CRYSTAL-Kyber

Rana Barua

IAI, TCG CREST
Kolkata

June 26, 2025

Introduction

- **KYBER** is a quantum safe Key Encapsulation Mechanism(KEM) that was standardized by NIST in FIPS 203.
- There it was called Module -Lattice-based Key Encapsulation Mechanism (ML-KEM)
- Kyber-KEM was constructed by applying the Fujisaki-Okamoto transform to a public-key encryption scheme (Kyber-PKE).

- **Modular Reduction:** Given an even(odd) positive integer k, let r' be the unique integer in the range $-k/2 < r' \leq k/2$(resp.$(k-1)/2 \leq r' \leq (k+1)/2$) such that $r' \equiv r \bmod k$. In this case we write

$$r' = r \bmod {}^{\pm}k.$$

- **Norm:** Let $\mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$ and $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$. For an element $a \in \mathbb{Z}_q$, $||a||_\infty$ denotes $|a \bmod {}^{\pm}q|$. The ℓ_∞ and ℓ_2 norms of an element $w = w_0 + w_1 X + \ldots + w_{n-1} X^{n-1} \in \mathcal{R}$ are as follows.

$$||w||_\infty = Max_i ||w_i||_\infty; \; ||w|| = \sqrt{||w_0||_\infty^2 + \ldots + ||w_{n-1}||_\infty^2}.$$

For a vector $\mathbf{w} = (w_1, \ldots, w_k) \in \mathcal{R}^k$ the norms are similarly defined. We define S_η by

$$S_\eta = \{w \in \mathcal{R} : ||w||_\infty \leq \eta\}.$$

Compression and Decompression

- For $d < \lceil \log q \rceil$, define

$$\textbf{Compress}_q(x, d) = \lceil (2^d/q)x \rfloor \mod 2^d.$$

We also define

$$\textbf{Decompress}_q(x, d) = \lceil (q/2^d).x \rfloor.$$

- One can check that if

$$x' = \textbf{Decompress}_q(\textbf{Compress}_q(x, d), d),$$

then

$$|x' - x \mod {}^{\pm}q| \leq \lceil (q/2^{d+1}) \rfloor.$$

Parameters of Kyber

- Kyber is parameterized by integers $n, k, q, \eta_1, \eta_2, d_u$ and d_v as given below.

	n	k	q	η_1	η_2	(d_u, d_v)	δ
Kyber 512	256	2	3329	3	2	(10,4)	2^{-139}
Kyber768	256	3	3329	2	2	(10,4)	2^{-164}
Kyber1024	256	4	3329	2	2	(11,5)	2^{-174}

Here δ denotes the failure probability.

Kyber-PKE(Simplified): Key Generation

Key generation algorithm consists of the following

- Choose $\mathbf{A} \in_R \mathcal{R}_q^{k \times k}$; $\mathbf{s} \in_R S_{\eta_1}^k$ and $\mathbf{e} \in_R S_{\eta_2}^k$
- Compute $\mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e}$
- Alice's **public key** is $(\mathbf{A}, \mathbf{t})$ and her **private key** is $\mathbf{s}$

Remark: Computing $\mathbf{s}$ from $\mathbf{A}$ and $\mathbf{t}$ is an instance of Module-LWE.

Kyber-PKE: Encryption

To encryption a message $\mathbf{m} \in \{0, 1\}^n$,

- choose $\mathbf{r} \in_R S_{\eta_1}^k$; $\mathbf{e}_1 \in_R S_{\eta_2}^k$, and $e_2 \in_R S_{\eta_2}$.
- Compute $\mathbf{u} = \mathbf{A}^T\mathbf{r} + \mathbf{e}_1$ and $v = \mathbf{t}^T.\mathbf{r} + e_2 + \lceil q/2 \rfloor \mathbf{m}$
- Ciphertext is $\mathbf{c} = (\mathbf{u}, v)$.

Remark: Note that $\mathbf{c} \in \mathcal{R}_q^k \times \mathcal{R}_q$.

Kyber-PKE: Decryption

To decrypt a ciphertext $\mathbf{c} = (\mathbf{u}, v)$,

- compute $\mathbf{m} := Compress_q(v - \mathbf{s}^T\mathbf{u}, 1)$.
- **return m**

Remark: Note that Alice uses her secret key **s**

Correctness

Recall that

- $\mathbf{u} = \mathbf{A}^T\mathbf{r} + \mathbf{e}_1$
- $v = \mathbf{t}^T\mathbf{r} + e_2 + \lceil q/2 \rfloor \mathbf{m}$; and $\mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e}$.

Hence, $v - \mathbf{s}^T\mathbf{u}$

$$= \mathbf{t}^T\mathbf{r} + e_2 + \lceil q/2 \rfloor \mathbf{m} - \mathbf{s}^T(\mathbf{A}^T\mathbf{r} + e_1)$$

$$= (\mathbf{t}^T - \mathbf{s}^T\mathbf{A})\mathbf{r} + e_2 - \mathbf{s}^T\mathbf{e}_1 + \lceil q/2 \rfloor \mathbf{m}$$

$$= \mathbf{e}^T\mathbf{r} + e_2 - \mathbf{s}^T\mathbf{e}_1 + \lceil q/2 \rfloor \mathbf{m} = w + \lceil q/2 \rfloor \mathbf{m}, \quad \text{say.}$$

Correctness

Let $\mathbf{m}' = Compress_q(v - \mathbf{s}^T\mathbf{u}, 1)$. **Assume** that $||w||_\infty < q/4$. Then

$$q/4 \geq ||v - \mathbf{s}^T\mathbf{u} - \lceil q/2 \rfloor \mathbf{m}'||_\infty$$
$$= ||w + \lceil q/2 \rfloor (\mathbf{m} - \mathbf{m}'|)|_\infty$$

Hence $\lceil q/2 \rfloor ||\mathbf{m} - \mathbf{m}'||_\infty < q/2$. This is possible only if $\mathbf{m} = \mathbf{m}'$.

Security

- If M-LWE is hard, then simplified Kyber-PKE is CPA secure.

Kyber-KEM

- Kyber-KEM is obtained from Kyber-PKE by applying the "Fujisaki-Okazmoto" transform to Kyber-PKE.
- The FO transform is a generic transform that converts a CPA-secure primitive to a CCA-secure primitive.
- It uses three hash functions

$$\mathbf{G} : \{0,1\}^* \to \{0,1\}^{512}; \mathbf{H} : \{0,1\}^* \to \{0,1\}^{256}, \text{ and}$$

$$\mathbf{J} : \{0.1\}^* \to \{0,1\}^{256}.$$

Kyber-KEM KeyGen

Output: Public key *pk*; secret key *sk*

- $z \leftarrow \mathcal{B}^{32}$
- $(pk, sk') \leftarrow \text{Kyber.CPAPKE.KeyGen}()$
- $sk := (sk' || pk || \mathbf{H}(pk) || z)$
- Alice's encapsulation key is *pk* and decapsulation key is *sk*

Kyber-KEM: Encapsulation

INPUT: Public key pk
OUTPUT: Ciphertext c; shared key $K \in \mathcal{B}^*$

- $m \leftarrow \mathcal{B}^{32}$
- $m \leftarrow \mathbf{H}(m)$
- $(\bar{K}, r) := \mathbf{G}(m||\mathbf{H}(pk))$
- $c := Enc.CPAPKE.Enc(pk, m, r)$
- $K := \mathbf{KDF}(\bar{K}||\mathbf{H}(c))$
- **return**(c, K).

Kyber-KEM: Decapsulation

INPUT: Ciphertext *c*, secret key *sk*
OUTPUT: Shared key $K \in \mathcal{B}^*$

- Extract *pk* from *sk*
- Extract $h \in \mathcal{B}^{32}$ from *sk*
- Extract *z*from *sk*
- $m' := Kyber.\mathrm{CPAPKE.Dec(sk, c)}$
- $(\bar{K}', r') := \mathbf{G}(m' || h)$
- $c' := Kyber.\mathrm{CPAPKE.Enc(pk, m', r')}$
- **If** $c = c'$ **then**
- **return** $K := \mathbf{J}(\bar{K}' || \mathbf{H}(c))$
- **else**
- **return** $K := \mathbf{J}(z || \mathbf{H}(c))$
- **end if**
- **return** K