CRYSTALS–Kyber

Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, Damien Stehlé

authors@pq-crystals.org
https://pq-crystals.org/kyber
April 12, 2018
The big picture

Kyber.CPAPKE: LPR encryption or “Noisy ElGamal”

\[ \begin{align*}
    s, e & \leftarrow \chi \\
    sk = s, pk = t = As + e \\
\end{align*} \]

\[ \begin{align*}
    r, e_1, e_2 & \leftarrow \chi \\
    u & \leftarrow A^T r + e_1 \\
    v & \leftarrow t^T r + e_2 + Enc(m) \\
    c & = (u, v) \\
\end{align*} \]

\[ m = Dec(v - s^T u) \]
The big picture

**Kyber.CPAPKE: LPR encryption or “Noisy ElGamal”**

\[
\begin{align*}
  s, e &\leftarrow \chi \\
  sk = s, pk = t = As + e
\end{align*}
\]

\[
\begin{align*}
  r, e_1, e_2 &\leftarrow \chi \\
  u &\leftarrow A^T r + e_1 \\
  v &\leftarrow t^T r + e_2 + Enc(m) \\
  c = (u, v)
\end{align*}
\]

\[m = Dec(v - s^T u)\]

**Kyber.CCAKEM: CCA-secure KEM via tweaked FO transform**

- Enforce “honest” encapsulation
- Generate all randomness in encryption via PRG, encrypt seed
- Recover seed during decapsulation
- Reencrypt and compare ciphertexts
• Use MLWE instead of LWE or RLWE
  • Performance similar to RLWE
  • Very easy to scale security and performance
  • Remove some of the cyclic structure of RLWE
• Use $\mathcal{R} = \mathbb{Z}_q[X]/(X^{256} + 1)$ with $q = 7681$
  • Fast, simple, in-place negacyclic NTT for multiplication
  • Most widely studied and best understood structure
• Use MLWE instead of LWE or RLWE
  • Performance similar to RLWE
  • Very easy to scale security and performance
  • Remove some of the cyclic structure of RLWE
• Use $\mathcal{R} = \mathbb{Z}_q[X]/(X^{256} + 1)$ with $q = 7681$
  • Fast, simple, in-place negacyclic NTT for multiplication
  • Most widely studied and best understood structure
• Use centered binomial noise
  • Efficient to sample without timing leakage
• Use MLWE instead of LWE or RLWE
  • Performance similar to RLWE
  • Very easy to scale security and performance
  • Remove some of the cyclic structure of RLWE
• Use $\mathcal{R} = \mathbb{Z}_q[X]/(X^{256} + 1)$ with $q = 7681$
  • Fast, simple, in-place negacyclic NTT for multiplication
  • Most widely studied and best understood structure
• Use centered binomial noise
  • Efficient to sample without timing leakage
• Generate $A$ via $XOF(\rho)$ (“NewHope style”)
  • Avoid “nothing-up-my-sleeves” discussions
  • Avoid all-for-the-price-of-one attacks
  • Sample $A$ in NTT domain: save $k^2$ NTTs
• Compress ciphertexts (round off least-significant bits)
  • Reduce bandwidth requirements
  • Introduce extra “LWR” noise
• Compress ciphertexts (round off least-significant bits)
  • Reduce bandwidth requirements
  • Introduce extra “LWR” noise

• Compress public keys
  • Reduce bandwidth requirements
  • Adds MLWR-style assumption instead of pure reduction from MLWE (thanks to D’Anvers for pointing this out)
  • No actual attacks or security problems
  • Could fix proof by re-randomizing after decompression
- Compress ciphertexts (round off least-significant bits)
  - Reduce bandwidth requirements
  - Introduce extra “LWR” noise
- Compress public keys
  - Reduce bandwidth requirements
  - Adds MLWR-style assumption instead of pure reduction from MLWE (thanks to D’Anvers for pointing this out)
  - No actual attacks or security problems
  - Could fix proof by re-randomizing after decompression
- Allow decapsulation failures
  - Failure probability $< 2^{-140}$
  - Avoiding failures would cost security (or performance)
FO transform – tweaks

- Hash public key into seed and shared key
  - Multitarget protection against precomputation attacks
  - Obtain contributory KEM
- Hash ciphertext into shared key
- Shared key depends on full KEM transcript
- More robust when building, e.g., AKE from Kyber
- Use Keccak-based functions for all hashes and XOF
FO transform – tweaks

- Hash public key into seed and shared key
  - Multitarget protection against precomputation attacks
  - Obtain contributory KEM
- Hash ciphertext into shared key
  - Shared key depends on full KEM transcript
  - More robust when building, e.g., AKE from Kyber
Tweaks:

- Hash public key into seed and shared key
  - Multitarget protection against precomputation attacks
  - Obtain contributory KEM
- Hash ciphertext into shared key
  - Shared key depends on full KEM transcript
  - More robust when building, e.g., AKE from Kyber
- Use Keccak-based functions for all hashes and XOF
## Parameter sets and performance

<table>
<thead>
<tr>
<th>Parameter set</th>
<th>Sizes (in Bytes)</th>
<th>Haswell Cycles (AVX2)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Kyber512 ((k = 2, \text{level 1}))</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Sizes (in Bytes)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sk:</td>
<td>1632</td>
<td>gen: 55 160</td>
</tr>
<tr>
<td>pk:</td>
<td>736</td>
<td>enc: 75 680</td>
</tr>
<tr>
<td>ct:</td>
<td>800</td>
<td>dec: 74 428</td>
</tr>
<tr>
<td><strong>Kyber768 ((k = 3, \text{level 3}))</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Sizes (in Bytes)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sk:</td>
<td>2400</td>
<td>gen: 85 472</td>
</tr>
<tr>
<td>pk:</td>
<td>1088</td>
<td>enc: 112 660</td>
</tr>
<tr>
<td>ct:</td>
<td>1152</td>
<td>dec: 108 904</td>
</tr>
<tr>
<td><strong>Kyber1024 ((k = 4, \text{level 5}))</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Sizes (in Bytes)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sk:</td>
<td>3168</td>
<td>gen: 121 056</td>
</tr>
<tr>
<td>pk:</td>
<td>1440</td>
<td>enc: 157 964</td>
</tr>
<tr>
<td>ct:</td>
<td>1504</td>
<td>dec: 154 952</td>
</tr>
</tbody>
</table>
https://pq-crystals.org/kyber