HARAKA

Efficient short-input hashing for post-quantum signature schemes

Stefan Kölbl[1] Martin M. Lauridsen[1] Florian Mendel[2] Christian Rechberger[1,2]

June 28th, 2016

[1]DTU Compute, Technical University of Denmark, Denmark

[2]IAIK, Graz University of Technology, Austria

DIGITAL SIGNATURES

Applications

- Electronic Signature
- Authentication / Integrity / Non-repudiation
- Code signing etc.

Popular schemes

- RSA
- DSA resp. ECDSA
- ElGamal
- Rabin
- ...

DIGITAL SIGNATURES

Applications

- Electronic Signature
- Authentication / Integrity / Non-repudiation
- Code signing etc.

Popular schemes

- RSA
- DSA resp. ECDSA
- ElGamal
- Rabin
- ...

Problem

Most widely used schemes are all broken by a quantum computer!

Hash-based signatures

Candidates for post-quantum secure signature schemes

- Code-based
- Lattice-based and Multivariate crypto
- Hash-based
  - Perhaps the most mature, security relies only on secure hash functions

HASH-BASED SIGNATURES

What is a signature scheme?

- Key generation: Generate key pair (X, Y)
- Signing: $\sigma = \text{sign}(m, X)$
- Verification: $\text{verify}(m, \sigma, Y) = \text{true}$

Security:

- **Total break**: Recover the signing key.
- **Universal forgery**: Forge signature for any message.
- **Selective forgery**: Forge signature on a message of the adversary's choice.
- **Existential forgery**: Find valid message/signature pair not already known to the adversary.

HASH-BASED SIGNATURES

Simplest hash-based signature scheme

- Key Generation: Pick two random values x and y.
- Public key: $Y = (H(x), H(y))$.
- Signature: If message is 0 then $\sigma = x$, otherwise $\sigma = y$.
- Verify: Check if $H(x)$ resp. $H(y)$ is in Y.

Hash-based signatures

Improvements for OTS

- W-OTS+ [Hül13]
  - Drops the requirement for collision-resistant hash function

State-full signature schemes

- CMSS [BGD+06]
- XMSS [BDH11]

State-less signature schemes

- Goldreich [Gol04]
- SPHINCS [BHH+15]

HASH-BASED SIGNATURES

Performance of hash-based signature schemes

- Many calls to the hash function...
- ...but using short input only.

Example SPHINCS:

- Provides 128-bit post-quantum security.
- Signing takes roughly 500.000 hash function evaluations.

CRYPTOGRAPHIC HASH FUNCTIONS

	Haswell			Skylake		
Primitive	Comp.	64-byte	4-KiB	Comp.	64-byte	4-KiB
Haraka-512/256	**1.77**	—	—	**0.95**	—	—
SHA-256	14.97	28.75	12.75	12.56	19.47	7.80
Keccak[$c = 512$]	27.94	24.28	10.51	19.67	20.31	9.22

If we want to speedup SPHINCS we need

- Efficient $H : \mathbb{F}_2^{512} \to \mathbb{F}_2^{256}$
- Efficient $F : \mathbb{F}_2^{256} \to \mathbb{F}_2^{256}$
- Only (second)-preimage resistance required

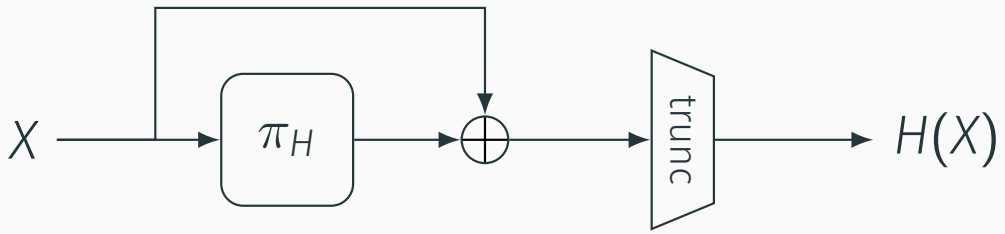
Solution

Design hash function specifically for this application.

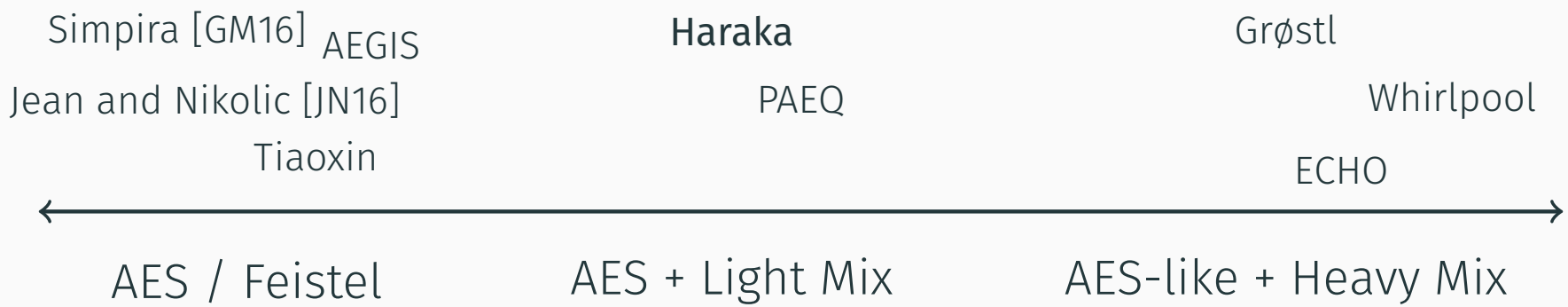
HARAKA

Our design Haraka (originally presented at PQ Meeting in Paris in 11/2015)

- Provides both $H : \mathbb{F}_2^{512} \rightarrow \mathbb{F}_2^{256}$ and $F : \mathbb{F}_2^{256} \rightarrow \mathbb{F}_2^{256}$.
- Davies-Meyer with 0 key.
- Very efficient permutation π_H using AES-NI.



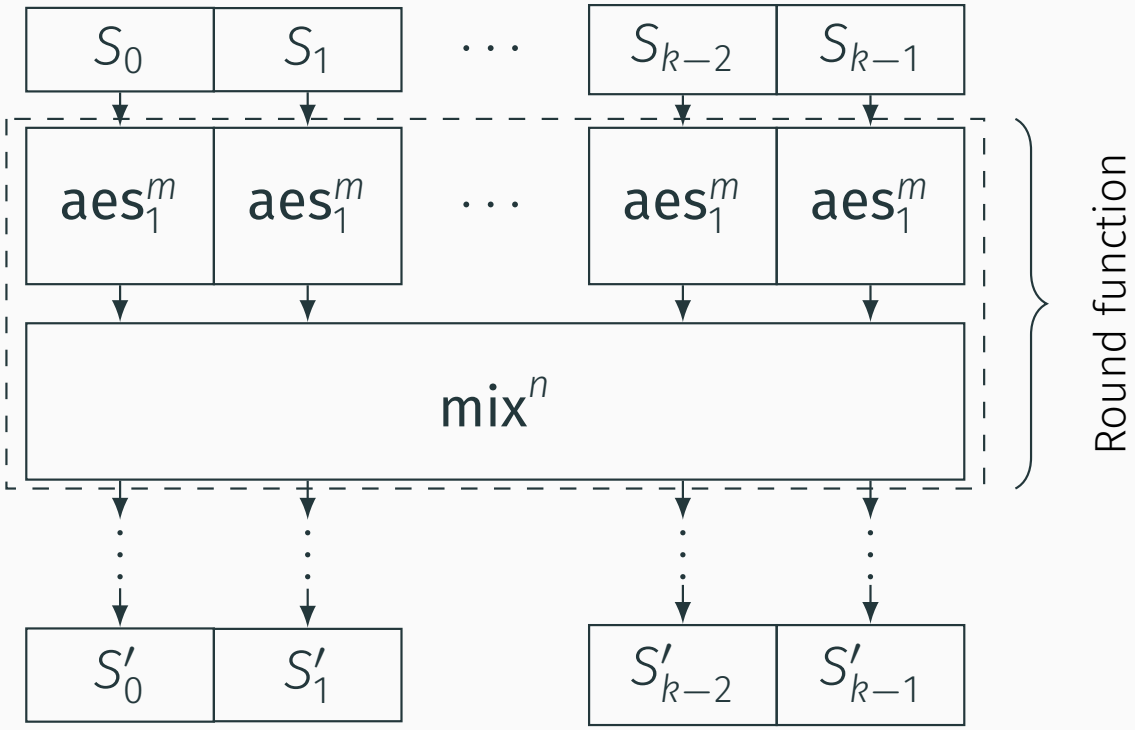
Design-space for AES-like permutations and primitives



- Popular both in SHA-3 and CAESAR competition.
- Wide-range of trade offs.
- Costs of linear layer can be hidden in latency of AES instructions.

Internal permutation of Haraka with 512-bit state

- SPN Network
- Round function: $\mathbf{mix} \circ \mathbf{aes}^m$



HARAKA

Design choices for π_H

- How many AES rounds m?
- How to mix the states?
- How many total rounds?
- Addition of constants[1]
- ...

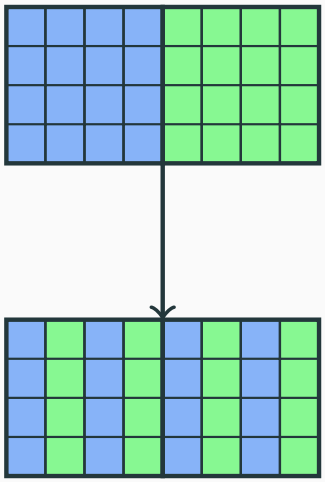
Requirements:

- 256-bit second-preimage resistance resp. 128-bit post-quantum.
- Very efficient.

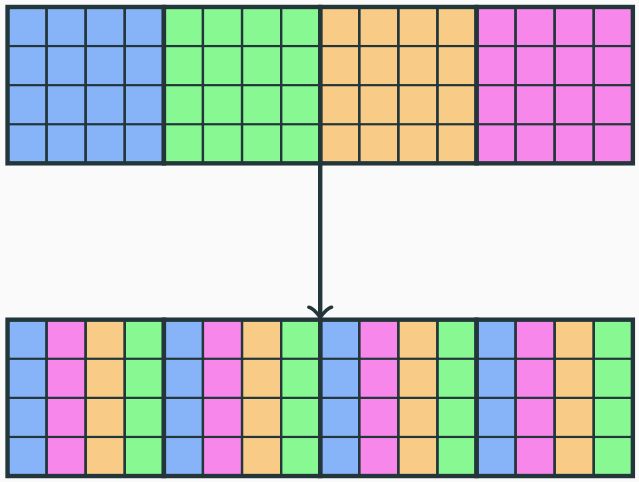
[1] Weakness in first version of Haraka [Jea16]

Mixing should provide good diffusion and be fast:

2 · 128-bit state



4 · 128-bit state



HARAKA - SECURITY

How to determine security for all the different set of parameters?

- Count active S-boxes (using MILP)
  - Attacker can bypass active S-boxes using degrees of freedom.
- Collision resistance implies second-preimage resistance ...
  - Powerful collision attacks exist (Rebound Attack), but we only require second-preimage resistance.
- Other attack vectors like MitM attacks.

Performance Decisions:

- Design is highly performance-motivated
  - Main factors: *state size* and *mixing*: **blend** vs. **mix** vs. **shuffle-xor**
- Make good use of AES-NI pipeline
- Consider P parallel inputs

HARAKA - PERFORMANCE

What can we gain?

- SPHINCS implementation uses highly optimized ChaCha12
- Haraka only slightly faster when parallel inputs are available

	Single input		Multiple inputs	
	Haswell	Skylake	Haswell	Skylake
Haraka-512/256	1.77	0.95	1.42	0.71
ChaCha12	11.58	10.97	1.63	$\approx$ 0.82

HARAKA - PERFORMANCE

SPHINCS using Haraka

- Not all inputs to the hash functions are independent.
- Signing takes roughly half the cycles on Skylake.

	Haswell		Skylake	
SPHINCS-256 w.	ChaCha12	Haraka	ChaCha12	Haraka
Key Generation	3 295 808	2 060 864	2 839 018	1 426 138
Signing	52 249 518	34 938 076	43 517 538	23 312 354
Verify	1 495 416	695 222	1 291 980	452 066

- SPHINCS-256-Haraka not optimized yet
  - 3-fold instead of 8-fold parallelization would be better

CONCLUSION

Haraka

- First design specifically for short-input hashing.
- Below 1 cycle per byte on Skylake for a single block.
- Extremely low latency (only 60 cycles)
- Use in current best SPHINCS implementation
  - Generally around 2x faster
  - Signature Verification is almost 3x faster on Skylake

Paper available at

`https://eprint.iacr.org/2016/098`

Implementation of Haraka and SPHINCS-256-Haraka:

`https://github.com/kste/haraka`

FUTURE WORK

- Analysis questions
  - Less rounds because we don’t need collision resistance, but only 2nd-preimage resistance?
- Implementations
  - ARM implementations of Haraka and SPHINCS-Haraka
  - XMSS implementation with Haraka
  - Benchmark framework
- Design questions
  - Generalizing to other permutation sizes
  - New design at other point in spectrum between AES-mix and AES-Feistel

Questions?

References I


Johannes A. Buchmann, Erik Dahmen, and Andreas Hülsing, *XMSS - A practical forward secure signature scheme based on minimal security assumptions*, Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, 2011, pp. 117–129.


Johannes A. Buchmann, Luis Carlos Coronado García, Erik Dahmen, Martin Döring, and Elena Klintsevich, *CMSS - an improved merkle signature scheme*, Progress in Cryptology - INDOCRYPT 2006, 2006, pp. 349–363.


Daniel J. Bernstein, Daira Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-O'Hearn, *SPHINCS: practical stateless hash-based signatures*, Advances in Cryptology - EUROCRYPT 2015, 2015, pp. 368–397.

References II

- Shay Gueron and Nicky Mouha, *Simpira v2: A Family of Efficient Permutations Using the AES Round Function*, Cryptology ePrint Archive, Report 2016/122, 2016, `http://eprint.iacr.org/`.
- Oded Goldreich, *The foundations of cryptography - volume 2, basic applications*, Cambridge University Press, 2004.
- Andreas Hülsing, *W-OTS+ - shorter signatures for hash-based signature schemes*, Progress in Cryptology - AFRICACRYPT 2013, 2013, pp. 173–188.
- Jérémy Jean, *Cryptanalysis of Haraka*, Cryptology ePrint Archive, Report 2016/396, 2016, `http://eprint.iacr.org/`.
- Jérémy Jean and Ivica Nikolic, *Efficient Design Strategies Based on the AES Round Function*, Fast Software Encryption, FSE 2016, 2016.