Topics in Post-Quantum Cryptography

Andreas Hülsing
State of affairs

• Standards track
  • Stateful hash-based signatures: XMSS, LMS (Internet drafts)
  • NTRUEncrypt (IEEE Std 1363.1, X9.98)
• Hundreds of proposed schemes
Initial recommendations

- **Symmetric encryption** Thoroughly analyzed, 256-bit keys:
  - AES-256
  - Salsa20 with a 256-bit key

Evaluating: Serpent-256, ...

- **Symmetric authentication** Information-theoretic MACs:
  - GCM using a 96-bit nonce and a 128-bit authenticator
  - Poly1305

- **Public-key encryption** McEliece with binary Goppa codes:
  - length $n = 6960$, dimension $k = 5413$, $t = 119$ errors

Evaluating: QC-MDPC, Stehlé-Steinfeld NTRU, ...

- **Public-key signatures** Hash-based (minimal assumptions):
  - XMSS with any of the parameters specified in CFRG draft
  - SPHINCS-256

Evaluating: HFEv-, ...
„Official“ developments

• Feb `13: First PQC draft in IRTF´s CFRG
• Sep `13: ETSI holds first PQC WS (afterwards annually)
• April `15: NIST holds conference on PQC
• Aug `15: NSA announces transition to PQC
• Feb `16: NIST announces ´PQC competition´
• Dec `16: NIST opens call for proposals

Scheduled:
• Nov `17: NIST submission deadline
• 2024: „Draft standards ready“ (NIST, Feb `16)
NIST Competition

NEWS -- December 15, 2016: The National Institute of Standards and Technology (NIST) is now accepting submissions for quantum-resistant public-key cryptographic algorithms. The deadline for submission is November 30, 2017. Please see the Post-Quantum Cryptography Standardization menu at left for the complete submission requirements and evaluation criteria.

In recent years, there has been a substantial amount of research on quantum computers – machines that exploit quantum mechanical phenomena to solve mathematical problems that are difficult or intractable for conventional computers. If large-scale quantum computers are ever built, they will be able to break many of the public-key cryptosystems currently in use. This would seriously compromise
NIST Competition

• Selection of
  • Digital signature and
  • Public key encryption / Key exchange

• Probably > 100 submissions

• No single winner

• Classically this will spark interest in cryptanalysis
Up next
(Quantum) security
Shor’s algorithm (1994)

- Quantum computers can do FFT very efficiently
- Can be used to find period of a function
- This can be exploited to factor efficiently (RSA)
- Shor also shows how to solve discrete log efficiently (DSA, DH, ECDSA, ECDH)
Grover’s algorithm (1996)

- Quantum computers can search $N$ entry DB in $\Theta(\sqrt{N})$
- Application to symmetric crypto
- Nice: Grover is provably optimal (For random function)
- Double security parameter.
(Quantum) security

• Are attacks using Grover efficient?
• Is Grover speed-up the only thing we can get?
  • Currently working to prove this for hash functions (under certain assumptions)
• Are the PQ problems classically secure?
• What is the exact security?

• We never had „provably secure crypto“
  • Can we classically break RSA? Who knows!
Results

- Sponges are quantum collision-resistant if block function is random function or random one-way permutation (does not cover SHA3!)
Quantum Cryptography
Why not beat ‘em with their own weapons?

- QKD: Quantum Key distribution.
  - Based on some nice quantum properties: entanglement & collapsing measurements
  - Information theoretic security (at least in theory)
    -> Great!
  - For sale today!
- So why don’t we use this?
- Only short distance, point-to-point connections!
  - Internet? No way!
- Longer distances require „trusted-repeaters“ 😊
  - We all know where this leads...
Implementation security
Side-channels

• Implementations might leak secret information through
  • timing,
  • cache-access patterns,
  • electro-magnetic radiation,
  • power consumption...

• Not covered by standard security models.
Implementation security

• Still hard for traditional schemes
• New PQ Problems come with new basic operations
• Not much research yet (for PQC)
• But a lot of experience
Discrete Gaussians

• Basic building block in lattice-based cryptography.
• Used to “hide” secret.
• Unknown how to implement efficiently in constant time.
Results

• Attack on BLISS [DDLL’13], implemented in StrongSwan library.
• Practical cache attack on both implemented samplers.
• First algorithm to “un-hide” secret key given side-channel information for Gaussian noise.
• Can compute secret key after < 5000 signatures.

Ongoing:
• Solution: Allow for constant-time sampler by changing the distribution.
Integration
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fish-1-316</td>
<td>0.01</td>
<td>364.11</td>
<td>201.17</td>
<td>32</td>
<td>64</td>
<td>108013</td>
<td>ROM</td>
</tr>
<tr>
<td>Fish-10-38</td>
<td>0.01</td>
<td>29.73</td>
<td>17.46</td>
<td>32</td>
<td>64</td>
<td>118525</td>
<td>ROM</td>
</tr>
<tr>
<td>Fish-42-14</td>
<td>0.01</td>
<td>13.27</td>
<td>7.45</td>
<td>32</td>
<td>64</td>
<td>152689</td>
<td>ROM</td>
</tr>
<tr>
<td>Picnic-10-38</td>
<td>0.01</td>
<td>31.31</td>
<td>16.30</td>
<td>32</td>
<td>64</td>
<td>195458</td>
<td>QROM</td>
</tr>
<tr>
<td>MQ 5pass</td>
<td>0.96</td>
<td>7.21</td>
<td>5.17</td>
<td>32</td>
<td>74</td>
<td>40952</td>
<td>ROM</td>
</tr>
<tr>
<td>SPHINCS-256</td>
<td>0.82</td>
<td>13.44</td>
<td>0.58</td>
<td>1088</td>
<td>1056</td>
<td>41000</td>
<td>SM</td>
</tr>
<tr>
<td>BLISS-I</td>
<td>44.16</td>
<td>0.12</td>
<td>0.02</td>
<td>2048</td>
<td>7168</td>
<td>5732</td>
<td>ROM</td>
</tr>
<tr>
<td>Ring-TESLA*</td>
<td>16k</td>
<td>0.06</td>
<td>0.03</td>
<td>12288</td>
<td>8192</td>
<td>1568</td>
<td>ROM</td>
</tr>
<tr>
<td>TESLA-768</td>
<td>48k</td>
<td>0.65</td>
<td>0.36</td>
<td>3216k</td>
<td>4128k</td>
<td>2336</td>
<td>(Q)ROM</td>
</tr>
<tr>
<td>FS-Véron</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>32</td>
<td>160</td>
<td>129024</td>
<td>ROM</td>
</tr>
<tr>
<td>SIDHp751</td>
<td>16.41</td>
<td>7.3k</td>
<td>5.0k</td>
<td>48</td>
<td>768</td>
<td>141312</td>
<td>QROM</td>
</tr>
</tbody>
</table>
Integration

• Smaller but less conservative signature choices exist
• PKE / KEX: Sizes better
• Can your protocol fit a 40KB public key / signature?
• How to deal with immaturity of PQ Problems?
  • Combiners -> pay in size / speed
Conclusion

• A lot of important questions ahead
  • Strengthen confidence
  • Secure implementations
• All solvable but need time & money
• Might have to rethink existing protocols
  • Will not get MUCH smaller
Thank you!

Questions?