




NCC-Sign: Non-Cyclotomic Polynomial과 Trinomial을 이용한 격자 기반 전자서명 알고리즘

NIMS 암호기술연구팀
심경아, 김정수, 권혁동



목 차

- 설계 방향/원칙
 - NCC-Sign
 - 안전성 분석 및 안전성 증명
 - 파라미터 설정
 - 최적 구현
 - 향후 계획
- 



격자 기반 전자서명 알고리즘

▪ [격자 기반 전자서명 알고리즘]

- GGH scheme, NTRUSign: Nguyen-Regev의 parallelepiped 공격
- Gentry-Peikert-Vaikuntanathan (GPV): hash-and-sign type 전자서명 제안
- Lyubashevsky: LWE/SIS 문제 기반 Fiat-Shamir aborts type 전자서명, general lattices에서의 worst-case problem로의 안전성 증명
- Guneyasu- Lyubashevsky-Poppelmann (GLP): DCK/RSIS 문제 기반 압축 기법
- Bai and Galbraith (BG): LWE 문제 기반 개선된 압축 기법
- BG framework : RLWE/RSIS 문제 기반 qTESLA, MLWE/MSIS 기반 Dilithium, AMLWE/AM SIS 문제 기반 Aegis-Sig (중국 표준), MLWR/MSIS 문제 기반 MLWRSig 등
- Hash-and-Sign type 전자서명: NTRU 문제 기반 FALCON, MITAKA, Module-NTRU 문제 기반 ModFalcon 등



설계 방향/원칙

- **[효율성에 초점을 맞춘 설계]** 대부분 효율적인 격자 기반 암호 power-of-2 cyclotomic polynomial
 - 상대적으로 짧은 키길이, 서명 길이, 우수한 성능
 - 공격의 원인이 되는 부가적 구조
 - ✓ $\mathbb{Q}[x]/(\phi(x))$ 의 subfield 이용
 - ✓ $\mathbb{Q}[x]/(\phi(x))$ 의 small Galois Group 이용
 - ✓ $\mathbb{Z}/q[x]/(\phi(x)) \rightarrow$ smaller ring: ring Homomorphism 이용
 - 양자 알고리즘
 - ✓ Fully Homomorphic Encryption을 깨는 다항식 시간 양자알고리즘 존재
 - Principal ideal 에서의 short generator를 찾는 문제
 - ✓ Soiloquy, cyclotomic case of Gentry's original HFE (STOC 2009), cyclotomic case of Garg-Gentry-Halevi multilinear map을 깨는 양자 알고리즘 존재
 - ✓ S-unit attack, Twisted-PHS 알고리즘: Approx-SVP on ideal lattices



설계 방향/원칙

- **[NCC-Sign]** Non-cyclotomic과 cyclotomic 모두 지원하는 Ring-LWE 기반 전자서명
 - power-of-2 cyclotomic 보다 강한 안전성 제공
 - power-of-2 cyclotomic polynomial 대신 파라미터 점프가 없는 polynomial ring 사용
 - 키/서명 길이의 축소보다는 안전성/속도에 초점을 둔 설계
- **[안전성 강화]** 최초의 non-cyclotomic 격자 기반 전자서명 알고리즘
 - Intermediate Security Guarantee : Standard lattice 기반 > Non-cyclotomic > power-of-2 cyclotomic
 - 구조의 최소화: prime-degree large Galois group, inert modulus, $\phi(X) = X^p - X + 1$
- **[보수적인 안전성 유지하면서 효율성 증대]** Trinomial 사용 $\phi(X) = X^n - X^{n/2} - 1$, $n = 2^a \cdot 3^b$
- **[Dilithium의 Design Paradigm]** Fiat-Shamir with aborts 기반 전자서명, 검증된 방법론, 공개키 압축 방법 이용한 공개키 축소
- **[보수적인 파라미터 선택]**
 - Core-SVP 복잡도 128, 192, 256에 밀접하거나 초과하도록 선택
 - Dilithium 보다 더 높은 복잡도 제공



설계 방향/원칙

▪ [파라미터 선택의 유연성]

- 기존 power-of-2 cyclotomic ring에서의 RLWE 기반 전자서명의 경우 파라미터가 2배 씩 커지고, MLWE 기반 전자서명의 경우 256 비트 씩 커지는 파라미터 점프로 유연한 파라미터 선택 어려움
- NCC-Sign은 이런 파라미터 jump가 없어 유연하고, 보수적인 파라미터 선택 가능

▪ [빠른 성능]

- NCC-Sign trinomial은 Dilithium 보다 성능 우위 (Reference 구현 기준)
- AVX2 최적 구현의 경우 현재 Dilithium 보다 느림, 향상의 여지

▪ [부채널 공격 대응]

- Uniform distribution 사용으로 discrete Gaussian distribution을 겨냥한 부채널 공격에 안전
- 시간차 공격 대응을 위한 constant-time 구현



설계 방향/원칙

▪ [파라미터 선택의 유연성]

- 기존 power-of-2 cyclotomic ring에서의 RLWE 기반 전자서명의 경우 파라미터가 2배 씩 커지고, MLWE 기반 전자서명의 경우 256 비트 씩 커지는 파라미터 점프로 유연한 파라미터 선택 어려움
- NCC-Sign은 이런 파라미터 jump가 없어 유연하고, 보수적인 파라미터 선택 가능

▪ [빠른 성능]

- NCC-Sign trinomial은 Dilithium 보다 성능 우위 (Reference 구현 기준)
- AVX2 최적 구현의 경우 현재 Dilithium 보다 느림, 향상의 여지

▪ [부채널 공격 대응]

- Uniform distribution 사용으로 discrete Gaussian distribution을 겨냥한 부채널 공격에 안전
- 시간차 공격 대응을 위한 constant-time 구현



Supporting 알고리즘

- SampleInBall: $\{0, 1\}^*$ 의 string을 B_τ 로 보내는 함수

Algorithm 2: Decompose $_q(r, \alpha)$

```

1  $r := r \bmod^+ q$ 
2  $r_0 := r \bmod^\pm \alpha$ 
3 if  $r - r_0 = q - 1$  then
4    $r_1 := 0$ 
5    $r_0 := r_0 - 1$ 
6 else
7    $r_1 := (r - r_0)/\alpha$ 
8 return  $(r_1, r_0)$ 

```

Algorithm 3: UseHint $_q(h, r, \alpha)$

```

1  $m := (q - 1)/\alpha$ 
2  $(r_1, r_0) := \text{Decompose}_q(r, \alpha)$ 
3 if  $h = 1$  and  $r_0 > 0$  then
4   return
      $(r_1 + 1) \bmod^+ m$ 
5 if  $h = 1$  and  $r_0 \leq 0$  then
6   return
      $(r_1 - 1) \bmod^+ m$ 
7 return  $r_1$ 

```

Algorithm 4: Power2Round $_q(r, d)$

```

1  $r := r \bmod^+ q$ 
2  $r_0 := r \bmod^\pm 2^d$ 
3 return  $((r - r_0)/2^d, r_0)$ 

```

Algorithm 5: HighBits $_q(r, \alpha)$

```

1  $(r_1, r_0) := \text{Decompose}_q(r, \alpha)$ 
2 return  $r_1$ 

```

Algorithm 6: LowBits $_q(r, \alpha)$

```

1  $(r_1, r_0) := \text{Decompose}_q(r, \alpha)$ 
2 return  $r_0$ 

```

Algorithm 7: MakeHint $_q(z, r, \alpha)$

```

1  $r_1 := \text{HighBits}_q(r, \alpha)$ 
2  $v_1 := \text{HighBits}_q(r + z, \alpha)$ 
3 return  $\llbracket r_1 \neq v_1 \rrbracket$ 

```




NCC-Sign

키 생성 알고리즘

- 공개키/비밀키 생성
- 공개키 압축: seed, 상위 비트 사용
- 전체 공개키, 압축 공개키 묶어주는 ph
- deterministic signing에서 random에 해당하는 값 생성 비밀키 K

Algorithm 8: KeyGen

```
1  $(\zeta, \zeta') \leftarrow \{0, 1\}^{256} \times \{0, 1\}^{256}$   
2  $(\xi_1, \xi_2, K) \in \{0, 1\}^{256} \times \{0, 1\}^{256} \times \{0, 1\}^{256} := H(\zeta')$   
3  $\mathbf{a} \in R_q := \text{ExpandA}(\zeta)$  (Trinomial version:  $\hat{\mathbf{a}}(\neq 0) \in T_q := \text{ExpandA}(\zeta)$ )  
4  $(\mathbf{s}_1, \mathbf{s}_2) \in S_\eta \times S_\eta := \text{ExpandS}(\xi_1, \xi_2)$   
5  $\mathbf{t} := \mathbf{a}\mathbf{s}_1 + \mathbf{s}_2$  (trinomial version:  $\mathbf{t} := \text{INTT}(\hat{\mathbf{a}} \circ \text{NTT}(\mathbf{s}_1)) + \mathbf{s}_2$ )  
6  $(\mathbf{t}_1, \mathbf{t}_0) := \text{Power2Round}_q(\mathbf{t}, d)$   
7  $ph \in \{0, 1\}^{256} := H(\zeta \parallel \mathbf{t}_1)$   
8 return  $(pk = (\zeta, \mathbf{t}_1), sk = (\zeta, ph, K, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0))$ 
```



NCC-Sign

- 전자 서명 생성 알고리즘 (sk, M)
 - 공개키 복구
 - y 생성, $w=Ay$, 상위 비트 w_1 계산
 - M 과 w_1 에 관한 해시 값 c 계산
 - $z=y+c s_1$
 - t_0 와 s_2 를 이용한 hint h 생성
 - 서명 값 $\sigma=(z, h, c)$

Algorithm 9: Sign(sk, M)

```

1  $a \in R_q := \text{ExpandA}(\zeta)$  (Trinomial version:  $\hat{a} \in T_q := \text{ExpandA}(\zeta)$ )
2  $\mu \in \{0, 1\}^{512} := H(ph \parallel M)$ 
3  $\kappa := 0, (z, h) := \perp$ 
4  $\rho \in \{0, 1\}^{512} := H(K \parallel \mu)$  (or  $\rho \leftarrow \{0, 1\}^{512}$  for randomized signing)
5 while  $(z, h) = \perp$  do
6    $y \in \tilde{S}_{\gamma_1} := \text{ExpandMask}(\rho, \kappa)$ 
7    $w := ay$  (Trinomial version:  $w := \text{INTT}(\hat{a} \circ \text{NTT}(y))$ )
8    $w_1 := \text{HighBits}_q(w, 2\gamma_2)$ 
9    $\tilde{c} \in \{0, 1\}^{256} := H(\mu \parallel w_1)$ 
10   $c \in B_\tau := \text{SampleInBall}_{p, \tau}(\tilde{c})$  (Trinomial version additionally stores
     $\hat{c} := \text{NTT}(c)$ )
11   $z := y + cs_1$  (Trinomial version:  $z := y + \text{INTT}(\hat{c} \circ \text{NTT}(s_1))$ )
12   $r_0 := \text{LowBits}_q(w - cs_2, 2\gamma_2)$  (Trinomial version performs
     $\text{INTT}(\hat{c} \circ \text{NTT}(s_2))$  to compute  $cs_2$ )
13  if  $\|z\|_\infty \geq \gamma_1 - \beta$  or  $\|r_0\|_\infty \geq \gamma_2 - \beta$  then
14     $(z, h) := \perp$ 
15  else
16     $h := \text{MakeHint}_q(-ct_0, w - cs_2 + ct_0, 2\gamma_2)$  (Trinomial version
    performs  $\text{INTT}(\hat{c} \circ \text{NTT}(t_0))$  to compute  $ct_0$ )
17    if  $\|ct_0\|_\infty \geq \gamma_2$  or the # of 1's in  $h$  is greater than  $\omega$ 
18      then
19         $(z, h) := \perp$ 
20     $\kappa := \kappa + 1$ 
21 return  $\sigma = (\tilde{c}, z, h)$ 

```



- 서명 검증 알고리즘 (pk, M, σ)
 - 공개키 복구
 - hint 이용 유효성 검증

Algorithm 10: $\text{Verify}(pk, M, \sigma = (\tilde{c}, \mathbf{z}, \mathbf{h}))$

- 1 $\mathbf{a} \in R_q := \text{ExpandA}(\zeta)$ (Trinomial version: $\hat{\mathbf{a}} \in T_q := \text{ExpandA}(\zeta)$)
- 2 $\mu \in \{0, 1\}^{512} := H(H(\zeta \parallel \mathbf{t}_1) \parallel M)$
- 3 $\mathbf{c} := \text{SampleInBall}(\tilde{c})$ (Trinomial version additionally stores $\hat{\mathbf{c}} := \text{NTT}(\mathbf{c})$)
- 4 $\mathbf{w}'_1 := \text{UseHint}_q(\mathbf{h}, \mathbf{az} - \mathbf{ct}_1 \cdot 2^d, 2\gamma_2)$ (Trinomial version performs $\text{INTT}(\hat{\mathbf{a}} \circ \text{NTT}(\mathbf{z}))$ and $\text{INTT}(\hat{\mathbf{c}} \circ \text{NTT}(2^d \cdot \mathbf{t}_1))$ to compute \mathbf{az} and $\mathbf{ct}_1 \cdot 2^d$, respectively)
- 5 return $\llbracket \|\mathbf{z}\|_\infty < \gamma_1 - \beta \rrbracket$ and $\llbracket \tilde{c} = H(\mu \parallel \mathbf{w}'_1) \rrbracket$ and $\llbracket \# \text{ of 1's in } \mathbf{h} \text{ is } \leq \omega \rrbracket$



안전성 증명 및 안전성 분석

Definition 2.2 (Decision RLWE Problem.) Given a pair (\mathbf{a}, t) decode with non-negligible advantage, whether it came from the RLWE distribution or it was generated uniformly at random from $R_q \times R_q$. The advantage of the adversary \mathcal{A} in solving decisional RLWE problem over the ring R_q is

$$\text{Adv}_{\chi}^{\text{RLWE}}(\mathcal{A}) := |Pr[b = 1 \mid \mathbf{a}, t \leftarrow R_q; b \leftarrow \mathcal{A}(\mathbf{a}, t)] - Pr[b = 1 \mid \mathbf{a} \leftarrow R_q, s_1, s_2 \leftarrow \chi; b \leftarrow \mathcal{A}(\mathbf{a}, \mathbf{a}s_1 + s_2)]|.$$

Definition 2.3 (RSIS Problem.) The advantage of the adversary \mathcal{A} to solve RSIS problem over the ring R_q is

$$\text{Adv}_{\gamma}^{\text{RSIS}}(\mathcal{A}) := Pr[0 < \|\vec{y}\|_{\infty} \leq \gamma \wedge [1 \mid \mathbf{a}_1 \mid \mathbf{a}_2] \cdot \vec{y} = 0 \mid \mathbf{a}_1, \mathbf{a}_2 \leftarrow R_q; \vec{y} \leftarrow \mathcal{A}(\mathbf{a}_1, \mathbf{a}_2)].$$

Definition 2.4. (SelfTargetRSIS Problem). For the cryptographic hash function H , the advantage of \mathcal{A} to solve SelfTargetRSIS problem $\text{Adv}_{H, \gamma}^{\text{SelfTargetRSIS}}(\mathcal{A})$ is defined as

$$Pr \left[\begin{array}{l} 0 \leq \|\vec{y}\|_{\infty} \leq \gamma \wedge \\ H(\mu \parallel \begin{bmatrix} 1 \\ \mathbf{a}_1 \mid \mathbf{a}_2 \end{bmatrix} \cdot \vec{y}) = \mathbf{c} \end{array} \mid \mathbf{a}_1, \mathbf{a}_2 \leftarrow R_q; \left(\vec{y} := \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{c} \end{bmatrix}, \mu \right) \leftarrow \mathcal{A}^{H(\cdot)}(\mathbf{a}_1, \mathbf{a}_2) \right].$$



안전성 증명 및 안전성 분석

- 안전성 증명 (Existential unforgeability in (Q)ROM)
 - 안전성 증명: 결정론적 UF-NMA 안전성 증명으로 QROM의 UF-CMA 안전성 증명
 - UF-CMA 증명을 위해 전자서명이 deterministic과 is zero-knowledge 를 만족하면 UF-NMA 증명으로 충분
 - Dilithium: CMA와 NMA reduction gap 발견, 수정
 - ✓ abort된 transcript의 확률 무시
 - ✓ 수정된 HVZK 시뮬레이터 rejected transcript를 제거하는 추가적인 하이브리드 단계
 - ✓ abort된 transcript의 확률 분석 -> commitment의 더 큰 minimum entropy가 요구됨



안전성 증명 및 안전성 분석

- 안전성 증명 (Existential unforgeability in (Q)ROM)
 - Dilithium의 안전성 증명이 유사하게 NCC-Sign 에도 적용
 - NCC-Sign: $k=l=1$
 - NCC-Sign은 두 다항식을 곱할 때 계수 증가, Dilithium보다 약간 bound가 더 커짐.
 - 최근의 수정된 Dilithium의 증명에서 더 큰 commitment min-entropy 요구, 안전성 증명을 위해서는 a 가 invertible 필요
 - ✓ Non-cyclotomic의 경우 a 는 항상 invertible
 - ✓ Trinomial의 경우는 a 의 invertible 여부를 확인해야함.
 - min-entropy 는 파라미터에 따라 약 $n/2 \sim n$ 으로, 안전도 1의 경우 500보다 큼.



안전성 증명 및 안전성 분석

- 안전성 분석

- RLWE, RSIS, SelfTargetRSIS, Primal attack, dual lattice attack
- MATZOV: improved dual lattice attack



Rejection sampling 반복 횟수

- 새로운 SampleInBall 함수

- 서로 다른 2개의 polynomial을 이용하여 challenge polynomial c 선택
- 반복횟수 $e^{(p_1\beta_2+p_2\beta_1)(1/\gamma_1+1/\gamma_2)}$, $\beta_1 = 2(\tau_1 + \tau_2)\eta$, $\beta_2 = (2\tau_1 + \tau_2)\eta$
- 향상 효과

Parameter	p	τ	κ	p_1, p_2	β_1, β_2	Exp. reps.	Speed-up
1	1021	25	190	510,511	104,76	5.44 (6.6)	1.21
3	1429	29	228	714,715	120,88	4.76 (5.7)	1.19
5	1913	32	259	956,957	128,96	4.42 (5.5)	1.24
1^c	1201	32	241	600,601	132,98	2.27 (2.5)	1.09
3^c	1607	32	254	803,804	132,98	2.7 (3.02)	1.11
5^c	2039	32	265	1019,1020	132,98	3.43 (3.95)	1.15

- 기존 반복횟수 $e^{p\beta(1/\gamma_1+1/\gamma_2)}$, $e^{n\beta(1/\gamma_1+1/\gamma_2)}$, $\beta = 2\tau\eta$.



안전한 파라미터 선택

▪ degree, modulus 선택

- NTRU Prime KEM: $p=653$, $q = 4621$, 상대적으로 작은 p , q 사용
- Dilithium: module 구조 이용 작은 degree, 모든 안전도에서 동일한 q 사용
- NCC-Sign non-cyclotomic

- ✓ 효율적인 rejection sampling 을 위한 큰 q 선택에 따른 안전한 p 선택
- ✓ 안전도에 따른 다른 q 선택: inert modulus q

- 주어진 p 에 대한 충분한 inert modulus 존재

p	q
1021	8348477, 8339581, 8333113
1429	8380087, 8376649, 8333131, 8332559
1913	8361623, 8343469, 8334383

➤ NCC-Sign trinomial

- ✓ m -th cyclotomic polynomial $\phi(X) = X^n - X^{n/2} - 1$, $m = 2^a \cdot 3^b$, $a, b \geq 1$, $n = \varphi(n) = m/3$.
- ✓ $n = 2^a \cdot 3^b$: 768, 864, 972, 1024, 1152, 1296, 1458, 1536, 1728, 1944, 2048, 2187, 2304.
- ✓ 안전도 1, 3에서 $1152 = 2^7 \cdot 3^2$, $1536 = 2^9 \cdot 3$.
- ✓ 안전도 5에서 $n = 2048 = 2^{11}$, $n = 2304 = 2^8 \cdot 3^2$
- ✓ $n=2048$ 의 경우 q 는 2^{23} 보다 작은 소수, $n=2304$ 의 경우 q 는 2^{23} 보다 큰 소수



안전한 파라미터 선택

■ Cost 분석

- Cost: Lattice estimator from <https://github.com/malb/lattice-estimator>
- LWE, SIS cost: <https://github.com/pq-crystals/security-estimates>
- Core SVP estimate: BKZ-b calls the SVP oracle of dimension b
 - ✓ Solving shortest vector problem in a lattices of dimension b cost
 - classical security: $2^{0.265b}$ quantum security: $2^{0.292b}$
 - ✓ Classical security: $2^{a+0.292b}$, quantum security: $2^{a/2+0.265b}$



NCC-Sign 파라미터

■ NCC-Sign non-cyclotomic version

➤ 보수적인 파라미터 선택

- ✓ Core-SVP estimate 복잡도 128, 192, 256에 밀접하게 선택

- ✓ NTT unfriendly ring

$$\mathbb{Z}_q[X]/(X^p - X + 1)$$

에서 NTT 사용

- LWE, SIS cost: <https://github.com/pq-crystals/security-estimates>

- LWE cost by the lattice estimator
<https://github.com/malb/lattice-estimator>.

Parameter/Security Level	1 ^c	3 ^c	5 ^c
p	1201	1607	2039
q	17279291	17305741	17287423
d [dropped bits from t] ($2^d \tau < \gamma_2$)	12	13	13
τ [# of ± 1 's in c]	32	32	32
challenge entropy [$\log(\frac{p}{\tau}) + \tau$]	241	254	265
γ_1 [y coefficient range]	2^{19}	2^{19}	2^{19}
γ_2 [low-order rounding range]	$(q-1)/70$ (= 246847)	$(q-1)/60$ (= 288429)	$(q-1)/58$ (= 298059)
η [secret key range]	2	2	2
β	128	128	128
ω [max # of 1's in hint]	80	80	80
Exp. reps. [$\approx e^{(p_1\beta_2+p_2\beta_1)(1/\gamma_1+1/\gamma_2)}$]	2.5	3.02	3.95
Key/Signature Size			
Public key size	1984	2443	3091
Secret key size	2800	3914	4940
Signature size	3186	4251	5385
SIS Hardness (Core-SVP)			
BKZ block size b to break SIS	463	666	895
Best known classical bit cost	135	194	261
Best known quantum bit cost	122	176	237
LWE Hardness (Core-SVP)			
BKZ block size b to break LWE	491	711	956
Best known classical bit cost	143	207	279
Best known quantum bit cost	130	188	253
LWE Estimator			
Cost to SIS (BKZ b)	155.5 (484)	218.1 (697)	289.7 (941)
Quantum cost to SIS	135.3	192.0	256.8
Cost to LWE (BKZ b)	167.3 (483)	229.3 (704)	298.1 (949)
Quantum cost to LWE	141.1	198.4	262.0



NCC-Sign 파라미터

■ NCC-Sign non-cyclotomic version

➤ 보수적인 파라미터 선택

- ✓ Core-SVP estimate 복잡도 128, 192, 256에 밀접하게 선택

- ✓ NTT unfriendly ring

$$\mathbb{Z}_q[X]/(X^p - X + 1)$$

에서 NTT 사용

➤ Hybrid attack: hybrid-decoding,

hybrid-dual attack:

https://github.com/bencrnts/hybrid_attacks

Parameter/Security Level	$1^{c,1}$	$3^{c,1}$	$5^{c,1}$
p	1201	1607	2039
q	17279291	17305741	17287423
d [dropped bits from t] ($2^d \tau < \gamma_2$)	12	13	13
τ [# of ± 1 's in c]	32	32	32
challenge entropy [$\log \binom{p}{\tau} + \tau$]	241	254	265
γ_1 [y coefficient range]	2^{19}	2^{19}	2^{19}
γ_2 [low-order rounding range]	$(q-1)/70 = 246847$	$(q-1)/60 = 288429$	$(q-1)/58 = 298059$
η [secret key range]	1	1	1
β	64	64	64
ω [max # of 1's in hint]	80	80	80
Exp. reps. [$\approx e^{(p_1\beta_2 + p_2\beta_1)(1/\gamma_1 + 1/\gamma_2)}$]	1.58	1.74	1.98
pk size	1984	2443	3091
sk size	2703	3817	4843
sig size	3936	5255	6659
BKZ block-size b to break SIS	463	666	895
Best Known Classical bit-cost	135	194	261
Best Known Quantum bit-cost	122	176	237
Best Plausible bit-cost	96	138	185
BKZ block-size b to break LWE	450	656	884
Best Known Classical bit-cost	131	191	258
Best Known Quantum bit-cost	119	174	234
Core-SVP cost by Lattice estimator			
BKZ block-size b to break LWE	442	642	863
Classical bit-cost (method)	129.1 (usvp)	187.8 (dual hybrid)	252.2 (dual hybrid)
Hybrid-decoding attack cost			
BKZ block-size b to break LWE	445	655	890
Classical bit-cost	168.6	231.4	301.5
Hybrid-dual attack cost			
BKZ block-size b to break LWE	430	621	842
Classical bit-cost	156.1	213.2	277.1



NCC-Sign 파라미터

■ NCC-Sign trinomial version

➤ 보수적인 파라미터 선택

✓ 안전도 마진 확보

✓ NTT friendly ring

$$\mathbb{Z}_q[X]/(X^n - X^{n/2} - 1)$$

에서의 NTT 사용

Parameter/Security Level	1	3	5'	5
n	1152	1536	2048	2304
q	8401537	8397313	8380417	8404993
d [dropped bits from t] ($2^d \tau < \gamma_2$)	12	12	11	13
τ [# of ± 1 's in c]	25	29	32	32
challenge entropy [$\log \binom{p}{\tau} + \tau$]	195	232	265	271
γ_1 [y coefficient range]	2^{18}	2^{18}	2^{18}	2^{19}
γ_2 [low-order rounding range]	131274	131208	130944	262656
η [secret key range]	1	1	1	1
β	50	58	64	64
ω [max # of 1's in hint]	80	80	80	80
Exp. reps. [$\approx e^{n\beta(1/\gamma_1+1/\gamma_2)}$]	1.93	2.76	4.49	2.32
Key/Signature Size				
pk size	1760	2336	3104	3200
sk size	2400	3168	3936	4992
sig size	2912	3872	5152	6080
SIS Hardness (Core-SVP)				
BKZ block-size b to break SIS	462	671	963	1005
Best Known Classical bit-cost	135	196	281	293
Best Known Quantum bit-cost	122	177	255	266
LWE Hardness (Core-SVP)				
BKZ block-size b to break LWE	451	652	934	1078
Best Known Classical bit-cost	131	190	273	315
Best Known Quantum bit-cost	119	172	247	285
Lattice estimator (Core-SVP)				
BKZ block-size b to break LWE	452	652	930	1072
Classical bit-cost	132	190.7	271.7	313.3
(method)	(usvp)	(dual hybrid)	(dual hybrid)	(dual hybrid)



NCC-Sign 구현

- Reference 구현: NCC-Sign non-cyclotomic

- Intel i9-10980XE 3.0GHz

- ✓ Toom-Cook

Algorithm/Security Level	I ^c	III ^c	V ^c
KeyGen	1,727,508	2,965,942	4,700,228
Sign	11,768,076	20,816,964	42,227,652
Verify	3,400,702	5,876,246	9,324,876

- ✓ NTT

Algorithm/Security Level	1 ^c	3 ^c	5 ^c
KeyGen	979,979	1,001,022	1,034,193
Sign	7,269,506	8,752,038	10,719,700
Verify	1,863,350	1,884,647	1,926,235



NCC-Sign 구현

■ NCC-Sign Trinomial

- Intel Xeon(R) Gold 6234 CPU, 3.3GHz
- 구현 결과 (Cycles): 10,000번 중간 값

■ Dilithium과 비교

- 레퍼런스 구현 Dilithium 보다 우수
- AVX2 최적구현은 Dilithium이 더 빠름.
 - ✓ Dilithium은 최적구현이 레퍼런스 구현보다 4-5배 빠름.
 - ✓ NCC-Sign은 2배 정도 빠름.
 - Dilithium의 SHAKE와 NTT의 최적화가 적용되지 않음.
 - NTT 최적화 등 향상의 여지

Scheme	Security Level	1	3	5
Performance (Reference Code, median cycles)				
NCC-Sign-Trinomial	KeyGen	240,496	324,140	488,168
	Sign	616,746	1,245,144	1,781,784
	Verify	339,698	460,808	722,320
Dilithium	KeyGen	283,234	535,272	821,502
	Sign	973,868	1,658,510	2,206,464
	Verify	311,572	512,900	852,874
Performance (AVX2-optimized, median cycles)				
NCC-Sign-Trinomial	KeyGen	164,184	218,772	335,440
	Sign	<u>290,396</u>	<u>553,728</u>	<u>838,432</u>
	Verify	158,138	200,242	340,382
Dilithium	KeyGen	73,720	126,556	198,860
	Sign	<u>178,436</u>	<u>289,862</u>	<u>353,008</u>
	Verify	79,534	128,602	199,366



NCC-Sign vs. Dilithium

■ 기반 문제

- RLWE/RSIS 문제 vs. MLWE/MSIS 문제
- 다른 파라미터 설정

■ 복잡도, 사이즈 비교

- Dilithium 대비 안전도 1,3,5에서
최소 8비트 최대 41비트
안전성 마진 제공

Scheme	Core-SVP/Size	1	3	5 (5')
Dilithium	SIS	123	186	265
	LWE	123	182	252
	Repetitions	4.25	5.1	3.85
	Public key size	1312	1952	2592
	Signature size	2420	3293	4595
NCC-Sign Non-Cyclotomic ($\eta = 2$)	SIS	135	194	261
	LWE	143	207	279
	Repetitions	2.27	2.7	3.43
	Public key size	1984	2443	3091
	Signature size	3186	4251	5385
NCC-Sign Non-Cyclotomic ($\eta = 1$)	SIS	135	194	261
	LWE	131	191	258
	Repetitions	1.58	1.74	1.98
	Public key size	1984	2443	3091
	Signature size	3936	5255	6659
NCC-Sign Trinomial	SIS	135	196	293 (281)
	LWE	131	190	315 (273)
	Repetitions	1.93	2.71	2.32 (4.49)
	Public key size	1760	2336	3200 (3104)
	Signature size	2912	3872	6080 (5152)



NCC-Sign 적용 사례

■ NCC-Sign Trinomial 이용한 인증 체계 구축

- 안전도 1
- 키쌍 생성, 인증서 생성
인증서 검증 10,000번
수행한 평균

Public key algorithm: NCC_tri_avx_mode2, Signature algorithm: SHAKE256WithNCC

```
# cd ~/server/bin
# ./pqcca gen ca cert -c rootca_NCC_tri_avx_mode2 -a NCC_tri_avx_mode2 -v 10 -p 1q2w3e4r -t 10000
Generate Certificate Info
- cn : [rootca_NCC_tri_avx_mode2]
- dn(auto gen) : [c=KR,o=NIMS,ou=cryptography R&D center,cn=rootca_NCC_avx_tri_mode2]
- public key algorithm : [NCC_avx_tri_mode2]
- validate date : [10]
- sign algorithm : [SHAKE256WithNCC]
- private key password : [ ]
- performance test count: [10000]
correct? (y/n) : y
```

```
-----
performance test result (total count:10000)
-----
total execution time
- generate key pair job : 572.0767ms
- generate cert job : 2076.8621ms
- verify cert job : 3011.7058ms
average execution time
- generate key pair job : 0.0572ms
- generate cert job : 0.2077ms
- verify cert job : 0.3012ms
```



향후 계획

- AVX2 최적 구현
 - SHAKE, NTT 최적화
 - Complete-NTT vs. Incomplete-NTT
- 최적의 파라미터 선택
 - 높은 복잡도를 유지하면서 다양한 파라미터 선택 가능
 - 가장 효율적인 파라미터 선택의 여지

감사합니다.