NEV: Faster and Smaller NTRU Encryption using Vector Decoding

Jiang Zhang, Dengguo Feng, Di Yan

State Key Laboratory of Cryptology

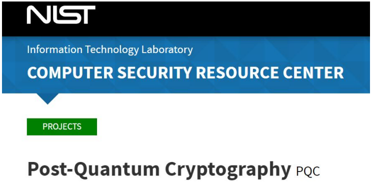
Asiacrypt 2023

Content

Content

NTRU

- NTRU, the first practical lattice-based encryption scheme [HPS98]
- One of the four PKEs/KEMs in NIST PQC Round 3 Finalist, but was not selected for standardization in the end [NIST-Round3].



Public-Key Encryption/KEMs

Classic McEliece

CRYSTALS–KYBER

NTRU

Saber

NTRU

- One main reason is that it is neither the fastest nor the smallest among the lattice KEM finalists [NIST-Status Report].

NIST IR 8413-upd1 Third Round Status Report

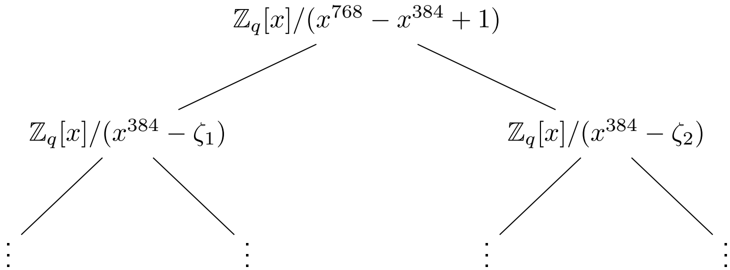
Overall assessment. One important feature of NTRU is that because it has been around for longer, its IP situation is more clearly understood. The original designers put their patents into the public domain [113], in addition to most of them having expired.

As noted by the submitters, NTRU may not be the fastest or smallest among the lattice KEM finalists, and for most applications and use cases, the performance would not be a problem. Nonetheless, as NIST has selected KYBER for standardization, NTRU will therefore not be considered for standardization in the fourth round.

- Compared to Kyber, NTRU has 8.3$\sim$18.6% larger public key and ciphertext sizes and is 8.21$\sim$45.34$\times$ slower in key generation.

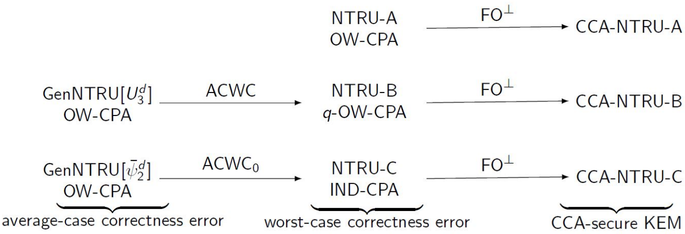
Recent Works: NTTRU

- Lyubashevsky and Seiler proposed NTTRU [LS19] over the specific cyclotomic ring $\mathbb{Z}_{7681}[x]/(x^{768} - x^{384} + 1)$ that supports NTT, and obtained significant speedup.



Recent Works: NTRU-A

- Duman et al. [DHK+21] extend the idea to other NTT-friendly rings of the same form $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$.
- Apply error-reducing transform to obtain 3 efficient NTRU designs NTRU-A/B/C with flexible parameter choices.



Recent Works: BAT

Despite of the efficiency improvement, the sizes of NTTRU and NTRU-A are still larger than that of Kyber at the same security levels.

Recent Works: BAT

Despite of the efficiency improvement, the sizes of NTTRU and NTRU-A are still larger than that of Kyber at the same security levels.

- Fouque et al. [FKPY22] proposed BAT, with a GGH-like encryption and decryption over the power of 2 cyclotomic ring $\mathbb{Z}_q[x]/(x^n + 1)$.
- BAT has the smallest size among all known lattice-based KEMs.
- It also enjoys fast encap/decap as Kyber and NTRU, but still suffers from a relatively slow key generation than Kyber and NTRU.

Our Scheme: NEV-PKE

NEV: a faster and smaller NTRU Encryption using Vector decoding over the power of 2 cyclotomic ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$.[1]

- Encode each plaintext bit into the most significant bits of multiple coefficients of the message polynomial;
- Use a vector of noised coefficients to decode each plaintext bit;
- Use (partial) NTT multiplications/inversions in R_q and precompute the inversion table to accelerate the scheme.

Reduce the size of q while keeping a reasonably negligible decryption failure and achieve faster implementation.

[1] One possible limitation: we cannot find a proper parameter for NIST L3 security.

Our Scheme: NEV-KEM

By applying the FO transformation, we obtain IND-CCA secure KEM.

For small modulus $q = 769$,

- NEV-512: $|pk| = |ct| = 615$ bytes, decryption failure $\leq 2^{-138}$
- NEV-1024: $|pk| = |ct| = 1229$ bytes, decryption failure $\leq 2^{-152}$
- 33 $\sim$ 48% (resp. 21%) more compact than NTRU (resp. Kyber)

In the round-trip time of ephemeral key exchange,

- NEV is 5.03 $\sim$ 29.94$\times$ faster than NTRU
- NEV is 1.42 $\sim$ 1.74$\times$ faster than Kyber

Optimized NEV'

NEV': better noise tolerance, smaller decryption failure and slightly better efficiency than NEV

- Based on a variant of RLWE problem, called Subset-Sum Parity RLWE (sspRLWE) problem;
- We show sspRLWE is polynomially equivalent to decisional RLWE for different parameters.

NEV' achieves decryption failure $\leq 2^{-200}$ at NIST L1 and L5 security.

Content

Hardness Assumptions for NTRU Encryption

Definition (Decisional NTRU Assumption)

The quotient $h = g/f$ of two randomly chosen small polynomials g, f is pseudorandom.

Definition (RLWE Assumption)

It is hard to recover e from $(h, hr + e)$ when h is uniformly random, and r, e are randomly chosen small polynomials.

Original NTRU Encryption

Original NTRU Encryption

- KeyGen: for small integer p and small polynomials f, g, output public key $h = pg/f \in R_q$ and keep secret key (f, g);
- Enc: compute $c = hr + m$;
- Dec: compute $u = fc = pgr + fm \in R_q$ and then $m = f^{-1}u \in R_p$.

Alternative Form

- To simplify the decryption, f is usually set to have the form of $f = pf' + 1$ s.t. $f^{-1} \bmod p = 1$;
- Then for decryption, we have $u = pgr + pf'm + m \in R_q$.

Disadvantages

$$u = fc = \underbrace{pgr + pf'm}_{\text{noise } \|\tilde{e}\|_\infty \le \frac{q-1}{2}} + m = \tilde{e} + m$$

Two main reasons why NTRU has larger public keys and ciphertexts sizes than its RLWE-based counterparts,

1. The decryption noise with $p = 3$ in NTRU is $1.5\times$ larger than that of its RLWE counterparts where $p = 2$ is typically used;
2. With a purposefully chosen "bad" message m, the noise term $pf'm$ may be utilized in a decryption failure attack[2] ; The naïve way to keep decryption error small is to increase q, which increases the sizes and weakens the security;

[2] This is why NTRU submitted to NIST sets its paras. to have no decryption failure.

Main idea

Using the plaintext encoding and vector decoding mechanism to increase the noise tolerance of NTRU and decrease the decryption failure.

- Our construction crucially relies on the power of 2 cyclotomic ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$.
- The small polynomial $v = (1 - x^{n/k})$ has a nice inverse polynomial $v^{-1} = \frac{q+1}{2}(1 + x^{n/k} + \cdots + x^{(k-1)n/k}) \in R_q$ s.t.,

$$
\begin{aligned}
v \cdot v^{-1} &= (1 - x^{n/k}) \cdot \frac{q+1}{2} \cdot (1 + x^{n/k} + \cdots + x^{(k-1)n/k}) \\
&= 1 \bmod q
\end{aligned}
$$

- We replace small integer p in NTRU with small polynomial v, and using v^{-1} as our plaintext encoding polynomial, i.e., $v^{-1}m$ copies k times the first n/k coefficients of m to obtain n coefficients.

Comparision

NTRU

- Public-key: $h = g/f = g/(pf' + 1)$
- Encryption: $c = phr + m$
- Decryption: $fc = pgr + pf'm + m$

NEV

- Public-key: $h = g/f = g/(vf' + 1)$
- Encryption: $c = hr + e + v^{-1}m$
- Decryption: $fc = gr + vf'e + f'm + e + v^{-1}m$

- NTRU encode the plaintext into the least significant bits of the coefficients of a message polynomial;
- NEV encode each plaintext bit into the most significant bits of multiple coefficients of the message polynomial;
- In decryption, a vector of noised coefficients can be used to decode each plaintext bit.

Noise Analysis

$$u = fc = \underbrace{gr + vf'e + f'm + e}_{\text{noise } \tilde{e} \ s.t. \ \|\tilde{e}\|_\infty \leq \frac{q-1}{4}} + v^{-1}m = \tilde{e} + v^{-1}m$$

The major reason that we can obtain a reasonably negligible decryption failure with very small modulus q is because,

1. The contribution of gr is much less than that of $vf'e$;
2. The size of $f'm$ is far smaller than that of gr because m only has non-zero binary coefficients at the first $l \leq n/k$ bits;
3. The magnitude of the major noise term $vf'e$ is at least $\sqrt{2}$ times smaller than that of using $p = 2, 3$ or $x + 2$;
4. The use of vector decoding will lower the decryption failure by roughly k times in the exponent.

Using v instead of p

We clarify that this slight modification will not require a stronger NTRU assumption because for publicly known fixed ring element v,

- The use of a polynomial $v = x + 2$ was recommended by the authors of NTRU as early as 2000 [HS00] and was investigated for years;
- The proof for the public key uniformity mainly depends on the properties of the distributions of g and f';
- The currently concrete security estimation also only cares about the distributions of g and f'.

Optimized NEV'-PKE

- When using PKE as KEM, the session key is randomly chosen and not necessarily known in advance;
- We can merge the sampling of encryption noise and random session key in a single step.

NEV'-PKE

- KeyGen: for random small polynomials f', g s.t $f = f' + v^{-1} \in R_q^*$, output public key $h = g/f \in R_q$ and keep secret key (f, g);
- Enc: for random small polynomials r, e, output $c = hr + e$;
- Dec: compute $u = fc = gr + f'e + v^{-1}e \in R_q$ and perform vector decoding $m' = \text{Poly2Pt}(u)$.

Optimized NEV'-PKE

In decryption algorithm where $u = fc = gr + f'e + v^{-1}e$,

- Let $\bar{v} = 1 + x^{n/k} + \cdots + x^{(k-1)n/k}$, let $e_0 = \bar{v}e \bmod 2$ and we have $2e_1 = \bar{v}e - e_0$;
- Since $v^{-1} = \frac{q+1}{2}\bar{v}$, then $v^{-1}e = e_1 + \frac{q+1}{2}e_0 \in R_q$ and we have

$$\begin{aligned} u &= fc = gr + f'e + v^{-1}e \\ &= gr + f'e + e_1 + \frac{q+1}{2} \underbrace{e_0}_{\bar{v}e \bmod 2} \in R_q \end{aligned}$$

- Let m be a polynomial only having n/k non-zero coefficients that are equal to the first n/k coefficients of e_0;
- Easy to check that e_0 is essentially a polynomial which copies k times the first n/k coefficients of m to obtain n coefficients;
- And we can use vector decoding again to recover m from u.

Binomial Noise Distribution

- To obtain an IND-CCA KEM, we have to convert NEV' into a PKE where m (or equivalently $\bar{v}e$ mod 2) is determined before e.
- Since $\bar{v}e$ essentially adds k coefficients of e to a single coefficient, we can easily achieve this goal by using binomial noise distribution B_η.

$$B_\eta = \left\{ \sum_{i=0}^{\eta-1} (a_i - b_i) : (a_0, \ldots, a_{\eta-1}, b_0, \ldots, b_{\eta-1}) \leftarrow \{0,1\}^{2\eta} \right\}$$

Example

For $\eta = 1$ and $k = 2$, we can "invert" a random bit b^* to 2 samples from B_1:

- Randomly choose $b_1, b_2, b_3 \leftarrow \{0, 1\}$;
- Set $b_0 = b^* \oplus b_1 \oplus b_2 \oplus b_3$;
- Output $e_0 = b_0 - b_1$, $e_1 = b_2 - b_3$.

Easy to check that $e_0 \pm e_1$ mod 2 $= b^*$, and $e_0, e_1 \sim B_1$ if b^* is random.

Comparision

NTRU

- Public-key: $h = pg/(pf' + 1)$
- Encryption: $c = hr + m$
- Decryption:
  $fc = pgr + pf'm + m$

NEV'

- Public-key: $h = vg/(vf' + 1)$
- Encryption: $c = hr + e$
- Decryption:
  $fc = gr + f'e + v^{-1}e$

NEV

- Public-key: $h = g/(vf' + 1)$
- Encryption: $c = hr + e + v^{-1}m$
- Decryption: $fc =$
  $gr + vf'e + f'm + e + v^{-1}m$

NEV'

- Public-key: $h = g/(f' + v^{-1})$
- Encryption: $c = hr + e$
- Decryption: $fc =$
  $gr + f'e + e_1 + \frac{q+1}{2}(\bar{v}e \bmod 2)$

sspRLWE Assumption

In order to prove the security of NEV', we introduce a variant of RLWE problem, called Subset-Sum Parity RLWE problem (sspRLWE).

Definition (sspRLWE Assumption)

It is hard to compute ve mod 2 for some fixed ring element $v \in R_2$ given an RLWE tuple $(h, hr + e)$ as input.

The name comes from the fact that, the i^{th} coefficient of ve mod 2 is essentially equal to the parity of the subset sum $\sum_{v_j=1} e_{(i-j) \bmod n}$ of the coefficient vector $e = (e_0, \cdots, e_{n-1})$.

Hardness of sspRLWE

Theorem

If there is a PPT algorithm $\mathcal{A}$ solving the sspRLWE problem with probability negligibly close to 1, then there is another PPT algorithm $\mathcal{A}'$ solving the DRLWE problem.

- For a DRLWE instance $(h, b = hr + e_1)$, $\mathcal{A}'$ can convert it to an sspRLWE instance $(h' = 2h, b' = 2b + e_0)$ with some noise e_0;
- Since $v(2e_1 + e_0) = ve_0 \bmod 2$, then by running algorithm $\mathcal{A}$ with input (h', b'), $\mathcal{A}'$ can obtain some $w \in R_2$ from $\mathcal{A}$.
- After checking $w \stackrel{?}{=} ve_0 \bmod 2$, $\mathcal{A}'$ returns 1 if $w = ve_0 \bmod 2$, and otherwise returns 0.

Hardness of sspRLWE

$\mathcal{C}'$		$\mathcal{A}'/\mathcal{C}$		$\mathcal{A}$
(h, b)	$\xrightarrow{(h,b)}$			
		$h' = 2h, b' = 2b + e_0$	$\xrightarrow{(h',b')}$	
			$\xleftarrow{w}$	
	$\xleftarrow{0/1}$	$w \stackrel{?}{=} ve_0 \bmod 2$		

- If (h, b) is a real DRLWE tuple, any PPT sspRLWE solver $\mathcal{A}$ will return $w = ve_0 \bmod 2$ with high probability;
- If (h, b) is randomly chosen, (h', b') is also randomly distributed and the adversary $\mathcal{A}$ can obtain no information.

Content

Practical Parameter Set

We present two parameter sets for NEV and NEV' aiming at NIST levels 1 and 5 security, respectively.

Parameters	(n, q)	Key Dist. (χ_f, χ_g)	Enc Dist. (χ_r, χ_e)	Size in Byte $(\lvert pk\rvert, \lvert ct\rvert)$	Dec Failure
NEV-512	(512,769)	(B_1, B_1)	$(B_1, \tau_{1/6})$	(615,615)	2^{-138}
NEV'-512	(512,769)	(B_1, B_1)	(B_1, B_1)	(615,615)	2^{-200}
NEV-1024	(1024,769)	(B_1, B_1)	$(B_1, \tau_{1/6})$	(1229,1229)	2^{-152}
NEV'-1024	(1024,769)	(B_1, B_1)	(B_1, B_1)	(1229,1229)	2^{-200}

Comparison with NTRU and Kyber

Comparison between our NEV, NTRU and Kyber in sizes and efficiency.

Schemes	Size in Byte $(\lvert pk\rvert, \lvert ct\rvert)$	Total in Byte $\lvert pk\rvert + \lvert ct\rvert$	Improv. Ratio	Speedup Ref/AVX2	NIST Security
Kyber-512	(800,768)	1568	21.56%↓	1.67/1.42↑	Level 1
NTRU-HPS2048677	(930,930)	1860	33.87%↓	18.46/5.74↑	
NTRU-HRSS701	(1138,1138)	2276	45.96%↓	19.92/5.03↑	
NEV-512	(615,615)	1230	-	-	
NEV'-512	(615,615)	1230	-	-	
Kyber-768	(1184,1088)	2272	$-8.19\%^{\dagger}$	$1.21/1.19^{\dagger}$	Level 3
NTRU-HPS4096821	(1230,1230)	2460	$0.08\%^{\dagger}$	$11.05/4.10^{\dagger}$	
Kyber-1024	(1568,1568)	3136	21.62%↓	1.74/1.62↑	Level 5
NTRU-HPS40961229	(1842,1842)	3684	33.28%↓	24.76/- ↑	
NTRU-HRSS1373	(2401,2401)	4802	48.81%↓	29.94/- ↑	
NEV-1024	(1229,1229)	2458	-	-	
NEV'-1024	(1229,1229)	2458	-	-	

†Note that we obtain figures for Kyber-768 and NTRUHPS-4096821 at Level 3 security by dividing that of our NEV-1024 at Level 5 security.

Comparison with State-of-the-Art

Comparison with recent NTRU variants in sizes and efficiency.

Schemes	Size in Byte $(\|pk\|, \|ct\|)$	Total in Byte $\|pk\| + \|ct\|$	Improv. Ratio	Speedup Ref/AVX2	LWE Estimator	NIST Security
NTRU-A$_{2593}^{576}$	(864,864)	1728	28.82%↓		154	Level 1
BAT-512	(521,473)	994	19.19%↑	140/973↑	144	
NTTRU-768	(1248,1248)	2496†			170	
NEV-512	(615,615)	1230			141	
NEV'-512	(615,615)	1230			145	
NTRU-A$_{3457}^{1152}$	(1728,1728)	3456	28.88%↓		305	Level 5
BAT-1024	(1230,1006)	2236	9.03%↑	334/2648↑	273	
NEV-1024	(1229,1229)	2458			281	
NEV'-1024	(1229,1229)	2458			292	

†Note that our NEV-1024 at NIST L5 Security is slightly more compact than NTTRU-768 at NIST L1 Security with comparable computational efficiency.

Conclusion

We present NEV, a faster and smaller NTRU Encryption using Vector Decoding

- $33\% \sim 48\%$ more compact and $5.03 \sim 29.94\times$ faster than NTRU
- 21% more compact and $1.42 \sim 1.74\times$ faster than Kyber

Thank You!

Full version: https://eprint.iacr.org/2023/1298

References

Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman
NTRU: A ring-based public key cryptosystem.
Algorithmic Number Theory. pp. 267C288. Springer, Berlin, Heidelberg (1998).

NIST Selected Algorithms 2022
https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

NIST.IR.8413 2022
Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process (2022).

Vadim Lyubashevsky, Gregor Seiler
NTTRU: Truly fast NTRU using NTT.
Cryptology ePrint Archive, https://eprint.iacr.org/2019/040 (2019).

Julien Duman, Kathrin Hövelmanns, Eike Kiltz, Vadim Lyubashevsky, Gregor Seiler, Dominique Unruh
A thorough treatment of highly-efficient NTRU instantiations.
Cryptology ePrint Archive, https://eprint.iacr.org/2021/1352 (2021).

References (cont.)

Pierre-Alain Fouque, Paul Kirchner, Thomas Pornin, Yang Yu
BAT: Small and fast kem over NTRU lattices.
IACR Transactions on CHES 2022(2), 240-265.

Jeffrey Hoffstein, Joseph H. Silverman
Optimizations for NTRU.
Public-Key Cryptography and Computational Number Theory, 2000: 77-88.