FrodoKEM
A simple and conservative KEM from generic lattices

Erdem Alkim      Joppe W. Bos      Léo Ducas      Patrick Longa      Ilya Mironov
Michael Naehrig    Valeria Nikolaenko    Chris Peikert    Ananth Raghunathan    Douglas Stebila
FrodoKEM Recap (part I)

FrodoKEM is conservative yet practical

- Plain LWE: generic, algebraically unstructured lattices
  - Minimizes potential attack surface: no algebraic ring structure

- Cautious parameterization: ‘medium-sized’ errors conforming to a worst-case/average-case reduction
  - Narrower errors ⇒ smaller parameters, better efficiency

- Concrete parameters chosen according to ‘core-SVP’ methodology
  - Lower-bound the first-order exponential time and space of SVP
FrodoKEM Recap (part II)

FrodoKEM has a simple design and implementation

- Matrix-vector products over $\mathbb{Z}_q^n$ with a power-of-2 modulus $q$
- Straightforward error sampling: approximation to rounded Gaussian
  
  E.g., using inversion sampling:
  
  - Table $T_X$ stores $(s + 1)$ integers related to discrete cumulative distribution function
  - Given a random value $r$, determine smallest index $i$ such that $r \leq T_X[i]$
  - Output $(-1)^b i$ for a random bit $b$

- x64 implementation consists of $\sim 350$ lines of C code
  (+ existing symmetric primitives)

- No use of hand-written assembly: additional implementation only differs by use of vector intrinsics for computing $A S + E$ and $S' A + E'$
FrodoKEM Recap (part III)

- Two (2) variants
  - Uses either AES-128 or SHAKE128 for the generation of a public matrix $A$
- Six (6) parameter sets in total:
  - FrodoKEM-640-XXX: targets security level 1 ($\geq$ AES-128)
  - FrodoKEM-976-XXX: targets security level 3 ($\geq$ AES-192)
  - FrodoKEM-1344-XXX: targets security level 5 ($\geq$ AES-256)

Dimension $n \in \{640, 976, 1344\}$, $XXX \in \{AES, SHAKE\}$
List of updates for Round3
Encryption check during decapsulation is arguably the most fragile point of failure in the KEM structure
  • Failures are not detected by ‘positive’ tests

Guo et al., CRYPTO 2020: A key-recovery timing attack on post-quantum primitives using the Fujisaki-Okamoto transformation and its application on FrodoKEM
  • Exploits timing leakage during encryption check
KEM decapsulation in constant-time
A cautionary tale

- Writing constant-time code can be tricky
  - “Traditional” testing is insufficient

```c
int8_t ct_verify(const uint16_t *a, const uint16_t *b, size_t len)
{
    // Returns 0 if the byte arrays a and b are equal, -1 otherwise.

    uint16_t r = 0;
    for (size_t i = 0; i < len; i++)
        r |= a[i] ^ b[i];
    return (int8_t)((-int16_t)r >> 15);
}
```

\[ r = \left( -\left( \text{int16}_t \right) \left( r \gg 1 \right) \right) \gg 15; \]
KEM decapsulation in constant-time
A cautionary tale

What we have added to the code:

- New ‘negative’ tests against changes in ciphertext
- Macros that use Valgrind to check for non-constant time code
  - Selection is done at compilation time
- Tests using clang’s address sanitizer and undefined behavior sanitizer
- All these tests are now run automatically with GitHub Actions

https://github.com/microsoft/PQCrypto-LWEKE
Recent developments (part I)

- FrodoKEM, at levels 3 and 5, is **recommended by the German Federal Office for Information Security (BSI)** as cryptographically suitable for long-term confidentiality protection.

  “BSI – Technical Guideline (Cryptographic Mechanisms: Recommendations and Key Lengths)”, BSI TR-02102-1, March 2021:
  

- We wrote a Python3 reference implementation of FrodoKEM
  
  [https://github.com/microsoft/PQCrypto-LWEKE](https://github.com/microsoft/PQCrypto-LWEKE)
Recent developments (part II)

  - Part of HACL*, a formally verified cryptographic library written in F*

  - Shows a significant speedup (~15x) on FPGA using Trivium for the generation of the public matrix $A$
  - Shows that FrodoKEM incurs a negligible overhead when adding arithmetic masking to protect decapsulation against first-order side-channel attacks
Recent developments (part III)

  - Faster matrix multiplication using a row-wise blocking and packing (RWCF) approach
  - Speedups of 12%, 14% and 16% are achieved for FrodoKEM-640-AES, FrodoKEM-976-AES and FrodoKEM-1344-AES, resp.
Performance results

- Performance (in $10^3$ cycles) on an x64 AMD Ryzen 9 3900XT @3.8GHz (Bos et al. 2021)

<table>
<thead>
<tr>
<th>Parameter set</th>
<th>Level</th>
<th>keygen</th>
<th>encaps</th>
<th>decaps</th>
</tr>
</thead>
<tbody>
<tr>
<td>FrodoKEM-640-AES</td>
<td>1</td>
<td>903</td>
<td>1068</td>
<td>1025</td>
</tr>
<tr>
<td>FrodoKEM-976-AES</td>
<td>3</td>
<td>1712</td>
<td>1955</td>
<td>1850</td>
</tr>
<tr>
<td>FrodoKEM-1344-AES</td>
<td>5</td>
<td>3017</td>
<td>3363</td>
<td>3221</td>
</tr>
</tbody>
</table>

E.g., one full FrodoKEM execution (at level 1) is completed in **0.79 msec.**, Encaps + Decaps runs in **0.55 msec.**
FrodoKEM
A simple and conservative KEM from generic lattices

Erdem Alkim      Joppe W. Bos      Léo Ducas      Patrick Longa      Ilya Mironov
Michael Naehrig      Valeria Nikolaenko      Chris Peikert      Ananth Raghunathan      Douglas Stebila

https://frodokem.org/