Three Bears round 2

updates

Mike Hamburg
Cryptographer

August 24, 2019
ThreeBears algorithm summary: similar to Kyber

Public key is seed to generate matrix $A$; $X := As + e$

Matrix $A$ is $2 \times 2$, $3 \times 3$ or $4 \times 4$ depending on security parameters

KEM header is $Y := s'A + e'$
KEM payload is $m$ masked by rounded digits of $s'X + e''$

Decrypt by calculating $Ys \approx s'As \approx s'X + e''$

Negligible failure probability

Fujisaki-Okamoto variant for CCA security
ThreeBears algorithm summary: innovations

Kyber: lattice is coefficients of polynomials mod $x^{256} + 1, q = 3329$

ThreeBears: lattice is digits of numbers mod $N = 2^{3120} − 2^{1560} − 1$
   Equivalently: coefficients of polynomials mod $x^{312} − x^{156} − 1, x = 2^{10}$

Lattices with bignum math instead of polynomial math
   Easy to pack digits efficiently
   Fast if CPU has a big multiplier
   No NTT, but ring has no zero divisors → better security proof

Slightly larger lattices than other systems → more conservative params
Constant-time 2-error-correcting code for better failure-vs-efficiency tradeoff
Changes in round 2: reduced variance in CCA versions

LWE design decision: more efficiency at the cost of rare failures
  Failure attacks possible, see eg D’Anvers-Vercauteren-Verbauwhede 2018
  DVV-style attacks considered in ThreeBears’ original 2017 design

ThreeBears’ error-correcting code makes analysis harder
  Can’t calculate exact failure probabilities

Round 2: more rigorous and conservative (over)estimates of failure probability
  Round 1 parameters are marginal vs $2^{64}$ queries
  Round 2: reduced variance, reduced fail prob $\rightarrow$ less risk of failure attack
  Slightly lower lattice security
Changes in round 2: reduced variance in CCA versions

<table>
<thead>
<tr>
<th>Param set</th>
<th>CCA secure</th>
<th>Ephemeral</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>variance</td>
<td>failure</td>
</tr>
<tr>
<td>BabyBear r1</td>
<td>5/8</td>
<td>$2^{-128}$</td>
</tr>
<tr>
<td>BabyBear r2</td>
<td>9/16</td>
<td>$2^{-156}$</td>
</tr>
<tr>
<td>MamaBear r1</td>
<td>1/2</td>
<td>$2^{-141}$</td>
</tr>
<tr>
<td>MamaBear r2</td>
<td>13/32</td>
<td>$2^{-206}$</td>
</tr>
<tr>
<td>PapaBear r1</td>
<td>3/8</td>
<td>$2^{-188}$</td>
</tr>
<tr>
<td>PapaBear r2</td>
<td>5/16</td>
<td>$2^{-256}$</td>
</tr>
</tbody>
</table>
Changes in round 2: implicit rejection in CCA versions

Initial submission: explicit rejection
   Supported by CCA security proof (for ThreeBears only)
   Simpler and faster
   Wanted to promote discussion about rejection modes

Since then, state of the art has settled on implicit rejection
   Better usability
   Encourages constant time
   Everyone can use same security analysis (see SXY, HKSU, BHHP, …)
Changes in round 2: implicit rejection in CCA versions

Optional in round 2 submission

Mandatory as of July 2019

- PRF key lengthened to 40 bytes, otherwise same
- Uses $U_m^L$, meaning ct isn’t hashed into key: faster and simpler
- [BHHP’19] says security equivalent to $U^L$ in the QROM

Software now constant-time, doesn’t return failure code

- Performance penalty: $\approx 10\%$ slower CCA decapsulation
Changes in round 2: new toys and challenges

Toy schemes intended to be broken:
  - GummyBear (new): dimension = 120; \( N \) not prime
  - TeddyBear: dimension = 240 (vs BabyBear: dimension = 624)

Challenges generated by cut+choose
  - All standard and toy bears, plus dimensions 80 ... 320 for granularity

Not intended to be broken: Koala and KoalaEphem
  - Could find use as lightweight ThreeBears variant
  - Dimension = 240 \( \cdot \) 2
  - Classical core-sieve difficulty 115 and 128 bits, resp.
Summary

ThreeBears is a competitive alternative for poly-LWE systems
  Uses bignum math instead of polynomial math; otherwise similar

Original design was to provoke more study of possible LWE variants

Round 2 changes make it more conservative
Questions?