HILA5: KEM and Public Key Encryption
From Ring-LWE and Error Correcting Codes

Markku-Juhani C. Saarinen
<mjos@mjos.fi>
F C. Box 1339, CB1 0BZ, Cambridge, UK
Tel. US +1 (202) 559 0658

First NIST PQC Standardization Workshop
Fort Lauderdale April 12, 2018
Key Encapsulation Mechanism (KEM) and Public Key Encryption

Following the NIST call [NI16] and Peikert [Pe14], our scheme is formalized as an IND-CPA Key Encapsulation Mechanism (KEM), consisting of three algorithms:

\[
(PK, SK) \leftarrow \text{KeyGen}().
\]

Generate a public key PK and a secret key SK.

\[
(CT, K) \leftarrow \text{Encaps}(PK).
\]

Encapsulate a (random) key K in ciphertext CT.

\[
K \leftarrow \text{Decaps}(SK, CT).
\]

Decapsulate shared key K from CT with SK.

In this model, reconciliation data is a part of ciphertext produced by Encaps(). The three KEM algorithms constitute a natural single-roundtrip key exchange:

<table>
<thead>
<tr>
<th>Alice</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td>((PK, SK) \leftarrow \text{KeyGen}())</td>
<td>(\text{PK} \rightarrow \text{CT})</td>
</tr>
<tr>
<td>(K \leftarrow \text{Decaps}(SK, CT))</td>
<td>((CT, K) \leftarrow \text{Encaps}(PK))</td>
</tr>
</tbody>
</table>

Thanks to its low failure rate \(< 2^{-128}\), due to novel reconciliation methods and error correction, HILA5 can also be used for public key encryption via (AEAD) Key Wrap.
Based on Ring-LWE (Learning with Errors in a Ring)

Let \( \mathcal{R} \) be a ring with elements \( v \in \mathbb{Z}_n \). We use fast NTT arithmetic in \( \mathbb{Z}_q[x]/(x^n + 1) \).

**Definition (Informal)**

With all distributions and computations in ring \( \mathcal{R} \), let \( s, e \) be elements randomly chosen from some non-uniform distribution \( \chi \), and \( g \) be a uniformly random public value. Determining \( s \) from \( (g, g \ast s + e) \) in ring \( \mathcal{R} \) is the (Normal Form Search) Ring Learning With Errors (RLWE\( _{\mathcal{R}, \chi} \)) problem.

Typically \( \chi \) is chosen so that each coefficient is a Discrete Gaussian or from some other “Bell-Shaped” distribution that is relatively tightly concentrated around zero.

The hardness of the problem is a function of \( n, q, \) and \( \chi \). HILA5 uses very fast and well-studied “New Hope” parameters: \( n = 1024, q = 3 \times 2^{12} + 1 = 12289, \chi = \Psi_{16} \).
Discrete Gaussian $D_{\sqrt{8}}$ and Binomial "bitcount" Distribution $\Psi_{16}$

Green bars are the probability mass of binomial distribution $P(X = x) = 2^{-32} \binom{32}{x+16}$. Blue line is the discrete Gaussian distribution $D_\sigma$ with deviation parameter $\sigma = \sqrt{8}$.

$$\rho_\sigma(x) \propto \exp\left(-\frac{x^2}{2\sigma^2}\right).$$  Very good approximation:  $$\rho_\sigma(x) \approx \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}.$$
Noisy Diffie-Hellman in a Ring

<table>
<thead>
<tr>
<th>Alice</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a \leftarrow \chi$</td>
<td>$b \leftarrow \chi$</td>
</tr>
<tr>
<td>$e \leftarrow \chi$</td>
<td>$e' \leftarrow \chi$</td>
</tr>
<tr>
<td>$A = g \ast a + e$</td>
<td>$B = g \ast b + e'$</td>
</tr>
<tr>
<td>$x = B \ast a$</td>
<td>$y = A \ast b$</td>
</tr>
</tbody>
</table>

Here $g$ is a uniform, public generator. By substituting variables in $A$ and $B$ we get

$$x = (g \ast b + e') \ast a = g \ast a \ast b + e' \ast a$$
$$y = (g \ast a + e) \ast b = g \ast a \ast b + e \ast b.$$ 

Because error terms are much smaller than the common term $g \ast a \ast b$ we have $x \approx y$. 
In **reconciliation**, we wish the holders of \(x\) and \(y\) (Alice and Bob, respectively) to arrive at exactly the same shared secret \(k\) with minimal communication \(c\).

In Peikert’s reconciliation [Pe14] Bob sends 1 “phase bit” \(c\) for each vector element. Since \(q\) is odd and cannot be evenly divided in half, a fresh random bit is needed to “smoothen” the divide. **New Hope’s** reconciliation of also needs random numbers.
As we don’t need $n = 1024$ bits, we can select “Safe Bits” away from the decision boundary in order to get unbiased secrets without using additional randomness.

We designed error correction codes to push the failure probability well under $2^{-128}$. 
Hey students! Pay attention in the coding theory classes!

I designed a linear block code, XE5, specifically for HILA5.

Security Requirement: Fast, constant-time implementatable.

After various considerations (SafeBits), ended up with a block size of 496 bits (256 message bits + 240 redundancy bits.)

Always corrects 5 random bit flips, more with high probability.

I first described similar constant-time error correction techniques (for TRUNC8) in:


https://eprint.iacr.org/2016/1058 (Original uploaded November 15, 2016)
There is a single point on p. 17 of the HILA5 specification which erroneously claims IND-CCA security. With (too) much speculation this was shown not to be correct in [BBLP17]. The original SAC 2017 academic paper never even mentions IND-CCA.

Furthermore even [BBLP17] itself clearly states that:

“We emphasize that our attack does not break the IND-CPA security of HILA5. If HILA5 were clearly labeled as aiming merely for IND-CPA security then our attack would merely be a cautionary note, showing the importance of not reusing keys.”

Creating an IND-CCA variant via Fujisaki–Okamoto transform is straightforward. I will propose such a variant, probably not very dissimilar to “HILA5FO” from [BBLP17].
What Distinguishes HILA5 from the Rest?

+ **It’s Very Fast and can do KEM and Public Key Encryption.** Only about 5% slower than fastest New Hope (CPA) implementation (Matching Ring-LWE parameters.) I’ll have to get better NTT code for the new version, my current NTT code sucks!

+ **Less randomness required.** Reconciliation method produces unbiased secrets without randomized smoothing; much less randomness is therefore required.

+ **HILA5 decryption doesn’t fail.** HILA5 has a failure rate well under $2^{-128}$. Non-negligible decryption failure rate is needed in public key encryption.

+ **Non-malleable.** Computation of the final shared secret in HILA5 KEM uses the full public key and ciphertext messages, thereby reinforcing non-malleability and making a class of adaptive attacks infeasible.

+ **Shorter messages.** Ciphertext messages are slightly smaller than New Hope’s.

+ **Patent free.** As the sender can “choose the message” (as in NEWHOPE-SIMPLE), Ding’s Ring-LWE key exchange patents less likely to be applicable.
Algorithm Purpose: Key Encapsulation and Public Key Encryption.
Underlying problem: Ring-LWE (New Hope: $n = 1024, q = 12289, \Psi_{16}$)
Public key size: 1824 Bytes (+32 Byte private key hash.)
Private key size: 1792 Bytes (640 Bytes compressed.)
Ciphertext size: 2012 Byte expansion (KEM) + payload + MAC.
Failure rate: $< 2^{-128}$, consistent with security level.
Classical security: $2^{256}$ (Category 5 - Equivalent to AES-256).
Quantum security: $2^{128}$ (Category 5 - Equivalent to AES-256).


Always get the latest code and specs at: [https://github.com/mjosaarinen/hila5](https://github.com/mjosaarinen/hila5)