SPHINCS+  
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
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

File $\rightarrow$ Hash $\rightarrow$ Sign $\rightarrow$ Signature

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

- Stateless
- Practical performance
- Conservative security
  - Collision resilience
  - n-bit hash == n-bit classical security
    (n/2-bit quantum security)
How to go stateless (from an OTS)

Security parameter $k$
1. Generate $2^k$ 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)

\[ \Rightarrow \text{ = Certification (Signature on PK)} \]
Stateless hash-based signatures
[NY89,Gol87,Gol04]

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 ($\pm 64$)

- \[ \text{tree height} \leq \log t \]
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
## Sizes

<table>
<thead>
<tr>
<th></th>
<th>sec</th>
<th>public key size</th>
<th>secret key size</th>
<th>signature size</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPHINCS⁺-128s</td>
<td>1</td>
<td>32</td>
<td>64</td>
<td>7856</td>
</tr>
<tr>
<td>SPHINCS⁺-128f</td>
<td>1</td>
<td>32</td>
<td>64</td>
<td>17088</td>
</tr>
<tr>
<td>SPHINCS⁺-192s</td>
<td>3</td>
<td>48</td>
<td>96</td>
<td>16224</td>
</tr>
<tr>
<td>SPHINCS⁺-192f</td>
<td>3</td>
<td>48</td>
<td>96</td>
<td>35664</td>
</tr>
<tr>
<td>SPHINCS⁺-256s</td>
<td>5</td>
<td>64</td>
<td>128</td>
<td>29792</td>
</tr>
<tr>
<td>SPHINCS⁺-256f</td>
<td>5</td>
<td>64</td>
<td>128</td>
<td>49856</td>
</tr>
</tbody>
</table>

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

|                                      | 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   |
News

• Fixed tight security proof.
  • Main technical part of proof already verified in EasyCrypt

• Implementation on Arm Cortex-{A55,A510,A78,A710,X1,X2}.

• Talk next session: Qian Wang. Optimization for SPHINCS+ using Intel® Secure Hash Algorithm Extensions

• ...

29 Nov 2022
https://sphincs.org/
We want feedback!
Lower $q_{\text{sign}}$?

- "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?
SPHINCS+C?

• No Spoilers! Watch next talk 😊
• What applications would benefit?
• What would be the number of expected signatures?
• Does the reduced size / better speed make a fundamental difference?
• How relevant is constant signing time
  (not be confused with constant time implementation)
• In short: Would the change be justified?
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
• PLEASE give us your feedback
Thank you!
Questions?