Fast Ideal Lattice-Based KEMs
on ARM Cortex-M4

Matthias J. Kannwischer, Joost Rijneveld, Peter Schwabe
matthias@kannwischer.eu

September 03, 2018

School on Security & Correctness in the Internet of Things 2018
Graz, Austria
Motivation: Lattice-Based Crypto

- Shor’s algorithm (1994) breaks RSA, DLP, and ECDLP
  - We need post-quantum cryptography (public key cryptography resistant to quantum attacks)
- NIST is currently running a PQC competition (69 candidates)
- Lattice-based schemes are the most practical
  - Mostly very fast, acceptable key sizes
  - 28 NISTPQC candidates are based upon lattices
- Ideal Lattice-based crypto: arithmetic in a polynomial ring
  - 22 of 28 candidates
  - Runtime is dominated by polynomial multiplication
Our work: Optimising Lattice KEMs on M4

- We want:
  - Fast poly mul for all modulus $q$ and poly degree $n$
- If $q$ prime and $n$ power of 2 (e.g., NewHope, Kyber)
  - Use number theoretic transform (NTT)-based mul
- Problem: Many schemes don’t satisfy this
- This work
  - Restrict to power of 2 $q (\leq 2^{16})$
  - Allow arbitrary $n (\leq 1024)$
  - Generate fast Cortex-M4 assembly
  - Integrate into PQM4
    (https://github.com/mupq/pqm4)
  - Plug into the five applicable NIST submissions: KINDI, NTRU-HRSS, NTRU-KEM, RLizard, and SABER
Fast polynomial multiplication

- Schoolbook method
  - \( N^2 \) coefficient multiplications
- Karatsuba multiplication
  - 3 multiplications of \( N/2 \)
- Toom-3 multiplication
  - 5 multiplications of \( N/3 \)
- Toom-4 multiplication
  - 7 multiplications of \( N/4 \)
- Apply methods recursively until schoolbook is faster
- What is the fastest method for a given \( N \)?
Example: NTRU-HRSS $N = 701$
New Speed Records for Lattice KEMs on Cortex-M4

Work in progress! Might still improve considerably.

<table>
<thead>
<tr>
<th>scheme</th>
<th>params</th>
<th>impl</th>
<th>key gen</th>
<th>encaps</th>
<th>decaps</th>
</tr>
</thead>
<tbody>
<tr>
<td>KINDI</td>
<td>(n = 256)</td>
<td>ref</td>
<td>22,942k</td>
<td>29,656k</td>
<td>37,817k</td>
</tr>
<tr>
<td></td>
<td>(q = 2^{14})</td>
<td>ours</td>
<td>1,101k</td>
<td>1,494k</td>
<td>1,726k</td>
</tr>
<tr>
<td>NTRU-HRSS</td>
<td>(n = 701)</td>
<td>ref</td>
<td>204,854k</td>
<td>5,166k</td>
<td>15,067k</td>
</tr>
<tr>
<td></td>
<td>(q = 2^{13})</td>
<td>ours</td>
<td>164,090k</td>
<td>451k</td>
<td>917k</td>
</tr>
<tr>
<td>NTRU-KEM</td>
<td>(n = 743)</td>
<td>ref</td>
<td>53,326k</td>
<td>7,144k</td>
<td>12,782k</td>
</tr>
<tr>
<td></td>
<td>(q = 2^{11})</td>
<td>ours</td>
<td>5,445k</td>
<td>1,825k</td>
<td>2,145k</td>
</tr>
<tr>
<td>SABER</td>
<td>(n = 256)</td>
<td>ref</td>
<td>7,123k</td>
<td>9,471k</td>
<td>12,304k</td>
</tr>
<tr>
<td></td>
<td>(q = 2^{13})</td>
<td>[1]</td>
<td>1,147k</td>
<td>1,444k</td>
<td>1,543k</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ours</td>
<td>982k</td>
<td>1,277k</td>
<td>1,323k</td>
</tr>
<tr>
<td>RLizard</td>
<td>(n = 1024)</td>
<td>ref</td>
<td>26,428k</td>
<td>32,211k</td>
<td>57,344k</td>
</tr>
<tr>
<td></td>
<td>(q = 2^{11})</td>
<td>ours</td>
<td>626k</td>
<td>1,513k</td>
<td>1,986k</td>
</tr>
</tbody>
</table>