key gen, encap, decap)

Table 6: Decoding complexity (theoretical)

Instances (security)	DFR	decodig complexity	total cycle
Layered ROLLO-1 (128)	2^{-31}	13.236	2,965,682
Layered ROLLO-1 (192)	2^{-31}	16.0587	2,629,226
Layered ROLLO-1 (128)	2^{-38}	16.9987	4,424,634
Modified Layered ROLLO-1 (128)	2^{-25}	15.236	1,419,768
Modified Layered ROLLO-1 (192)	2^{-22}	17.6734	1,464,002
Modified Layered ROLLO-1 (128)	2^{-32}	17.9695	2,940,852

3. PALOMA

proposed by 김동찬, 전창열, 김영효, 김민지

■ PALOMA의 설계 배경

- 이진 고파 부호 사용
  - 전통적인 McEliece 암호시스템에 사용됨
  - 기반 문제인 신드롬 복호화 문제가 NP-hard라는 것이 이미 검증되어 있음
  - 효율적인 복호화 알고리즘을 보유하고 있음 (Extended Patterson 복호화 알고리즘)

- 키 설정
  - 이진 separable Goppa 부호 $\mathcal{C}$를 scramble한 부호 $\hat{\mathcal{C}}$사용
  - $\hat{\mathcal{C}}$의 systematic 패리티 검사 행렬의 submatrix를 공개키로 설정
  - $\mathcal{C}$의 scrambling과 디코딩에 대한 정보를 비밀키로 설정

■ PALOMA의 Encryption 과 Decryption

Input: A public key $pk = \widehat{\mathbf{H}}_{[n-k:n]} \in \mathbb{F}_2^{(n-k)\times n}$ and an error vector $\widehat{e} \in \mathbb{F}_2^n$ with $w_H(\widehat{e}) = t$
Output: A syndrome vector $\widehat{s} \in \mathbb{F}_2^{n-k}$

procedure ENCRYPT($pk = \widehat{\mathbf{H}}_{[n-k:n]}; \widehat{e}$)
  $\widehat{\mathbf{H}} \leftarrow [\mathbf{I}_{n-k} \mid \widehat{\mathbf{H}}_{[n-k:n]}] \in \mathbb{F}_2^{(n-k)\times n}$
  $\widehat{s} \leftarrow \widehat{\mathbf{H}}\widehat{e} \in \mathbb{F}_2^{n-k}$
  return $\widehat{s}$
end procedure

Input: A secret key $sk = (L, g(X), \mathbf{S}^{-1}, r)$ and a syndrome vector $\widehat{s} \in \mathbb{F}_2^{n-k}$
Output: An error vector $\widehat{e} \in \mathbb{F}_2^n$ with $w_H(\widehat{e}) = t$

procedure DECRYPT($sk = (L, g(X), \mathbf{S}^{-1}, r); \widehat{s}$)
  $s \leftarrow \mathbf{S}^{-1}\widehat{s}$
  $e \leftarrow$ RECERRVEC($L, g(X); s$) ▷ Algorithm 7
  $\mathbf{P}, \mathbf{P}^{-1} \leftarrow$ GENRANDPERMMAT(r) ▷ Algorithm 5
  $\widehat{e} \leftarrow \mathbf{P}^{-1}e$
  return $\widehat{e}$
end procedure

- Algorithm 7 : Extended Patterson Decoding을 사용하여 $\mathcal{C}$에서 에러 벡터를 복구하는 알고리즘
- Algorithm 5 : 랜덤 permutation 행렬 P를 생성하는 알고리즘

■ PALOMA의 Encapsulation

Input: A public key $pk \in \{0,1\}^{(n-k)\times n}$

Output: A key $k \in \{0,1\}^{256}$ and a ciphertext $c = (\widehat{r}, \widehat{s}) \in \{0,1\}^{256} \times \{0,1\}^{n-k}$

procedure ENCAP(pk)
  $r^* \xleftarrow{\$} \{0,1\}^{256}$
  $e^* \leftarrow \text{GENRANDERRVEC}(r^*)$ ▷ Algorithm 13
  $\widehat{r} \leftarrow \text{RO}_G(e^*)$ ▷ $\widehat{r} \in \{0,1\}^{256}$
  $\mathbf{P}, \mathbf{P}^{-1} \leftarrow \text{GENRANDPERMMAT}(\widehat{r})$ ▷
  $\widehat{e} \leftarrow \mathbf{P}e^*$
  $\widehat{s} \leftarrow \text{ENCRYPT}(pk; \widehat{e})$ ▷ $\widehat{s} \in \{0,1\}^{n-k}$
  $k \leftarrow \text{RO}_H(e^* \| \widehat{r} \| \widehat{s})$ ▷ $k \in \{0,1\}^{256}$
  return k and $c = (\widehat{r}, \widehat{s})$
end procedure

- Algorithm 13 : 랜덤 error vector를 생성하는 알고리즘

■ PALOMA의 Decapsulation

Input: A secret key $sk = (L, g(X), \mathbf{S}^{-1}, r)$ and a ciphertext $c = (\widehat{r}, \widehat{s}) \in \{0,1\}^{256} \times \{0,1\}^{n-k}$
Output: A key $k \in \{0,1\}^{256}$

procedure DECAP($sk = (L, g(X), \mathbf{S}^{-1}, r); c = (\widehat{r}, \widehat{s})$)
  $\widehat{e} \leftarrow$ DECRYPT($sk; \widehat{s}$) ▷ Algorithm 6
  $\mathbf{P}, \mathbf{P}^{-1} \leftarrow$ GENRANDPERMMAT($\widehat{r}$) ▷ Algorithm 5
  $e^* \leftarrow \mathbf{P}^{-1}\widehat{e}$ ▷ $\widehat{e} \in \{0,1\}^n$
  $\widehat{r}' \leftarrow \mathrm{RO}_G(e^*)$
  $\widetilde{e} \leftarrow$ GENRANDERRVEC(r)
  if $\widehat{r}' \neq \widehat{r}$ **then** ▷ $k \in \{0,1\}^{256}$
    return $k \leftarrow \mathrm{RO}_H(\widetilde{e} \| \widehat{r} \| \widehat{s})$
  end if
  $k \leftarrow \mathrm{RO}_H(e^* \| \widehat{r} \| \widehat{s})$ ▷ $k \in \{0,1\}^{256}$
  return k
end procedure

- Algorithm 6 : PALOMA의 encryption & decryption 알고리즘
- Algorithm 5 : 랜덤 permutation 행렬 P를 생성하는 알고리즘

■ PALOMA와 classic McEliece 의 키 크기, 퍼포먼스 속도 비교

Table 7: PLOMA & classic McEliece 비교

Security	Algorithm	PK_{size} (bytes)	SK_{size} (bytes)	CT_{size} (bytes)	Key (bytes)	키 생성 (millisec)	ENCAP (millisec)	DECAP (millisec)
128	mceliece348864	261,120	6,452	128	32	74	0.04	18
	PALOMA-128	319,488	94,496	136	32	64	0.03	9
192	mceliece460896	524,160	13,568	188	32	211	0.06	42
	PALOMA-192	812,032	355,400	240	32	258	0.04	58
256	mceliece6688128	1,044,992	13,892	240	32	517	0.10	82
	PALOMA-256	1,025,024	357,064	240	32	323	0.04	58

■ PALOMA와 classic McEliece 알고리즘에 대한 공격들의 계산복잡도 비교

	BJMM-ISD	Improved Birthday-type Decoding	Birthday-type Decoding	Exhaustive Search
PALOMA-128	$2^{166.21}$ $(l = 67, p = 14)$	$2^{225.78}$	$2^{244.11}$	$2^{476.52}$
PALOMA-192	$2^{267.77}$ $(l = 105, p = 22)$	$2^{399.67}$	$2^{448.91}$	$2^{885.11}$
PALOMA-256	$2^{289.66}$ $(l = 126, p = 26)$	$2^{415.59}$	$2^{464.66}$	$2^{916.62}$
mceliece348864	$2^{161.97}$ $(l = 66, p = 14)$	$2^{220.26}$	$2^{238.75}$	$2^{465.91}$
mceliece460896	$2^{215.59}$ $(l = 86, p = 18)$	$2^{311.80}$	$2^{345.58}$	$2^{678.88}$
mceliece6688128	$2^{291.56}$ $(l = 126, p = 26)$	$2^{416.95}$	$2^{466.01}$	$2^{919.32}$
mceliece6960119	$2^{289.92}$ $(l = 136, p = 28)$	$2^{402.41}$	$2^{443.58}$	$2^{874.57}$
mceliece8192128	$2^{318.34}$ $(l = 157, p = 32)$	$2^{436.05}$	$2^{484.90}$	$2^{957.10}$

Sven. Schange from Eindhoven university is in the process of cryptoanalysis

4. REDOG

proposed by 김종락, 홍지훈, T.S.C. Lau, 임영재, 원병선

■ REDOG의 설계 배경

- Breif History
  - McEliece와 Niederreiter 암호시스템의 특징을 결합한 McNie 제안 (LRPC 부호 기반)
  - Gaborit이 제안한 메세지 복구 공격으로 공개키의 차원이 감소됨
  - McNie를 개선한 Dual-Ouroboros 암호시스템 제안됨 (LRPC 부호 기반) - decoding failure probability 존재
  - Dual-Ouroboros에 Gabidulin 부호를 접목시킨 Do.Gab-PKE 제안 (zero-decoding failure probability)
  - Do.Gab-PKE 알고리즘에서 비밀키 S의 선택에 따른 암호 알고리즘의 부적합 발견
  - 비밀키 S를 적절하게 선택하여 잘 작동하게 되는 **RE**inforced modified **D**ual-**O**uroboros based on **G**abidulin 제안

■ Gabidulin 부호의 정의

- $\mathbb{F}_{q^m}^n$ 위에서의 Gabidulin 부호 $Gab_{n,k}(g)$ 다음과 같이 정의됨
  - Let $\mathbf{g} \in \mathbb{F}_{q^m}^n$ with $\mathrm{rk}(\mathbf{g}) = n \leq m$.
  - 차원이 k인 생성 벡터 $\mathbf{g}$로 만들어진 $[n, k]$ Gabidulin 부호 $\mathrm{Gab}_{n,k}(\mathbf{g})$ over $\mathbb{F}_{q^m}^n$는 $\mathbf{g}$ 로부터 얻어지는 Moore 행렬 G로 생성되는 부호임

  $\mathbf{g}$ 로부터 얻어지는 Moore 행렬 G는 다음과 같이 정의됨

$$G = \begin{bmatrix} g_1 & g_2 & \cdots & g_n \\ g_1^{[1]} & g_2^{[1]} & \cdots & g_n^{[1]} \\ \vdots & \vdots & \vdots & \vdots \\ g_1^{[n-1]} & g_2^{[n-1]} & \cdots & g_n^{[n-1]} \end{bmatrix}.$$

where $[l] = q^l$ as the lth Frobenius power for an integer l

■ REDOG의 Setup & Key generation

Setup: Generate global parameters with integers m, n, l, r, k such that $l < n$ and $\lambda t \leqslant r \leqslant \left\lfloor \frac{n-k}{2} \right\rfloor$. Output parameters $= (m, n, l, k, r, \lambda, t)$.

Key.Gen: Let $[H_1 H_2]$ be a parity check matrix for a $[2n-k, n]$ Gabidulin code $\mathcal{C}$ over $\mathbb{F}_{q^m}$, where $H_2 \in \mathrm{GL}_{n-k}(\mathbb{F}_{q^m})$. Let Φ_H be an efficient decoding algorithm for $\mathcal{C}$ with error correcting capability of $r = \left\lfloor \frac{n-k}{2} \right\rfloor$. Let $\mathcal{H}$ be a hash function from $\mathbb{F}_{q^m}^{2n-k}$ to $\mathbb{F}_{q^m}^{l}$.

Generate a generator matrix G for a random $[n, l]$ code over $\mathbb{F}_{q^m}$. Generate a random $n \times n$ isometric matrix P.

Generate a random λ-dimensional subspace, $\Lambda \subset \mathbb{F}_{q^m}$ such that $1 \in \Lambda$.

Generate a random $(n-k) \times (n-k)$ invertible matrix $S^{-1} \in \mathrm{GL}_{n-k}(\Lambda)$.

Output public key and secret key pair

$\mathsf{pk} = (G, F = GP^{-1}H_1^T[H_2^T]^{-1}S), \mathsf{sk} = (P, H, S, \Phi_H)$.

■ REDOG의 Encryption과 Decryption

$\mathsf{Enc}(\mathsf{pk}, \mathbf{m})$: Let $\mathbf{m} \in \mathbb{F}_{q^m}^{l}$ be the plaintext message to be encrypted. Generate randomly vector $\mathbf{e} = (e_1, e_2) \in \mathbb{F}_{q^m}^{2n-k}$ such that $\mathrm{rk}(\mathbf{e})=t$, $e_1 \in \mathbb{F}_{q^m}^{n}$ and $e_2 \in \mathbb{F}_{q^m}^{n-k}$. Let $\mathbf{m}' = \mathbf{m} + \mathcal{H}(\mathbf{e})$. Compute $c_1 = \mathbf{m}'G + e_1, c_2 = \mathbf{m}'F + e_2$. Output ciphertext $\mathbf{c} = (c_1, c_2)$.

$\mathsf{Dec}(\mathsf{sk}, c)$: Compute

$$
\begin{aligned}
& c_1 P^{-1} H_1^T - c_2 S^{-1} H_2^T \\
&\quad = \mathbf{m}' G P^{-1} H_1^T + e_1 P^{-1} H_1^T - \mathbf{m}' G P^{-1} H_1^T [H_2^T]^{-1} S S^{-1} H_2^T - e_2 S^{-1} H_2^T \\
&\quad = e_1 P^{-1} H_1^T - e_2 S^{-1} H_2^T \\
&\quad = (e_1 P^{-1}, -e_2 S^{-1}) \begin{bmatrix} H_1^T \\ H_2^T \end{bmatrix}
\end{aligned}
$$

Let $\mathbf{e}' = (e_1 P^{-1}, -e_2 S^{-1})$. Since $\mathrm{rk}(\mathbf{e}') \leqslant r$, apply Φ_H to obtain $\mathbf{e}'$.
Compute $e_1 = e_1 P^{-1} P$ and $e_2 = e_2 S^{-1} S$ to obtain $\mathbf{e} = (e_1, e_2)$.
Finally, solve the system $\mathbf{m}'G = c_1 - e_1$ to recover $\mathbf{m} = \mathbf{m}' - \mathcal{H}(e)$.

■ REDOG과 classic McEliece 의 키 크기 비교

Table 8: REDOG & classic McEliece 키 크기

Security	Algorithm	PK_{size} (KB)	SK_{size} (KB)	CT_{size} (KB)
128	mceliece348864	261.12	6.45	0.128
	REDOG-128	14.25	1.45	0.83
192	mceliece460896	524.16	13.57	0.188
	REDOG-192	32.84	2.52	1.44
256	mceliece6688128	1,044.99	13.89	0.240
	REDOG-256	62.98	3.89	2.23

- REDOG-128의 파라미터 : $(n, k, l, q, m, r, \lambda, t) = (44, 8, 37, 2, 83, 18, 3, 6)$
- REDOG-192의 파라미터 : $(n, k, l, q, m, r, \lambda, t) = (58, 10, 49, 2, 109, 24, 3, 8)$
- REDOG-256의 파라미터 : $(n, k, l, q, m, r, \lambda, t) = (72, 12, 61, 2, 135, 30, 3, 10)$

- 다음과 같은 안전성을 달성하고 공격들에 대해 안전함이 증명됨.
  - IND – CPA security
  - Key recovery 공격
  - Our plaintext recovery 공격
  - Message recovery 공격

• Lange-Pellegrini-Ravagnani 공격(7/13) : REDOG의 암호화 scheme에서 에러 벡터의 weight에 대한 공격

- REDOG의 파라미터 : $(n, k, l, q, m, r, \lambda, t)$

REDOG의 파라미터는 위와 같이 되어 있으며, S를 λ-dimensional subspace에서 선택하므로 error correcting bound r이 λ에 영향을 받음. 따라서 기존에 계산된 complexity가 reduction이 된다고 보임.

기존에 계산한 파라미터와 키 크기는 error-correcting bound인 r로 계산한 결과이므로, $(n, k, l, q, m, r, \lambda, t) = (44, 8, 37, 2, 83, 18, 3, 6)$이 128 security가 아닌 34 수준의 security를 갖는다고 설명함.

• Lange-Pellegrini-Ravagnani 공격(7/13) 에 대한 해결 방안

에러 벡터의 e의 구성 원소인 e_1, e_2 각각에 대한 weight을 설정하여 해결함. 자세한 내용은 아래와 같음.

- Decryption에서 $\mathbf{e}' = (e_1 P^{-1}, -e_2 S^{-1})$이 나타나므로 $\mathrm{rk}(e_1) \leq \frac{r}{2}$ & $\mathrm{rk}(e_2) \leq \frac{r}{2\lambda}$ 의 디테일 추가
- 위와 같이 설정을 추가하여 변경하면, 이전 페이지에서 언급한보안 수준의 변화는 다음과 같음.

Table 9: (44,8,37,2,83,18,3,6) 파라미터에 대한 security level

기존 document	Lange et al.의 계산	error weight 수정 후 계산
128	34	91

- $(n, k, l, q, m, r, \lambda, t) = (58, 10, 49, 2, 109, 24, 3, 8)$ 파라미터에 대한 보안 수준 : 192 → 149
- $(n, k, l, q, m, r, \lambda, t) = (72, 12, 61, 2, 135, 30, 3, 10)$ 파라미터에 대한 보안 수준 : 256 → 223

→ 큰 reduction이 발생하지 않음을 확인할 수 있음.

■ REDOG의 Encryption과 Decryption

$\mathsf{Enc}(\mathsf{pk}, \mathbf{m})$: Let $\mathbf{m} \in \mathbb{F}_{q^m}^{l}$ be the plaintext message to be encrypted. Generate randomly vector $\mathbf{e} = (e_1, e_2) \in \mathbb{F}_{q^m}^{2n-k}$ such that $\mathsf{rk}(\mathbf{e})=t$, $e_1 \in \mathbb{F}_{q^m}^{n}$ and $e_2 \in \mathbb{F}_{q^m}^{n-k}$. Let $\mathbf{m}' = \mathbf{m} + \mathcal{H}(\mathbf{e})$. Compute $c_1 = \mathbf{m}'G + e_1$, $c_2 = \mathbf{m}'F + e_2$. Output ciphertext $\mathbf{c} = (c_1, c_2)$.

$e = (e_1, e_2)$, where $\mathsf{rk}(e_1) \leq \frac{r}{2}$ and $\mathsf{rk}(e_2) \leq \frac{r}{2\lambda}$

$\mathsf{Dec}(\mathsf{sk}, c)$: Compute

$$
\begin{aligned}
& c_1 P^{-1} H_1^T - c_2 S^{-1} H_2^T \\
&\quad = \mathbf{m}' G P^{-1} H_1^T + e_1 P^{-1} H_1^T - \mathbf{m}' G P^{-1} H_1^T [H_2^T]^{-1} S S^{-1} H_2^T - e_2 S^{-1} H_2^T \\
&\quad = e_1 P^{-1} H_1^T - e_2 S^{-1} H_2^T \\
&\quad = (e_1 P^{-1}, -e_2 S^{-1}) \begin{bmatrix} H_1^T \\ H_2^T \end{bmatrix}
\end{aligned}
$$

Let $\mathbf{e}' = (e_1 P^{-1}, -e_2 S^{-1})$. Since $\mathsf{rk}(\mathbf{e}') \leqslant r$, apply Φ_H to obtain $\mathbf{e}'$.
Compute $e_1 = e_1 P^{-1} P$ and $e_2 = e_2 S^{-1} S$ to obtain $\mathbf{e} = (e_1, e_2)$.
Finally, solve the system $\mathbf{m}'G = c_1 - e_1$ to recover $\mathbf{m} = \mathbf{m}' - \mathcal{H}(e)$.

Error vector의 범위를 수정한 REDOG scheme을 기준으로 128-security를 만족하는 파라미터 소개

■ REDOG과 classic McEliece 의 키 크기 비교 2

Table 10: REDOG & classic McEliece 키 크기

Security	Algorithm	PK_{size} (KB)	SK_{size} (KB)	CT_{size} (KB)
128	mceliece348864	261.12	6.45	0.128
	new-REDOG-128	32.8	2.52	1.444
192	mceliece460896	524.16	13.57	0.188
	new-REDOG-192	62.98	3.89	2.227

- REDOG-128의 파라미터 : $(n, k, l, q, m, r, \lambda, t) = (58, 10, 49, 2, 109, 24, 3, 8)$
- REDOG-196의 파라미터 : $(n, k, l, q, m, r, \lambda, t) = (72, 12, 61, 2, 135, 30, 3, 10)$

암호문의 크기는 Classic McEliece보다 크지만, 여전히 공개키 크기와 비밀키 크기는 같은 수준의 McEliece보다 작음을 확인할 수 있음.

- Jong-Seon No, et al. “Enhanced pqsigRM: Code-Based Digital Signature Scheme with Short Signature and Fast Verification for Post-Quantum Cryptography” KpqC Round 1 (2022).
- Chanki Kim, et al. “Layered ROLLO-I: Faster rank-metric code-based KEM using ideal LRPC codes” KpqC Round 1 (2022).
- Dong-Chan Kim, et al. “PALOMA: Binary Separable Goppa-based KEM” KpqC Round 1 (2022).
- Jon-Lark Kim, et al. “REDOG” KpqC Round 1 (2022).