SPHINCS+
Lessons learned

Andreas Hülsing,
Eindhoven University of Technology

RWPQC 2023
Take-away #1: It's a team effort!

Jean-Philippe Aumasson, Daniel J. Bernstein, Ward Beullens, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Andreas Hülsing, Panos Kampanakis, Stefan Kölbl, Mike Kudinov, Tanja Lange, Martin M. Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe, Bas Westerbaan

----------

19 People!
Hash-based signatures
(Merkle ‘89)

Boring crypto:
• Dates back to beginning of public key cryptography
• No fancy new mathematical assumption: Only requires a secure hash function („minimal security assumptions“)
• Stateful schemes are first PQ-signatures standardized (LMS & XMSS)
Signatures & Hash Functions
One-time signatures (Lamport’76)

(1-bit)
SPHINCS (Eurocrypt 2015)

Joint work with Daniel J. Bernstein, Daira Hopwood, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-O’Hearn

https://sphincs.org/
SPHINCS(+) Design Criteria

- Stateless
- Practical performance
- Conservative security
  - Collision resilience
  - $n$-bit hash == $n$-bit classical security
    (n/2-bit quantum security)

29 Nov 2022
https://sphincs.org/
How to go stateless (from an OTS)

Security parameter k
1. Generate $2^{2k}$ OTS key pairs
2. Authenticate all OTS public keys
3. Sign message with random OTS
4. Sig is OTS sig + authentication information
Merkle Tree [Merkle’79]
Certification Tree [Merkle’87]
(for 2-time signature)

→ = Certification (Signature on PK)

Diagram:

```
PK = PK
   /    
PK    PK
  /     /
PK    PK
 /     /
PK    PK
   /     /
PK    PK
  /     /
PK    PK
/     /
PK    PK
     /     /
      PK    PK
               /     /
                PK    PK
```
Stateless hash-based signatures \[NY89,\text{Gol}87,\text{Gol}04\]

Goldreich's approach [Gol04]:

Security parameter $k = 128$

- Use binary certification tree with OTS
- Key pairs are generated pseudorandomly
- Requires huge tree to avoid collisions (height 256)

Ok speed but **HUGE** signatures
SPHINCS $[\text{BHH}^+15]$ 

• Select index (pseudo-)randomly
• Mix both methods: Use a certification tree of Merkle trees
• Use a few-time signature key-pair on leaves to sign messages
  • Few index collisions allowed
  • Allows to reduce tree height ($\pm 64$)
SPHINCS$^+$ vs SPHINCS

- Allow for $2^{64}$ instead of $2^{50}$ signatures per key pair
- Add multi-target attack mitigation (Tweakable hash functions)
- “Simple” and “Robust” parameters
- New few-time signature scheme FORS
- Verifiable index selection
- Optional non-deterministic signatures

https://sphincs.org/
## Sizes

<table>
<thead>
<tr>
<th></th>
<th>public key size</th>
<th>secret key size</th>
<th>signature size</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPHINCS$^+$-128s</td>
<td>32</td>
<td>64</td>
<td>7 856</td>
</tr>
<tr>
<td>SPHINCS$^+$-128f</td>
<td>32</td>
<td>64</td>
<td>17 088</td>
</tr>
<tr>
<td>SPHINCS$^+$-192s</td>
<td>48</td>
<td>96</td>
<td>16 224</td>
</tr>
<tr>
<td>SPHINCS$^+$-192f</td>
<td>48</td>
<td>96</td>
<td>35 664</td>
</tr>
<tr>
<td>SPHINCS$^+$-256s</td>
<td>64</td>
<td>128</td>
<td>29 792</td>
</tr>
<tr>
<td>SPHINCS$^+$-256f</td>
<td>64</td>
<td>128</td>
<td>49 856</td>
</tr>
</tbody>
</table>

Table 8: Key and signature sizes in bytes
## Speed
(on single core of 3Ghz CPU)

| Algorithm                     | Sign  | Verify | |sig| |
|-------------------------------|-------|--------|------|
| SPHINCS+ -SHA2-128s-simple    | ~ 214 ms | ~ 0.28 ms | 7856 byte |
| SPHINCS+ -SHA2-128f-simple    | ~ 11 ms | ~ 0.72 ms | 17088 byte |
| SPHINCS+ -SHA2-192s-simple    | ~ 415 ms | ~ 0.48 ms | 16224 byte |
| SPHINCS+ -SHA2-192f-simple    | ~ 18 ms | ~ 1.17 ms | 35664 byte |
Take-away #2: Avoid splits between implementation and proofs

- Avoid "scheme implemented ≠ scheme analyzed"
- Positive example: Tweakable hash functions
Take-away #3: Proofs are tough!
(To write AND to read)

• Most conservative scheme? (Tight) proof was wrong!

• New proof step verified in EasyCrypt.
Take-away #4: Proofs stop at some level!
(And SHA2 is a bad RO)

• Morgan Stern and John Kelsey: 
  \textit{MGF1-SHA2-256 does not give you level V security}

• Sydney Anotonov: 
  \textit{SHA2-256 does not achieve perfect multi-target mitigation}

Reason?

\textit{Inner state collisions too easy to find.}

\textit{-> SHA2 is not a good random oracle!}
Take-away #5: You are never done

- There are always new ideas / insights!
- See SPHINCS+C
- After (lacking) feedback, we suggest to not implement SPHINCS+C
- We encourage NIST to standardize a low #sig version (in a different SP – maybe the one for stateful schemes?)
- Next important topic: Do we allow pre-hashing? If so, how? (see discussions e.g. in CFRG)
Lower $q_{\text{sign}}$? [Kölbel, "A note on SPHINCS+ parameter sets". https://eprint.iacr.org/2022/1725]

• "NIST asks for public feedback on a version of SPHINCS+ with a lower number of maximum signatures."
Lower $q_{\text{sign}}$?

- Note to come to ePrint soon.
- **Factor > 2 size reduction** (for $2^{20}$ sigs)!
- Results for NIST level I security -> Interest in higher levels?
- What applications would benefit?
- What would be the number of expected signatures?
- Does the reduced size / better speed make a fundamental difference?
Conclusion

• The most conservative selected signature scheme.
• No size & speed records, but for many applications...
  (e.g., code-signing, email & document signatures, etc.)
  • ... size is negligible compared to data, and
  • ... runtime is not that critical
  • ... (long-term) security is of utmost importance
• Possible synergies with stateful hash-based signatures
Thank you!
Questions?