Practical CCA2-Secure and Masked Ring-LWE Implementation

Tobias Oder\textsuperscript{1}, Tobias Schneider\textsuperscript{2}, Thomas Pöppelmann\textsuperscript{3}, Tim Güneysu\textsuperscript{1,4}

\textsuperscript{1}Ruhr-University Bochum, \textsuperscript{2}Université Catholique de Louvain, \textsuperscript{3}Infineon Technologies AG, \textsuperscript{4}DFKI

CHES 2018

10.09.2018
Motivation
Ring-LWE

• NIST post-quantum standardization project
• Various NIST submissions are based on Ring-LWE including
  – NewHope
  – LIMA
  – (Kyber)
  – ...

CCA2-Secure and Masked Ring-LWE | Tobias Oder | Ruhr-University Bochum | 10.09.2018
• NIST post-quantum standardization project
• Various NIST submissions are based on Ring-LWE including
  – NewHope
  – LIMA
  – (Kyber)
  – ...

Previous work


CCA2-Security

- Plain Ring-LWE encryption is only secure against chosen-plaintext attackers (CPA)

- Many use cases require security against chosen-ciphertext attackers (CCA)

- Generic Fujisaki-Okamoto transform
  - Assumes negligible decryption error
  - Tweak by Targhi and Unruh for post-quantum security [TU16]
  - Expensive re-encryption in decryption

CCA2-Security

CCA2-secure Decryption
CCA2-Security

CCA2-secure Decryption

\[ \text{RLWE.CPA}_{\text{dec}}^{\text{NTT}} \]

\[ \tilde{c}_1 \]

\[ \tilde{r}_2 \]

\[ c_2 \]

\[ c_3 \]

\[ m_{\text{cpa}} \]

\[ \text{RLWE.Enc} \]

\[ \text{msg} \]

\[ H' \]

\[ c_4^* \]

\[ \tilde{c}_1^* \]

\[ c_2^* \]

\[ \text{RLWE.CPA}_{\text{enc}}^{\text{NTT}} \]

\[ \tilde{a} \]

\[ \text{PRNG init} \]

\[ 256 \text{ bits} \]

\[ m_{\text{cca}} \]

\[ \text{fail} \]

\[ \text{else} \]
CCA2-Security

CCA2-secure Decryption
CCA2-Security

CCA2-secure Decryption
Contribution
Embedded Implementation

- Our contribution:
  CCA2-secure first-order masked Ring-LWE implementation
Embedded Implementation

• Our contribution:
  
  CCA2-secure first-order masked Ring-LWE implementation

• Target platform ARM Cortex-M4
  – Constrained computing capabilities/memory
Embedded Implementation

• Our contribution:
  CCA2-secure first-order masked Ring-LWE implementation

• Target platform ARM Cortex-M4
  – Constrained computing capabilities/memory

• Secret-independent execution time as countermeasure against timing attacks

• Masking as countermeasure against Differential Power Analysis
  – Boolean vs. arithmetic
Components to be masked in CCA2-secure Ring-LWE

- PRNG/Hash
- NTT
  - Polynomial multiplication
- Binomial sampler (BS)
- Encoding/Decoding

Ring-LWE CPA Encryption

Ring-LWE CPA Decryption
Components to be masked in CCA2-secure Ring-LWE

- PRNG/Hash $\rightarrow$ [BDPVA10]

- NTT $\rightarrow$ straight-forward
  - Polynomial multiplication

- Binomial sampler (BS)

- Encoding/Decoding

Masking Ring-LWE

Components to be masked in CCA2-secure Ring-LWE

- PRNG/Hash $\rightarrow$ [BDPVA10]
- NTT $\rightarrow$ straight-forward
  - Polynomial multiplication
- Binomial Sampler (BS)
- Encoding/Decoding

Encoding
Masked encoding

- *Encoding* transforms a bit string into a polynomial
  
  - Without masking:
    
    \[
    coeff = bit \cdot \left\lfloor \frac{q}{2} \right\rfloor
    \]
Masked encoding

- **Encoding** transforms a bit string into a polynomial
  - Without masking:
    \[
    \text{coeff} = \text{bit} \cdot \left\lfloor \frac{q}{2} \right\rfloor
    \]
  - With \( \text{bit}' \oplus \text{bit}'' = \text{bit} \):
    \[
    \text{coeff}' = \text{bit}' \cdot \left\lfloor \frac{q}{2} \right\rfloor
    \]
    \[
    \text{coeff}'' = \text{bit}'' \cdot \left\lfloor \frac{q}{2} \right\rfloor
    \]
Masked encoding

- **Encoding** transforms a bit string into a polynomial
  - Without masking:
    \[ \text{coeff} = \text{bit} \cdot \left\lfloor \frac{q}{2} \right\rfloor \]
  - With \( \text{bit}' \oplus \text{bit}'' = \text{bit} \):
    \[ \text{coeff}' = \text{bit}' \cdot \left\lfloor \frac{q}{2} \right\rfloor \]
    \[ \text{coeff}'' = \text{bit}'' \cdot \left\lfloor \frac{q}{2} \right\rfloor \]

- \( q \) is odd \( \Rightarrow \left\lfloor \frac{q}{2} \right\rfloor + \left\lfloor \frac{q}{2} \right\rfloor \neq q \)

**Problem:** Result is off by one if \( \text{bit}' = 1 \) and \( \text{bit}'' = 1 \)
Solution: Add $bit' \cdot bit''$ to the result

- Compute $bit' \cdot bit''$ by splitting into subshares

\[
(bit'(1) + bit'(2)) \cdot (bit''(1) + bit''(2))
\]

\[
= bit'(1) \cdot bit''(1) + bit'(1) \cdot bit''(2) + bit'(2) \cdot bit''(1) + bit'(2) \cdot bit''(2)
\]

- Use fresh randomness to securely sum the cross-products
Decoding
**Masked decoding**

**Input:** Coefficient $\in [0, q - 1]$

**Output:** Decoded bit

**Idea:**
- Shift distribution of coefficients
- Apply arithmetic-to-Boolean conversion
- Extract sign bit
Masked decoding

**Input:** Coefficient $\in [0, q - 1]$

**Output:** Decoded bit

**Idea:**
- Shift distribution of coefficients
- Apply arithmetic-to-Boolean conversion
- Extract sign bit
Masked decoding

**Input:** Coefficient ∈ [0, q − 1]

**Output:** Decoded bit

**Idea:**
- Shift distribution of coefficients
- Apply arithmetic-to-Boolean conversion
- Extract sign bit
Masked decoding

**Input:** Coefficient $\in [0, q - 1]$

**Output:** Decoded bit

**Idea:**
- Shift distribution of coefficients
- Apply arithmetic-to-Boolean conversion
- Extract sign bit
Binomial Sampler
Masked sampler

- **Input**: Boolean shares; **Output**: Arithmetic shares

- Count Hamming weight as
  \[
  \sum_{i=0}^{7} (bit'(i) \oplus bit''(i)) = \sum_{i=0}^{7} bit'(i) + bit''(i) - 2bit'(i)bit''(i)
  \]

- Compute \(bit'(i) \cdot bit''(i)\) by splitting into subshares
Results
Side-Channel Evaluation

T-test evaluation of the decoding (example)

- *Blue*: first-order evaluation
- *Dashed red*: second-order evaluation
Cortex-M4 Performance

- Dimension $n = 1024$
- Modulus $q = 12289$
- Standard deviation $\varsigma = 2$

<table>
<thead>
<tr>
<th>Operation</th>
<th>Cycle Counts</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unmasked</td>
</tr>
<tr>
<td>Key Generation</td>
<td>2,669,559</td>
</tr>
<tr>
<td>CCA2-secured Encryption</td>
<td>4,176,684</td>
</tr>
<tr>
<td>CCA2-secured Decryption</td>
<td>4,416,918</td>
</tr>
<tr>
<td>CPA-RLWE Encryption</td>
<td>3,910,871</td>
</tr>
<tr>
<td>CPA-RLWE Decryption</td>
<td>163,887</td>
</tr>
<tr>
<td>Shake-128</td>
<td>87,738</td>
</tr>
<tr>
<td>NTT</td>
<td>83,906</td>
</tr>
<tr>
<td>INTT</td>
<td>104,010</td>
</tr>
<tr>
<td>Uniform Sampling (TRNG)</td>
<td>60,014</td>
</tr>
<tr>
<td>SampleNoisePoly (PRNG)</td>
<td>1,142,448</td>
</tr>
<tr>
<td>PRNG (64 bytes)</td>
<td>88,778</td>
</tr>
</tbody>
</table>
## Cortex-M4 Performance

- Dimension $n = 1024$
- Modulus $q = 12289$
- Standard deviation $\varsigma = 2$

<table>
<thead>
<tr>
<th>Operation</th>
<th>Cycle Counts</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unmasked</td>
<td>Masked</td>
</tr>
<tr>
<td>Key Generation</td>
<td>2,669,559</td>
<td>-</td>
</tr>
<tr>
<td>CCA2-secured Encryption</td>
<td>4,176,684</td>
<td>-</td>
</tr>
<tr>
<td><strong>CCA2-secured Decryption</strong></td>
<td>4,416,918</td>
<td>25,334,493</td>
</tr>
<tr>
<td>CPA-RLWE Encryption</td>
<td>3,910,871</td>
<td>19,315,432</td>
</tr>
<tr>
<td>CPA-RLWE Decryption</td>
<td>163,887</td>
<td>550,038</td>
</tr>
<tr>
<td>Shake-128</td>
<td>87,738</td>
<td>201,997</td>
</tr>
<tr>
<td>NTT</td>
<td>83,906</td>
<td>-</td>
</tr>
<tr>
<td>INTT</td>
<td>104,010</td>
<td>-</td>
</tr>
<tr>
<td>Uniform Sampling (TRNG)</td>
<td>60,014</td>
<td>-</td>
</tr>
<tr>
<td>SampleNoisePoly (PRNG)</td>
<td>1,142,448</td>
<td>6,031,463</td>
</tr>
<tr>
<td>PRNG (64 bytes)</td>
<td>88,778</td>
<td>202,454</td>
</tr>
</tbody>
</table>
Cortex-M4 Performance

- Dimension $n = 1024$
- Modulus $q = 12289$
- Standard deviation $\varsigma = 2$

<table>
<thead>
<tr>
<th>Operation</th>
<th>Cycle Counts</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unmasked</td>
</tr>
<tr>
<td>Key Generation</td>
<td>2,669,559</td>
</tr>
<tr>
<td>CCA2-secured Encryption</td>
<td>4,176,684</td>
</tr>
<tr>
<td>CCA2-secured Decryption</td>
<td>4,416,918</td>
</tr>
<tr>
<td>CPA-RLWE Encryption</td>
<td>3,910,871</td>
</tr>
<tr>
<td>CPA-RLWE Decryption</td>
<td>163,887</td>
</tr>
<tr>
<td>Shake-128</td>
<td>87,738</td>
</tr>
<tr>
<td>NTT</td>
<td>83,906</td>
</tr>
<tr>
<td>INTT</td>
<td>104,010</td>
</tr>
<tr>
<td>Uniform Sampling (TRNG)</td>
<td>60,014</td>
</tr>
<tr>
<td>SampleNoisePoly (PRNG)</td>
<td>1,142,448</td>
</tr>
<tr>
<td>PRNG (64 bytes)</td>
<td>88,778</td>
</tr>
</tbody>
</table>
Cortex-M4 Performance

- Dimension $n = 1024$
- Modulus $q = 12289$
- Standard deviation $\varsigma = 2$

<table>
<thead>
<tr>
<th>Operation</th>
<th>Cycle Counts Unmasked</th>
<th>Cycle Counts Masked</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Generation</td>
<td>2,669,559</td>
<td>-</td>
</tr>
<tr>
<td>CCA2-secured Encryption</td>
<td>4,176,684</td>
<td>-</td>
</tr>
<tr>
<td>CCA2-secured Decryption</td>
<td>4,416,918</td>
<td>25,334,493</td>
</tr>
<tr>
<td>CPA-RLWE Encryption</td>
<td>3,910,871</td>
<td>19,315,432</td>
</tr>
<tr>
<td>CPA-RLWE Decryption</td>
<td>163,887</td>
<td>550,038</td>
</tr>
<tr>
<td>Shake-128</td>
<td>87,738</td>
<td>201,997</td>
</tr>
<tr>
<td>NTT</td>
<td>83,906</td>
<td>-</td>
</tr>
<tr>
<td>INTT</td>
<td>104,010</td>
<td>-</td>
</tr>
<tr>
<td>Uniform Sampling (TRNG)</td>
<td>60,014</td>
<td>-</td>
</tr>
<tr>
<td>SampleNoisePoly (PRNG)</td>
<td>1,142,448</td>
<td>6,031,463</td>
</tr>
<tr>
<td>PRNG (64 bytes)</td>
<td>88,778</td>
<td>202,454</td>
</tr>
</tbody>
</table>
Conclusion

• First masking of a Ring-LWE-based scheme that covers CCA2-security with first-order proof

• New masked encoder & decoder

• New masked sampler

• *Future work*: Higher-order masking
Thank You For Your Attention!